Online Learning of Optimal Strategies in Unknown Environments

Santiago Paternain and Alejandro Ribeiro

***Abstract*—Define an environment as a set of convex constraint functions that vary arbitrarily over time and consider a cost function that is also convex and arbitrarily varying. Agents that operate in this environment intend to select actions that are feasible for all times while minimizing the cost's time average. Such action is said optimal and can be computed *offline* if the cost and the environment are known a priori. An *online* policy is one that depends causally on the cost and the environment. To compare online policies to the optimal offline action define the fit of a trajectory as a vector that integrates the constraint violations over time and its regret as the cost difference with the optimal action accumulated over time. Fit measures the extent to which an online policy succeeds in learning feasible actions while regret measures its success in learning optimal actions. This paper proposes the use of online policies computed from a saddle point controller. It is shown that this controller produces policies with bounded regret and fit that grows at a sublinear rate. These properties provide an indication that the controller finds trajectories that are feasible and optimal in a relaxed sense. Concepts are illustrated throughout with the problem of a shepherd that wants to stay close to all sheep in a herd. Numerical experiments show that the saddle point controller allows the shepherd to do so.**

I. Introduction

A shepherd wants to stay close to a herd of sheep while also staying as close as possible to a preferred sheep. The movements of the sheep, are unknown a priori and arbitrary, perhaps strategic. However, their time varying positions are such that it is possible for the shepherd to stay within a prescribed distance of all of them. The shepherd observes the sheep movements and responds to this online information through a causal dynamical system. This paper shows that an online version of the saddle point algorithm of Arrow and Hurwicz [1] succeeds in keeping the shepherd close to all sheep while maintaining a distance to the preferred sheep that is not much worse that the distance he would maintain had he known the sheep's paths a priori. More generically, we consider an agent that operates in an environment that we define as a set of time varying functions of the agent's actions – the distance between the sheep and the shepherd – as well as a cost function that is also time varying and dependent on the agent's actions – the distance to the preferred sheep.

The problem of operating in unknown convex environments with unknown costs generalizes operation in known environments with known costs, which in turn generalizes plain cost minimization. The latter is a canonical problem that can be solved with gradient descent controllers; see e.g., [2]. These algorithms converge to local minima and to the global minimum if the cost is convex. These costs can represent natural constraints or artificial potentials and are common methodologies to solve, e.g., navigation problems [3]–[5].

The novelty of this work is to consider constraints and costs that are unknown a priori and can change arbitrarily over time. In this case, cost minimization can be formulated in the language of regret [6], [7] whereby agents operate online by selecting plays that incur a cost selected by nature. The cost functions are revealed to the agent ex post and used to adapt subsequent plays. It is a remarkable fact that online gradient descent is able to find plays whose regret grows at a sublinear rate when the cost is a convex function [8], [9] – therefore suggesting vanishing per-play penalties of online plays with respect to the clairvoyant play. Our main contribution is to show that an online saddle point algorithm that observes costs and constraints ex post succeeds in finding policies with regret and fit that, at worst, grow at a sublinear rate – and stay bounded with more stringent hypotheses.

The online learning of strategies that are feasible is formulated in the language of fit, defined as the accumulation of the constraint violation over time (Section II). In the main part of the paper we propose to control fit and regret growth with the use of an online saddle point controller that moves along a linear combination of the negative gradients of the constraints and the objective function. The coefficients of these linear combinations are adapted dynamically by the constraint functions as well (Section III). This online saddle point controller is a generalization of (offline) saddle point in the same sense that an online gradient controller generalizes (offline) gradient descent. We show that the online saddle point controller achieves bounded regret and the fit grows sub linearly with the time horizon (Theorem 1). Throughout the paper we illustrate concepts with the problem of a shepherd that has to follow a herd of sheep (Section II-B). A numerical analysis of this problem closes the paper (Section IV).

Notation. A multivalued function $f : \mathbb{R}^n \to \mathbb{R}^m$ is defined by stacking the components functions, i.e., $f := [f_1, \dots, f_m]^T$. The notation $\int f(x)dx := [\int f_1(x)dx, \dots, \int f_m(x)dx]^T$ represents a vector stacking each individual integral. An inequality $x \leq y$ between vectors of equal dimension $x, y \in \mathbb{R}^n$ is interpreted componentwise.

Work in this paper is supported by NSF CCF-0952867 and ONR N00014-12-1-0997. The authors are with the Department of Electrical and Systems Engineering, University of Pennsylvania, 200 South 33rd Street, Philadelphia, PA 19104. Email: {spater, aribeiro}@seas.upenn.edu.

II. Viability, feasibility and optimality

We consider a continuous time environment in which an agent selects an action that results in a time varying set of penalties. Use t to denote time and let $X \subseteq \mathbb{R}^n$ be a closed convex set from which the agent selects action $x \in X$. The penalties incurred at time t for selected action x are given by the value $f(t,x)$ of the vector function $f : \mathbb{R}\times\mathbb{R}^n \to \mathbb{R}^m$. We interpret f as a definition of the environment. We consider situations where the agent is faced with an environment f and must choose a actions $x \in X$ that guarantees nonpositive penalties $f(t,x(t)) \leq 0$ for all times t not exceeding a time horizon T. Since the existence of this trajectory depends on the specific environment we define a viable environment as one in which it is possible to select an action with nonpositive penalty for times $0 \leq t \leq T$ as we formally specify next.

Definition 1 (Viable environment). *We say that a given environment $f : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^m$ is viable over the time horizon T for an agent that selects actions $x \in X$ if there exists an action $x^\dagger \in X$ such that*

$$f(t,x^\dagger) \leq 0, \quad \textit{for all } t \in [0,T]. \tag{1}$$

An action $x^\dagger$ satisfying (1) *is said feasible and the set $X^\dagger := \{x^\dagger \in X : f(t,x^\dagger) \leq 0, \textit{ for all } t \in [0,T]\}$ is termed the feasible set of actions.*

For such environments it is possible to have multiple feasible actions, thus it is desirable to select one that is optimal with respect to some criterion of interest. Introduce then the objective function $f_0 : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}$, where for a given time $t \in [0,T]$ and action $x \in X$ the agent suffers a loss $f_0(t,x)$. The optimal action is defined as the one that minimizes the accumulated loss $\int_0^T f_0(t,x)\,dt$ among all viable actions, i.e.,

$$\begin{aligned} x^* := \operatorname*{argmin}_{x\in X} & \int_0^T f_0(t,x)\,dt \\ \text{s.t. } & f(t,x) \leq 0, \text{ for all } t \in [0,T]. \end{aligned} \tag{2}$$

For the definition in (2) to be valid the function $f_0(t,x)$ has to be integrable with respect to t. In subsequent analyses we also require integrability of the environment f as well as convexity with respect to x as we formally state next.

Assumption 1. The functions $f(t,x)$ and $f_0(t,x)$ are integrable with respect to t in the interval $[0,T]$.

Assumption 2. The functions $f(t,x)$ and $f_0(t,x)$ are convex with respect to x for all times $t \in [0,T]$.

We further require the objective function to be lower bounded. Since the function $f_0(t,x)$ is convex, a lower bound exists if the action space X is bounded, as is the case in most applications of practical interest.

Assumption 3. The objective functions $f_0(t,x)$ are lower bounded on the action space X. I.e., there is a finite constant K independent of the time horizon T such that

$$K \geq f_0(t,x) - \min_{x\in X} f_0(t,x). \tag{3}$$

If the environment f and function f_0 are known, finding the action in a viable environment that minimizes the aggregate cost is equivalent to solve (2). A number of algorithms are known to solve this problem. Here, we consider the problem of adapting a strategy $x(t)$ when the functions $f(t,x)$ and $f_0(t,x)$ are *arbitrary* and *revealed causally*. I.e., we want to choose the action $x(t)$ using observations of viability $f(t,x)$ and cost $f_0(t,x)$ in the open interval $[0,t)$. This implies that $f(t,x(t))$ and $f_0(t,x(t))$ are not observed before choosing $x(t)$.

A. Regret and fit

We evaluate the performance of trajectories $x(t)$ through the concepts of regret and fit. To define regret we compare the accumulated cost $\int_0^T f_0(t,x(t))\,dt$ incurred by $x(t)$ with the cost that would had been incurred by the optimal action x^* defined in (2),

$$\mathcal{R}_T := \int_0^T f_0(t,x(t))\,dt - \int_0^T f_0(t,x^*)\,dt. \tag{4}$$

Analogously, we define the fit of the trajectory $x(t)$ as the accumulated penalties $f(t,x(t))$ incurred for times $t \in [0,T]$,

$$\mathcal{F}_T := \int_0^T f(t,x(t))\,dt. \tag{5}$$

The regret $\mathcal{R}_T$ and fit $\mathcal{F}_T$ can be interpreted as performance losses associated with online causal operation as opposed to offline clairvoyant operation. If the fit $\mathcal{F}_T$ is positive in a viable environment we are in a situation in which, had the environment f be known, we could have selected an action $x^\dagger$ with $f(t,x^\dagger) \leq 0$. The fit measures how far the trajectory $x(t)$ comes from achieving that goal. Likewise, if the regret $\mathcal{R}_T$ is positive having prior knowledge of environment and cost would had resulted in the selection of a better action x^* – and in that sense $\mathcal{R}_T$ indicates how much we regret not having had that information available. A good learning strategy is one in which $x(t)$ approaches x^*. In that case, the regret and fit grow for small T but eventually stabilize or, at worst, grow at a sublinear rate. Considering regret $\mathcal{R}_T$ and fit $\mathcal{F}_T$ separately, this observation motivates the definitions of feasible trajectories and strong optimal trajectories that we formally state next.

Definition 2. *Given an environment $f : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^m$, a cost $f_0 : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}$, and a trajectory $x(t)$ we say that*

Feasibility. *The trajectory $x(t)$ is feasible in the environment if the fit $\mathcal{F}_T$ grows sublinearly with T. I.e., if there exist a function $h(T)$ with $\limsup_{T\to\infty} h(T)/T = 0$ and a constant vector C such that for all times T it holds,*

$$\mathcal{F}_T := \int_0^T f(t,x(t))\,dt \leq Ch(T). \tag{6}$$

Strong optimality. *The trajectory $x(t)$ is strongly optimal in the environment if the regret $\mathcal{R}_T$ is bounded for all T. I.e., if there exists a constant C such that for all times T it holds,*

$$\mathcal{R}_T := \int_0^T f_0(t,x(t))\,dt - \int_0^T f_0(t,x^*)\,dt \leq C. \tag{7}$$

In this work we solve the problem of finding feasible, strongly optimal trajectories. We develop this solutions in section III. Before that, we clarify concepts with the introduction of an example.

B. The shepherd problem

Consider a target tracking problem in which an agent – the shepherd – follows m targets – the sheep. Specifically, let $z(t) = [z_1(t), z_2(t)]^T \in \mathbb{R}^2$ denote the position of the shepherd at time t. To model smooth paths write each position component $z_k(t)$ as

$$z_k(t) = \sum_{j=0}^{n-1} x_{kj} p_j(t), \tag{8}$$

where $p_j(t)$ are time polynomials that parameterize the space of possible trajectories. The action space is then given by the vector $x = [x_{10}, \dots, x_{1,n-1}, x_{20}, \dots, x_{2,n-1}]^T \in \mathbb{R}^{2n}$ that stacks the coefficients of the parameterization in (8).

Further define $y_i(t) = [y_{i1}(t), y_{i2}(t)]^T$ as the position of the ith sheep at time t for $i = 1, \dots, m$ and introduce a maximum allowable distance r_i between the shepherd and each of the sheep . The goal of the shepherd is to find a path $z(t)$ that is within distance r_i of sheep i for all sheep. This can be captured by defining an m-dimensional environment f with each component function f_i defined as

$$f_i(t, x) = \|z(t) - y_i(t)\|^2 - r_i^2 \quad \text{for all} \quad i = 1..m. \tag{9}$$

That the environment defined by (9) is viable means that it is possible to select a vector of coefficients x so that the shepherd's trajectory generated by (8) stays close to all sheep for all times. To the extent that (8) is a loose parameterization – we can approximate arbitrary functions with sufficiently large index n –, this simply means that the sheep are sufficiently close to each other at all times.

Say that the first target – the black sheep – is preferred in that the shepherd wants to stay as close as possible to it. Introduce then the objective function

$$f_0(t, x) = \|z(t) - y_1(t)\|^2. \tag{10}$$

Alternatively, we can require the shepherd to minimize the work required to follow the sheep. This behavior can be induced by minimizing the integral of the acceleration which in turn can be accomplished by defining the optimality criterion [cf. (2)],

$$f_0(t, x) = \left\|\ddot{z}(t)\right\| = \left\| \left[\sum_{j=0}^{n-1} x_{1j} \ddot{p}_j(t),\ \sum_{j=0}^{n-1} x_{2j} \ddot{p}_j(t) \right] \right\|. \tag{11}$$

Trajectories $x(t)$ differ from actions in that they are allowed to change over time, i.e., the constant values x_{kj} in (8) are replaced by the time varying values $x_{kj}(t)$. A feasible trajectory $x(t)$ means that the shepherd is repositioning to stay close to all sheep. An optimal trajectory with respect to (10) is one in which he does so while staying as close as possible to the black sheep. An optimal trajectory with respect to (11) is one in which the work required to follow the sheep is minimized.

III. Saddle point algorithm

Given an environment $f(t, x)$ and an objective function $f_0(t, x)$ verifying assumptions 1 and 2 we set our attention towards two different problems: design a controller that gives origin to strongly feasible trajectories and a controller that gives origin to feasible and strongly optimal trajectories. As already noted, when the environment is known beforehand the problem of finding such trajectories is a constrained convex optimization problem, which we can solve using the saddle point algorithm of Arrow and Hurwicz [1]. Following this idea, let $\lambda \in \Lambda = \mathbb{R}^m_+$, be a multiplier and define the time-varying Lagrangian associated with the online problem as

$$\mathcal{L}(t, x, \lambda) = f_0(t, x) + \lambda^T f(t, x). \tag{12}$$

Saddle point methods rely on the fact that for a constrained convex optimization problem, a pair is a primal-dual optimal solution if and only if the pair is a saddle point of the Lagrangian associated with the problem; see e.g. [10]. The main idea of the algorithm is then to generate trajectories that descend in the opposite direction of the gradient of the Lagrangian with respect to x and that ascend in the direction of the gradient with respect to λ. To avoid restricting attention to functions that are differentiable with respect to x, we introduce the notion of subgradient.

Definition 3 (Subgradient). *Let $g : X \to \mathbb{R}$, be a convex function where $X \subset \mathbb{R}^n$. Then g_x is a subgradient of g at a point $x \in X$ if*

$$g(y) \geq g(x) + g_x(x)^T (y - x) \quad \textit{for all} \quad y \in X \tag{13}$$

In general, subgradients are defined at all points for all convex functions. At the points where the function f is differentiable the subgradient and the gradient coincide. In the case of vector functions $f : \mathbb{R}^n \to \mathbb{R}^m$ we group the subgradients of each component into a subgradient matrix $f_x(x) \in \mathbb{R}^{n \times m}$ that we define as

$$f_x(x) = \begin{bmatrix} f_{1,x}(x) & f_{2,x}(x) & \cdots & f_{m,x}(x) \end{bmatrix} \tag{14}$$

where $f_{i,x}(x)$ is a subgradient of $f_i(x)$ as per Definition 3. In addition, since the action must always be selected from the set X and the multipliers have to be in the positive orhtant we define the controller in a way that the actions and the multipliers are the solution of a projected dynamical system over the set $X \times \Lambda$. The solution has been studied in [11], [12] and we define the notion as follow.

Definition 4 (Projected dynamical system)**.** *Let X be a closed convex set.*

Projection of a point. *For any $z \in R^n$, there exits a unique element in X, denoted $P_X(z)$ such that*

$$P_X(z) = \arg \inf_{y \in X} \|y - z\|. \tag{15}$$

Projection of a vector at a point. *Let $x \in X$ and v a vector, we define the projection of v over the set X at the point x, $\Pi_X(x, v)$ as*

$$\Pi_X(x, v) = \lim_{\delta \to 0^+} \left(P_X(x + \delta v) - x\right) / \delta. \tag{16}$$

Projected dynamical system. *Given a closed convex set X and a vector field $F(t,x)$ which takes elements from $\mathbb{R}\times X$ into $\mathbb{R}^n$ the projected differential equation associated with X and F is defined to be*

$$\dot{x}(t) = \Pi_X\left(x, F(t,x)\right). \tag{17}$$

In the above projection if the point x is in the interior of X then the projection is equal to the original vector field i.e. $\Pi_X(x,F(t,x)) = F(t,x)$. On the other hand if x is in the border of X, then the projection is just the component of the vector field that is tangential to the set X at the point x.

Since the Lagrangian is differentiable with respect to λ, we denote by $\mathcal{L}_\lambda(t,x,\lambda) = f(t,x)$ the derivative of the Lagrangian with respect to λ. On the other hand, since the functions $f_0(\cdot,x)$ and $f(\cdot,x)$ are convex, the Lagrangian is also convex with respect to x. Thus, its subgradient with respect to x always exist, let us denote it by $\mathcal{L}_x(t,x,\lambda)$. Let ε be the gain of the controller, then following the ideas in [1] we define a controller that descends in the direction of the subgradient with respect to the action x

$$\begin{aligned}\dot{x} &= \Pi_X\left(x, -\varepsilon\mathcal{L}_x(t,x,\lambda)\right)\\ &= \Pi_X\left(x, -\varepsilon(f_{0,x}(t,x) + f_x(t,x)\lambda)\right),\end{aligned} \tag{18}$$

and that ascends in the direction of the subgradient with respect to the multiplier λ

$$\dot{\lambda} = \Pi_\Lambda\left(\lambda, \varepsilon\mathcal{L}_\lambda(t,x,\lambda)\right) = \Pi_\Lambda\left(\lambda, \varepsilon f(t,x)\right). \tag{19}$$

The projection over the set X in (18) is done to assure that the trajectory is always in the set of possible actions. The projection concerning the dual variable λ in (19) is done to assure that $\lambda(t)\in\mathbb{R}^m_+$ for all times $t\in[0,T]$. An important observation regarding (18) and (19) is that the environment is observed locally in space and causally in time. The values of the environment constraints and its subgradients are observed at the current trajectory position $x(t)$ and the values of $f(t,x(t))$ and $f_x(t,x(t))$ affect the derivatives of $x(t)$ and $\lambda(t)$ only.

A block diagram for the controller in (18) - (19) is shown in Figure 1. The controller operates in an environment to which it inputs at time t an action $x(t)$ that results in a penalty $f(t,x(t))$ and cost $f_0(t,x(t))$. The value of these functions and their subgradients $f_x(t,x(t))$ and $f_{0,x}(t,x(t))$ are observed and fed to the multiplier and action feedback loops. The action feedback loop behaves like a weighted gradient descent controller. We move in the direction given by a linear combination of the the gradient of the objective function $f_{0,x}(t,x(t))$ and the constraint subgradients $f_i(t,x(t))$ weighted by their corresponding multipliers $\lambda_i(t)$. Intuitively, this pushes $x(t)$ towards the minimum of the objective function in the set where constraints are satisfied. However, the question remains of how much weight to give to each constraint. This is the task of the multiplier feedback loop. When constraint i is violated we have $f_i(t,x(t))>0$. This pushes the multiplier $\lambda_i(t)$ up, thereby increasing the force $\lambda_i(t)f_i(t,x(t))$ pushing $x(t)$ towards satisfying the constraint. If the constraint is satisfied, we have $f_i(t,x(t))<0$, the multiplier $\lambda_i(t)$ being decreased, and the corresponding force decreasing. The more that constraint i is violated, the faster the multiplier increases, and the more the force that pushes $x(t)$ towards satisfying $f_i(t,x(t))<0$ is increased. If the constraint is satisfied, the force is decreased and may eventually vanish if we reach the point of making $\lambda_i(t)=0$.

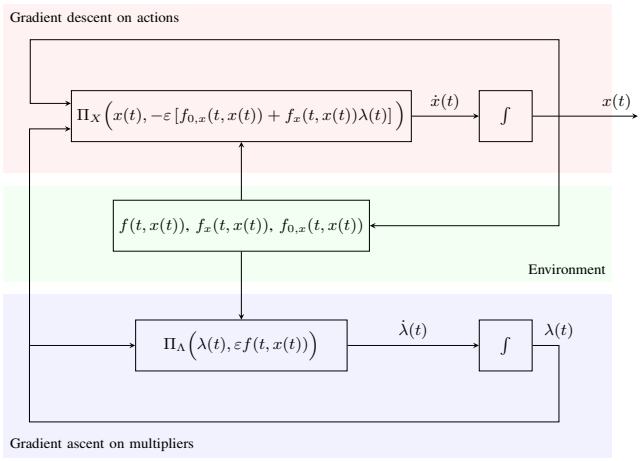

Fig. 1: Block diagram of the saddle point controller. Once that action $x(t)$ is selected at time t, we measure the corresponding values of $f(t,x)$, $f_x(t,x)$ and $f_{0,x}(t,x)$. This information is fed to the two feedback loops. The action loop defines the descent direction by computing weighted averages of the subgradients $f_x(t,x)$ and $f_{0,x}(t,x)$. The multiplier loop uses $f(t,x)$ to update the corresponding weights.

This section presents bounds on the growth of the fit and the regret of the trajectories $x(t)$ generated by the saddle point controller defined by (18) and (19). These bounds ensure that the trajectory is feasible and strongly optimal in the sense of Definition 2. Those bounds depend on the value of the following energy function. Consider an arbitrary fixed action $\bar{x}\in X$ and multiplier $\bar{\lambda}\in\Lambda$ and let

$$V_{\bar{x},\bar{\lambda}}(x,\lambda) = \frac{1}{2}\left(\|x-\bar{x}\|^2 + \|\lambda-\bar{\lambda}\|^2\right). \tag{20}$$

Using the constant in (3) and the definition of the energy function in (20) we can write down regret and fit bounds for an action trajectory $x(t)$ that follows the saddle point dynamics defined by (18) and (19). We state these bounds in the following theorem.

Theorem 1. *Let $f:\mathbb{R}\times X\to\mathbb{R}^m$ and $f_0:\mathbb{R}\times X\to\mathbb{R}$, where f and f_0 and 3 where $X\subset\mathbb{R}^n$ is a convex set. If the environment is viable, then the controller defined by* (18) *and* (19) *produces trajectories $x(t)$ that are feasible and strongly optimal for all time horizons $T>0$. In particular, the fit is bounded by*

$$\mathcal{F}_{T,i} \le \left(\frac{1}{\varepsilon}V_{x^*,\left[\int_0^T f(t,x)\,dt\right]^+}(x(0),\lambda(0)) + KT\right)^{1/2}, \tag{21}$$

and the regret is bounded by

$$\mathcal{R}_T \le \frac{1}{\varepsilon}V_{x*,0}\left(x(0),\lambda(0)\right), \tag{22}$$

where $V_{\bar{x},\bar{\lambda}}(x,\lambda)$ is the energy function defined in (20)*, x^* is the solution to the problem in* (2) *and K is the constant defined in* (3)*.*

Proof. See Appendix A ∎

Theorem 1 assures that if an environment is viable for an agent that selects actions over a set X, the controller defined by (18) and (19) gives origin to a trajectory $x(t)$ that is feasible and strongly optimal in the sense of Definition 2. This result is not trivial, since the function f that defines the environment is observed causally and can change arbitrarily over time. In particular, the agent could be faced with an adversarial nature that changes the function f and f_0 in a way that makes the value of $f(t,x(t))$ and $f_0(t,x(t))$ larger. The caveat is that the choice of the function f must respect the viability condition that there exists a feasible action $x^\dagger$ such that $f(t,x^\dagger)\leq 0$ for all $t\in[0,T]$. This restriction still leaves significant leeway for strategic behavior. E.g., in the shepherd problem of Section II-B we can allow for strategic sheep that observe the shepherd's movement and respond by separating as much as possible. The strategic action of the sheep are restricted by the condition that the environment remains viable, which in this case reduces to the condition that the sheep stay in a ball of radius $2r$ if all $r_i = r$.

Further note that, the initial value of the energy function used to bound both regret and fit is related with the square of the distance between the initial action and the optimal offline solution of problem (2). Therefore, the closer we start from this action the smaller the bound of regret and fit will be. Likewise, the larger the gain ε, the smaller the regret bound is. This is not possible in practice because larger ε entails trajectories with larger derivatives which cannot be implemented in systems with physical constraints. In the example in Section II-B the derivatives of the state $x(t)$ control the speed and acceleration of the shepherd. Notice however that the fit cannot be made arbitrarily small and that it grows at least as $\sqrt{KT}$.

IV. Numerical experiments

We evaluate performance of the saddle point algorithm defined by (18)-(19) in the solution of the shepherd problem introduced in Section II-B. We determine sheep paths using a perturbed polynomial characterization akin to the one in (8). Specifically, letting $p_j(t)$ be elements of a polynomial basis, the k-th component, with $k=1,2$, of the path $y_i(t)=[y_{i1}(t),y_{i2}(t)]^T$ followed by the ith sheep is given by

$$y_{ik}(t)=\sum_{j=0}^{n_i-1} y_{ikj}p_j(t)+w_{ik}(t), \tag{23}$$

where n_i denotes the total number of polynomials that parameterize the path followed by sheep i, and y_{ikj} represent the corresponding n_i coefficients. The noise terms $w_{ik}(t)$ are Gaussian white with zero mean, standard deviation σ and independent across components and sheep.

To determine y_{ikj} we make $w_{ik}(t)=0$ in (23) and require all sheep to start at position $y_i(0)=[0,0]^T$ and finish at position $y_i(T)=[1,1]^T$. A total of L random points $\{\tilde{y}_l\}_{l=1}^L$ are drawn independently and uniformly at random in $[0,1]^2$. Sheep $i=1$ is required to pass trough points $\tilde{y}_l$ at times $lT/(L+1)$, i.e., $y_1(lT/(L+1))=\tilde{y}_l$. For each of the other sheep $i\neq 1$ we draw L random offsets $\{\Delta\tilde{y}_{il}\}_{l=1}^L$ uniformly at random from $[-\Delta,\Delta]^2$ and require the ith sheep path to satisfy $y_i(lT/(L+1))=\tilde{y}_l+\Delta\tilde{y}_{il}$. Paths $y_i(t)$ are then chosen as those that minimize the path integral of the acceleration squared subject to the constraints of each individual path, i.e.,

$$\begin{aligned} y_i^* = \operatorname{argmin} & \int_0^T \|\ddot{y}_i(t)\|^2 dt, \\ \text{s.t.} \quad & y_i(0)=[0,0]^T, \quad y_i(T)=[1,1]^T, \\ & y_i(lT/(L+1))=\tilde{y}_l+\Delta\tilde{y}_{il}, \end{aligned} \tag{24}$$

where, by construction $\Delta\tilde{y}_{il}=0$ for $i=1$. The problem (24) can be solved as a quadratic program [14]. Let $y_i^*(t)$ be the trajectory given by (23) when we set $y_{ikj}=y_{ikj}^*$. We obtain the paths $y_{ik}(t)$ by adding $w_{ik}(t)$ to $y_i^*(t)$.

In subsequent numerical experiments we consider $m=5$ sheep, a time horizon $T=1$, and set the proximity constraint in (9) to $r_i=0.3$. We use the standard polynomial basis $p_j(t)=t^j$ in both, (8) and (23). The number of basis elements in both cases is set to $n=n_i=30$. To generate sheep paths we consider a total of $L=3$ randomly chosen intermediate points, set the variation parameter to $\Delta=0.1$, and the perturbation standard deviation to $\sigma=0.1$. These problem parameters are such that the environment is most likely viable in the sense of Definition 1. We check that this is true by solving the offline feasibility problem. If the environment is not viable a new one is drawn before proceeding to the implementation of (18)-(19).

We emphasize that even if the trajectory of the sheep is known to us, the information is not used by the controller. The controller is fed information of the position of the sheep at the current time, which it uses to evaluate the environment functions $f_i(t,x)$ in (9), their gradients $f_{ix}(t,x)$ and the gradient of $f_0(t,x)$. In Section IV-A $f_0(t,x)$ takes the form of (10) while in Section IV-B takes the form of (11).

A. Preferred sheep problem

Besides satisfying the constraints defined in (9), the shepherd in being as close as possible from the first sheep. This translates into the optimality criterion defined in (10). Since we construct sheep trajectories that are viable the hypotheses of Theorem 1 hold. Thus, if the shepherd follows the dynamics described by (18) and (19), the resulting action trajectory is feasible and strongly optimal.

Since the trajectory is feasible, we expect the fit to be bounded by a sublinear function of T. This does happen, as can be seen in Figure 3 where a gain $\varepsilon=50$ is used. In fact, the fit does not grow and is bounded by a constant for all time horizons T. The trajectory is therefore strongly feasible. This does not contradict Theorem 1 because strong feasibility implies feasibility. The reason why it's reasonable to see bounded fit here is that the objective function pushing the

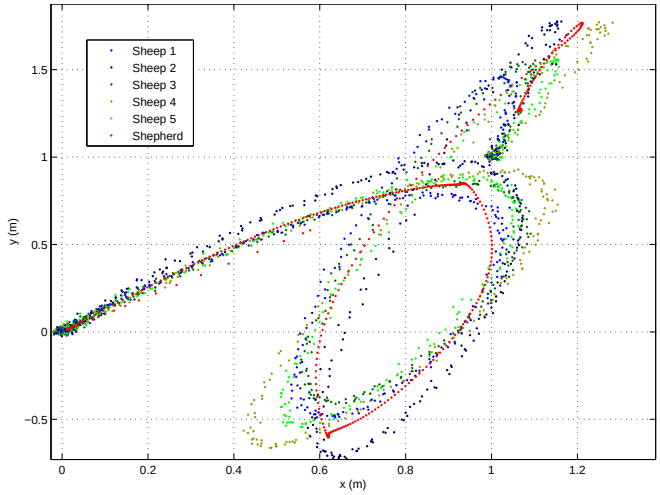

Fig. 2: Path of the sheep and the shepherd for the preferred sheep problem (Section IV-A) when the gain of the saddle point controller is set to be $\varepsilon = 50$. The shepherd succeed in following the herd since its path – in red – is close to the path of all sheep.

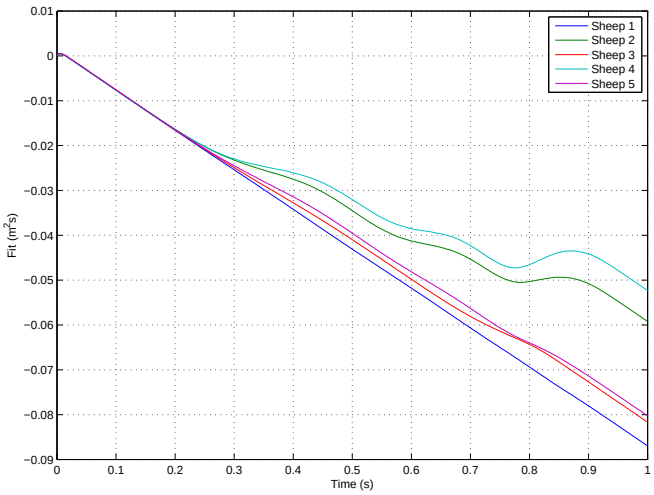

Fig. 3: Fit $\mathcal{F}_T$ for the preferred sheep problem (Section IV-A) when the gain of the saddle point controller is set to be $\varepsilon = 50$. As predicted by Theorem 1 the trajectory is feasible since the fit is bounded, and, in fact, appears to be strongly feasible.

shepherd closer to the sheep is redundant with the constraints that push the shepherd to stay closer to all sheep. The regret trajectory for this experiment with $\varepsilon = 50$ is shown in Figure 4. Since the trajectory is strongly optimal as per Theorem 1, we expect regret to be bounded. This is the case in Figure 4 where regret is actually negative for all times $t \in [0, T]$. Negative regret implies that the trajectory of the shepherd is incurring a total cost that is smaller than the one associated with the optimal solution. Notice that while the optimal fixed action minimizes the total cost as defined in (2) it does not minimize the objective at all times. Thus, by selecting different actions the shepherd can suffer smaller instantaneous losses than the ones associated with the optimal action. If this is the case, regret – which is the integral of the difference between these two losses – can be negative.

B. Minimum acceleration problem

We consider an environment defined by the distances between the shepherd and the sheep given by (9) and the minimum acceleration objective defined in (11). Since the construction of the target trajectories gives a viable environment we satisfy, the hypotheses of Theorem 1. Hence, for a shepherd following the dynamics given by (18) and (19), the action trajectory is feasible and strongly optimal. For the simulation the gain of the controller is set to $\varepsilon = 50$.

A feasible trajectory implies that the fit must be bounded by a function that grows sub linearly with the time horizon T. Notice that this is the case in Figure 6. Periods of growth are observed, yet the presence of inflection points is an evidence of the growth being controlled. The fit in this problem is larger than in problem IV-A (c.f figures 3 and 6). This result is predictable since the constraints and the objective push the action in different directions. For instance, suppose that all constraints are satisfied and that the Lagrange multipliers are zero. Then, the subgradient of the Lagrangian is equal to the subgradient of the objective function. Hence the action will be modified trying to minimize the acceleration without taking the constraints (distance with the sheep) into account. Hence, pushing the action to the boundary of the feasible set. In this problem, this translates into the fact that the shepherd does not follow the sheep as closely as in the problem in section IV-A (c.f Figure 5).

Since the trajectory is strongly optimal, we should observe a regret bounded by a constant. This is the case in Figure 7. Notice that regret increases since the initial action differs from the optimal. However, as in the case of the fit, the inflection point at the end of the simulation is the evidence that the regret is being controlled. Compared with the regret of the black sheep problem (c.f Figure 4), the regret in this problem is larger. This is again explained by the fact that in this problem objective and constraints can push the action in different directions while in Section IV-A both point in the same general direction.

V. Conclusion

We considered a continuous time environment in which an agent must select actions to satisfy a set of constraints. These constraints are time varying and the agent does not have information regarding their future evolution. We defined a viable environment as one in which there is a fixed action that verifies all the constraints at all times. An objective function was considered as well to select a strategy that meets an optimality criterion from the set of strategies that satisfy the constraints. We proposed an online version of the saddle point controller of Arrow-Hurwicz to generate trajectories with small fit and regret. We showed that for any viable environment the trajectories that follow the dynamics of this controller are feasible and strongly optimal. Numerical experiments on a shepherd that tries to follow a herd of sheep support these theoretical results.

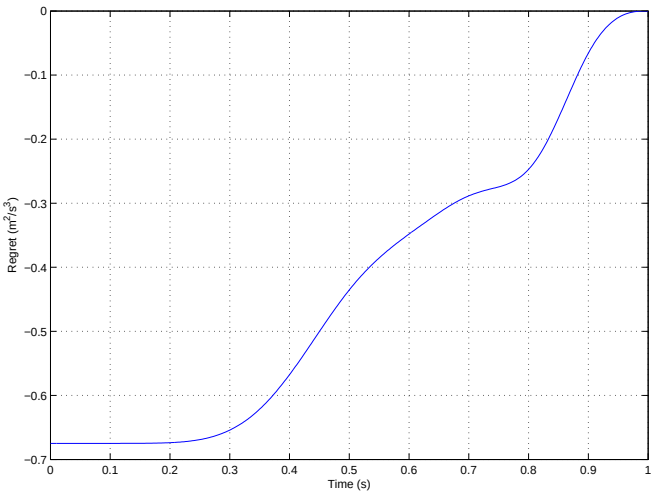

Fig. 4: Regret $\mathcal{R}_T$ for the preferred sheep problem (Section IV-A) when the gain of the saddle point controller is set to be $\varepsilon = 50$. The trajectory is strongly feasible, as predicted by Theorem 1.

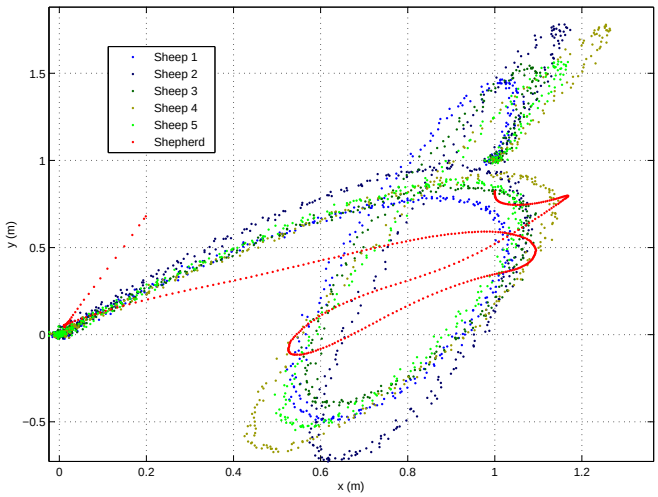

Fig. 5: Path of the sheep and the shepherd for the minimum acceleration problem (Section IV-B) when the gain of the saddle point controller is set to be $\varepsilon = 50$. Observe that the shepherd path – in red – is not as close to the path of the sheep as in Figure 2. This is reasonable because the objective function and the constraints push the shepherd in different directions.

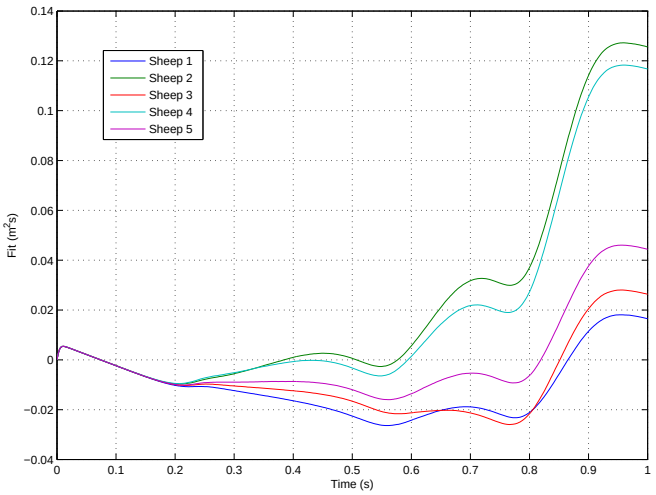

Fig. 6: Fit $\mathcal{F}_T$ for the minimum acceleration problem (Section IV-B) when the gain of the saddle point controller is set to $\varepsilon = 50$. Since the fit is bounded, the trajectory is feasible in accordance with Theorem 1. Since the gradient of the objective function and the gradient of the feasibility constraints tend to point in different directions, the fit is larger than in Section IV-A (c.f Figure 3).

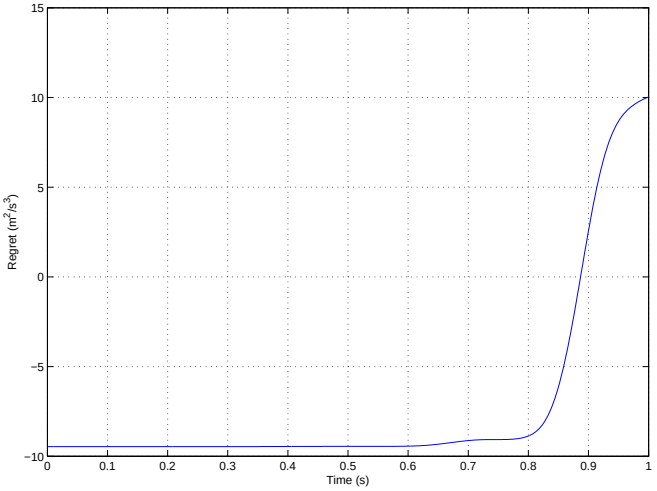

Fig. 7: Regret $\mathcal{R}_T$ for the minimum acceleration problem (Section IV-B) when the gain of the saddle point controller is set to be $\varepsilon = 50$. The trajectory is strongly optimal as predicted by Theorem 1. Since the gradient of the objective function and the gradient of the feasibility constraints tend to point in different directions, regret is larger than the regret of the preferred sheep problem (c.f Figure 3).

APPENDIX

A. Proof of Theorem 1

Let us state the following lemma, needed in the proof of Theorem1, concerning the projection of a vector over a set.

Lemma 1. *Let X be a convex set and $x_0, x \in X$. Then*

$$(x_0 - x)^T \Pi_X(x_0, v) \leq (x_0 - x)^T v. \tag{25}$$

Proof. See Lemma 1 [13] ■

Consider action trajectories $x(t)$ and multiplier trajectories $\lambda(t)$ and the energy function $V_{\bar{x},\bar{\lambda}}(x(t), \lambda(t))$ in (20), for arbitrary given action $\bar{x} \in \mathbb{R}^n$ and multiplier $\bar{\lambda} \in \Lambda$. The derivative of the energy with respect to time is then given by

$$\dot{V}_{\bar{x},\bar{\lambda}}(x(t), \lambda(t)) = (x(t) - \bar{x})^T \dot{x}(t) + (\lambda(t) - \bar{\lambda})^T \dot{\lambda}(t). \tag{26}$$

If the trajectories $x(t)$ and $\lambda(t)$ follow from the saddle point dynamical system defined by (18) and (19) respectively we can substitute the action and multiplier derivatives by their corresponding values and reduce (26) to

$$\dot{V}_{\bar{x},\bar{\lambda}}(x(t), \lambda(t)) = (x(t) - \bar{x})^T \Pi_X(x, -\varepsilon(f_{0,x}(t, x(t)) + f_x(t, x(t))\lambda(t)) + (\lambda(t) - \bar{\lambda})^T \Pi_\Lambda(x, \varepsilon f(t, x(t))). \tag{27}$$

Then, in virtue of Lemma 1 we have that

$$\dot{V}_{\bar{x},\bar{\lambda}}(x(t), \lambda(t)) \leq \varepsilon[-(x(t) - \bar{x})^T (f_{0,x}(t, x(t)) + f_x(t, x(t))\lambda(t)) + (\lambda(t) - \bar{\lambda})^T f(t, x(t))]. \tag{28}$$

Notice that $\mathcal{L}(t,x(t),\lambda(t)) = f_0(t,x(t)) + \lambda(t)^T f(t,x(t))$ is a convex function with respect to the actions since it is a sum of convex functions with respect to x. Then, using the definition of subgradient (c.f. Definition 3) we can upper bound the inner product

$$\begin{aligned} -(x(t)-\bar{x})^T(f_{0,x}(t,x(t)) + f_x(t,x(t))\lambda(t)) \\ = -(x(t)-\bar{x})^T \mathcal{L}_x(t,x(t),\lambda(t)) \end{aligned} \tag{29}$$

by the difference $\mathcal{L}(t,\bar{x},\lambda(t)) - \mathcal{L}(t,x(t),\lambda(t))$. Then, we can upper bound the right hand side of the equation 28 and obtain

$$\begin{aligned} \dot{V}_{\bar{x},\bar{\lambda}}(x(t),\lambda(t)) \leq \varepsilon[f_0(t,\bar{x}) + \lambda^T(t)f(t,\bar{x}) - f_0(t,x(t)) \\ - \lambda^T(t)f(t,x(t)) + (\lambda(t)-\bar{\lambda})^T f(t,x(t))]. \end{aligned} \tag{30}$$

Notice that on the right hand side of the above inequality the fourth and the fifth term cancel out, then it reduces to

$$\begin{aligned} \dot{V}_{\bar{x},\bar{\lambda}}(x(t),\lambda(t)) \leq \varepsilon[f_0(t,\bar{x}) + \lambda^T(t)f(t,\bar{x}) \\ -f_0(t,x(t)) - \bar{\lambda}^T f(t,x(t))]. \end{aligned} \tag{31}$$

Rewriting the above equation and then integrating both sides with respect to the time from time $t=0$ to $t=T$, we obtain

$$\begin{aligned} \int_0^T f_0(t,x(t)) - f_0(t,\bar{x}) + \bar{\lambda}^T f(t,x(t)) - \lambda^T(t)f(t,\bar{x})dt \\ \leq -\frac{1}{\varepsilon}\int_0^T \dot{V}_{\bar{x},\bar{\lambda}}(x(t),\lambda(t))dt. \end{aligned} \tag{32}$$

Since the energy function (20) is positive, the above equation reduces to

$$\begin{aligned} \int_0^T f_0(t,x(t)) - f_0(t,\bar{x}) + \bar{\lambda}^T f(t,x(t)) - \lambda^T(t)f(t,\bar{x})dt \\ \leq \frac{1}{\varepsilon} V_{\bar{x},\bar{\lambda}}(x(0),\lambda(0)). \end{aligned} \tag{33}$$

Since (33) holds for any $\bar{x} \in X$ and any $\bar{\lambda} \in \Lambda$, it holds for the particular choice $\bar{x} = x^*$, $\bar{\lambda} = 0$. Since $\lambda^T(t)f(t,x^*)\,dt \leq 0 \quad \forall t \in [0,T]$ we can lower bound the left hand side of (33) to obtain:

$$\int_0^T f_0(t,x(t)) - f_0(t,x^*)dt \leq \frac{1}{\varepsilon} V_{x^*,0}(x(0),\lambda(0)). \tag{34}$$

Notice that the left hand side of the above equation is the definition of regret given in 4. Thus, we have shown that the upper bound for the regret is the one stated in (22). And since the right hand side of the above equation is a constant for all $T > 0$ we proved that the trajectory generated by the saddle point controller is strongly optimal. It remains to prove that the trajectory generated is feasible. In order to do so, choose $\bar{x} = x^*$, and use the Assumption 3, to transform (33) into

$$\begin{aligned} \int_0^T \bar{\lambda}^T f(t,x(t)) - \lambda^T(t)f(t,x^*)\,dt \\ \leq \frac{1}{\varepsilon} V_{x^*,\bar{\lambda}}(x(0),\lambda(0)) + KT. \end{aligned} \tag{35}$$

Since $\lambda^T(t)f(t,x^*)\,dt \leq 0 \quad \forall t \in [0,T]$ we can again lower bound the left hand side of the above equation by $\bar{\lambda}^T \int_0^T f(t,x(t))$ and obtain

$$\bar{\lambda}^T \int_0^T f(t,x(t))dt \leq \left(V_{x^*,\bar{\lambda}}(x(0),\lambda(0))\right)/\varepsilon + KT. \tag{36}$$

Now let's choose $\bar{\lambda} = \left[\int_0^T f(t,x(t))\,dt\right]^+$. The projection on the positive orthant is needed because $\bar{\lambda} \in \mathbb{R}^m_+$. Let $I = \{i = 1..m | \int_0^T f_i(t,x(t))\,dt \geq 0)\}$. Notice that if $i \notin I$, then $\bar{\lambda}_i \int_0^T f_i(t,x(t))\,dt = 0$. On the other hand, if $i \in I$, $\bar{\lambda}_i \int_0^T f_i(t,x(t))\,dt = \left(\int_0^T f_i(t,x(t))\,dt\right)^2 \geq 0$. Therefore, for all $i \in I$ we have:

$$\begin{aligned} \left(\int_0^T f_i(t,x(t))\,dt\right)^2 \\ \leq \frac{1}{\varepsilon} V_{x^*,\left[\int_0^T f(t,x)\,dt\right]^+}(x(0),\lambda(0)) + KT. \end{aligned} \tag{37}$$

Notice that, the left hand side of the above equation is the square of the ith component of the fit. Thus for all $i \in I$

$$\mathcal{F}_{T,i} \leq \left(\frac{1}{\varepsilon} V_{x^*,\left[\int_0^T f(t,x)\,dt\right]^+}(x(0),\lambda(0)) + KT\right)^{1/2}. \tag{38}$$

If $i \notin I$ then $\mathcal{F}_{T,i} < 0$ therefore it is also smaller than the bound in (38). Which proves that the trajectories generated by the saddle point controller defined by (18) and (19) are feasible.

REFERENCES

[1] K. J. Arrow and L. Hurwicz, *Studies in linear and nonlinear programming*. CA: Stanford University Press, 1958.

[2] M. W. Hirsch, S. Smale, and R. L. Devaney, *Differential equations, dynamical systems, and an introduction to chaos*, vol. 60. Academic press, 2004.

[3] E. Rimon and D. E. Koditschek, "Exact robot navigation using artificial potential functions," *Robotics and Automation, IEEE Transactions on*, vol. 8, no. 5, pp. 501–518, 1992.

[4] C. W. Warren, "Global path planning using artificial potential fields," in *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on*, pp. 316–321, IEEE, 1989.

[5] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. Rob. Res.*, vol. 5, pp. 90–98, Apr. 1986.

[6] S. Shalev-Shwartz, "Online learning and online convex optimization," *Foundations and Trends in Machine Learning*, vol. 4, no. 2, pp. 107–194, 2011.

[7] V. Vapnik, *The nature of statistical learning theory*. Springer, 2000.

[8] M. Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *ICML*, pp. 928–936, 2003.

[9] E. Hazan, A. Agarwal, and S. Kale, "Logarithmic regret algorithms for online convex optimization," *Machine Learning*, vol. 69, no. 2-3, pp. 169–192, 2007.

[10] D. P. Bertsekas, *Nonlinear Programming*. Belmont, MA: Athena Scientific, 1999.

[11] M.-G. Cojocaru and L. Jonker, "Existence of solutions to projected differential equations in hilbert spaces," *Proceedings of the American Mathematical Society*, vol. 132, no. 1, pp. 183–193, 2004.

[12] D. Zhang and A. Nagurney, "On the stability of projected dynamical systems," *J. Optim. Theory Appl.*, vol. 85, pp. 97–124, Apr. 1995.

[13] S. Paternain and A. Ribeiro, "Online learning of feasible strategies in unknown environments,"

[14] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2011.