2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

Efficient Iterative Linear-Quadratic Approximations for Nonlinear Multi-Player General-Sum Differential Games

David Fridovich-Keil, Ellis Ratner, Lasse Peters, Anca D. Dragan, and Claire J. Tomlin

***Abstract*— Many problems in robotics involve multiple decision making agents. To operate efficiently in such settings, a robot must reason about the impact of its decisions on the behavior of other agents. Differential games offer an expressive theoretical framework for formulating these types of multi-agent problems. Unfortunately, most numerical solution techniques scale poorly with state dimension and are rarely used in real-time applications. For this reason, it is common to predict the future decisions of other agents and solve the resulting decoupled, i.e., single-agent, optimal control problem. This decoupling neglects the underlying interactive nature of the problem; however, efficient solution techniques do exist for broad classes of optimal control problems. We take inspiration from one such technique, the iterative linear-quadratic regulator (ILQR), which solves repeated approximations with linear dynamics and quadratic costs. Similarly, our proposed algorithm solves repeated linear-quadratic *games*. We experimentally benchmark our algorithm in several examples with a variety of initial conditions and show that the resulting strategies exhibit complex interactive behavior. Our results indicate that our algorithm converges reliably and runs in real-time. In a three-player, 14-state simulated intersection problem, our algorithm initially converges in < 0.25 s. Receding horizon invocations converge in < 50 ms in a hardware collision-avoidance test.**

I. Introduction

Many problems in robotics require an understanding of how multiple intelligent agents interact. For example, in the intersection depicted in Fig. 1, two cars and a pedestrian wish to reach their respective goals without colliding or leaving their lanes. Successfully navigating the intersection requires either explicit, or perhaps implicit, coordination amongst the agents. Often, these interactions are *decoupled*, with each autonomous agent predicting the behavior of others and then planning an appropriate response. This decoupling necessitates strong predictive assumptions on how agents' decisions impact one another. Differential game theory provides a principled formalism for expressing these types of multi-agent decision making problems without requiring *a priori* predictive assumptions.

Unfortunately, most classes of differential games have no analytic solution, and many numerical techniques suffer from the so-called "curse of dimensionality" [1]. Numerical dynamic programming solutions for general nonlinear systems have been studied extensively, though primarily in cases with *a priori* known objectives and constraints which permit offline computation, such as automated aerial refueling [2]. Approaches such as [3, 4] which separate offline game analysis from online operation are promising. Still, scenarios with more than two players remain extremely challenging, and the practical restriction of solving games offline prevents them from being widely used in many applications of interest, such as autonomous driving.

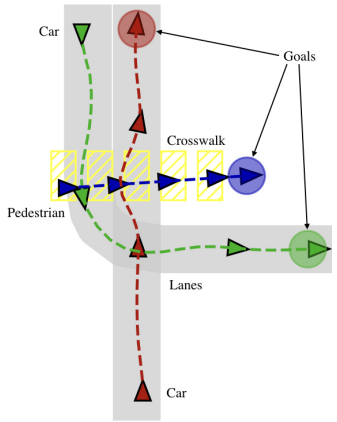

Fig. 1: Demonstration of the proposed algorithm for a three-player general-sum game modeling an intersection. Two cars (red and green triangles) navigate the intersection while a pedestrian (blue triangle) traverses a crosswalk. Observe how both cars swerve slightly to avoid one another and provide extra clearance to the pedestrian.

To simplify matters, decision making problems for multiple agents are often decoupled (see, e.g., [5–7]). For example, the red car in Fig. 1 may wish to simplify its decision problem by *predicting* the future motion of the other agents and *plan reactively*. This simplification reduces the differential game to an optimal control problem, for which there often exist efficient solution techniques. However, the decisions of the other agents *will* depend upon what the red car chooses to do. By ignoring this dependence, the red car is incapable of discovering strategies which exploit the reactions of others, and moreover, trusting in a nominal prediction—e.g., that the pedestrian will get of the way—may lead to unsafe behavior. A differential game formulation of this problem, by contrast, explicitly accounts for the mutual dependence of all agents' decisions.

We propose a novel *local* algorithm for recovering interactive strategies in a broad class of differential games. These strategies qualitatively resemble local Nash equilibria, though there are subtle differences. By solving the underlying game

Department of EECS, UC Berkeley, {dfk, eratner, lasse.peters, anca, tomlin}@eecs.berkeley.edu.
This research is supported by an NSF CAREER award, the Air Force Office of Scientific Research (AFOSR), NSF's CPS FORCES and VeHICaL projects, the UC-Philippine-California Advanced Research Institute, the ONR MURI Embedded Humans, a DARPA Assured Autonomy grant, and the SRC CONIX Center. D. Fridovich-Keil is supported by an NSF Graduate Research Fellowship. E. Ratner is supported by a NASA Space Technology Research Fellowship.

we account for the fundamental interactive nature of the problem, and by seeking a local solution we avoid the curse of dimensionality which arises when searching for global Nash equilibria. Our algorithm builds upon the iterative linear-quadratic regulator (ILQR) [8], a local method used in smooth optimal control problems [9–11]. ILQR repeatedly refines an initial control strategy by efficiently solving approximations with linear dynamics and quadratic costs. Like linear-quadratic (LQ) optimal control problems, LQ games also afford an efficient closed form solution [12]. Our algorithm exploits this analytic solution to solve successive LQ approximations, and thereby finds a local solution to the original game in real-time. For example, our algorithm initially solves the three-player 14-state intersection scenario of Fig. 1 in $< 0.25\,\text{s}$, and receding horizon problems converge in $< 50\,\text{ms}$ in a hardware collision-avoidance test.

II. Background & Related Work

A. General-sum games

Initially formulated in [13, 14], general-sum differential games generalize zero-sum games to model situations in which players have competing—but not necessarily opposite—objectives. Like zero-sum games, general-sum games are characterized by Hamilton-Jacobi equations [13] in which all players' Hamiltonians are coupled with one other. Both zero-sum and general-sum games, and especially games with many players, are generally difficult to solve numerically. However, efficient methods do exist for solving games with linear dynamics and quadratic costs, e.g. [12, 15]. Dockner et al. [16] also characterize classes of games which admit tractable *open loop*, rather than *feedback*, solutions.

B. Approximation techniques

While general-sum games may be analyzed by solving coupled Hamilton-Jacobi equations [13], doing so requires both exponential time and computational memory. A number of more tractable approximate solution techniques have been proposed for zero-sum games, many of which require linear system dynamics, e.g. [17–20], or decomposable dynamics [21]. Approximate dynamic programming techniques such as [22] are not restricted to linear dynamics or zero-sum settings. Still, scalability to online, real-time operation remains a challenge.

Iterative best response algorithms form another class of approximate methods for solving general-sum games. Here, in each iteration every player solves (or approximately solves) the optimal control problem that results from holding other players' strategies fixed. This reduction to a sequence of optimal control problems is attractive; however, it can also be computationally inefficient. Still recent work demonstrates the effectiveness of iterative best response in lane changes [4] and multi-vehicle racing [23].

Another similarly-motivated class of approximations involves changing the information structure of the game. For example, Chen et al. [24] solve a multi-player reach-avoid game by pre-specifying an ordering amongst the players and allowing earlier players to communicate their intended strategies to later players. Zhou et al. [25] and Liu et al. [26] operate in a similar setting, but solve for open-loop conservative strategies.

C. Iterative linear-quadratic (LQ) methods

Iterative LQ approximation methods are increasingly common in the robotics and control communities [9–11, 27]. Our work builds directly upon the iterative linear-quadratic regulator (ILQR) algorithm [8, 28].

At each iteration, ILQR simulates the full nonlinear system trajectory, computes a discrete-time linear dynamics approximation and quadratic cost approximation, and solves a LQR subproblem to generate the next control strategy iterate. While structurally similar to ILQR, our approach solves a LQ game at each iteration instead of a LQR problem. This core idea is related to the sequential linear-quadratic method of [29, 30], which is restricted to the two-player zero-sum context. In this paper, we show that LQ approximations can be applied in N-player, general-sum games. In addition, we experimentally characterize the quality of solutions in several case studies and demonstrate real-time operation.

III. Problem Formulation

We consider a N-player finite horizon general-sum differential game characterized by nonlinear system dynamics

$$\dot{x} = f(t, x, u_{1:N})\,, \tag{1}$$

where $x \in \mathbb{R}^n$ is the state of the system, and $u_i \in \mathbb{R}^{m_i}, i \in [N] \equiv \{1, \ldots, N\}$ is the control input of player i, and $u_{1:N} \equiv (u_1, u_2, \ldots, u_N)$. Each player has a cost function J_i defined as an integral of running costs g_i. J_i is understood to depend implicitly upon the state trajectory $x(\cdot)$, which itself depends upon initial state $x(0)$ and control signals $u_{1:N}(\cdot)$:

$$J_i\big(u_{1:N}(\cdot)\big) \triangleq \int_0^T g_i\big(t, x(t), u_{1:N}(t)\big)dt, \forall i \in [N]\,. \tag{2}$$

We shall presume that f is continuous in t and continuously differentiable in $\{x, u_i\}$ uniformly in t. We shall also require g_i to be twice differentiable in $\{x, u_i\}, \forall t$.

Ideally, we would like to find time-varying state feedback control strategies $\gamma_i^* \in \Gamma_i$ for each player i which constitute a global Nash equilibrium for the game defined by (1) and (2). Here, the strategy space Γ_i for player i is the set of measurable functions $\gamma_i : [0, T] \times \mathbb{R}^n \to \mathbb{R}^{m_i}$ mapping time and state to player i's control input. Note that, in this formulation, player i only observes the state of the system at each time and is unaware of other players' control strategies. With a slight abuse of notation $J_i(\gamma_1; \ldots; \gamma_N) \equiv J_i\big(\gamma_1(\cdot, x(\cdot)), \ldots, \gamma_N(\cdot, x(\cdot))\big)$, the global Nash equilibrium is defined as the set of strategies $\{\gamma_i\}$ for which the following inequalities hold (see, e.g., [12, Chapter 6]):

$$\begin{aligned} J_i^* &\triangleq J_i(\gamma_1^*; \ldots; \gamma_{i-1}^*; \gamma_i^*; \gamma_{i+1}^*; \ldots \gamma_N^*) \\ &\leq J_i(\gamma_1^*; \ldots; \gamma_{i-1}^*; \gamma_i; \gamma_{i+1}^*; \ldots \gamma_N^*), \forall i \in [N]\,. \end{aligned} \tag{3}$$

In (3), the inequalities must hold for all $\gamma_i \in \Gamma_i, \forall i \in [N]$. Informally, a set of feedback strategies $(\gamma_1^*, \ldots, \gamma_N^*)$ is a global Nash equilibrium if no player has a unilateral incentive to deviate from their current strategy.

Since finding a global Nash equilibrium is generally computationally intractable, recent work in adversarial learning [31] and motion planning [23, 32] consider local Nash equilibria instead. Further, [23, 32] simplify the information structure of the game and consider open loop, rather than feedback, strategies. Local Nash equilibria are characterized similarly to (3), except that the inequalities may only hold in a local neighborhood within the strategy space [33, Definition 1]. In this paper, we shall seek a related type of equilibrium, which we describe more precisely in Section IV-B. Intuitively, we seek strategies which satisfy the *global* Nash conditions (3) for the limit of a sequence of *local* approximations to the game. Our experimental results indicate that it does yield highly interactive strategies in a variety of differential games.

IV. Iterative Linear-Quadratic Games

We approach the N-player general-sum game with dynamics (1) and costs (2) from the perspective of classical LQ games. It is well known that Nash equilibrium strategies for finite-horizon LQ games satisfy coupled Riccati differential equations. These coupled Riccati equations may be derived by substituting linear dynamics and quadratic running costs into the generalized HJ equations [14] and analyzing the first order necessary conditions of optimality for each player [12, Chapter 6]. These coupled differential equations may be solved approximately in discrete-time using dynamic programming [12]. We will leverage the existence and computational efficiency of this discrete-time LQ solution to solve successive approximations to the original *non*linear *non*quadratic game.

A. Iterative LQ game algorithm

Our iterative LQ game approach proceeds in stages, as summarized in Algorithm 1. We begin with an initial state $x(0)$ and initial feedback control strategies $\{\gamma_i^0\}$ for each player i, and integrate the system forward (line 3 of Algorithm 1) to obtain the current trajectory iterate $\xi^k \equiv \{\hat{x}(t), \hat{u}_{1:N}(t)\}_{t\in[0,T]}$. Next (line 4) we obtain a Jacobian linearization of the dynamics f about trajectory ξ^k. At each time $t \in [0, T]$ and for arbitrary states $x(t)$ and controls $u_i(t)$ we define deviations from this trajectory $\delta x(t) = x(t) - \hat{x}(t)$ and $\delta u_i(t) = u_i(t) - \hat{u}_i(t)$. Thus equipped, we compute a continuous-time linear system approximation about ξ^k:

$$\dot{\delta x}(t) \approx A(t)\delta x(t) + \sum_{i\in[N]} B_i(t)\delta u_i(t), \tag{4}$$

where $A(t)$ is the Jacobian $D_{\hat{x}} f\big(t, \hat{x}(t), \hat{u}_{1:N}(t)\big)$ and $B_i(t)$ is likewise $D_{\hat{u}_i} f\big(t, \hat{x}(t), \hat{u}_{1:N}(t)\big)$.

We also obtain a quadratic approximation to the running cost g_i for each player i (see line 5 of Algorithm 1)

$$\begin{aligned} &g_i\big(t, x(t), u_{1:N}(t)\big) \approx \\ &\quad g_i\big(t, \hat{x}(t), \hat{u}_{1:N}(t)\big) + \frac{1}{2}\delta x(t)^T \left(Q_i(t)\delta x(t) + 2l_i(t)\right) + \\ &\quad \frac{1}{2}\sum_{j\in[N]} \delta u_j(t)^T \left(R_{ij}(t)\delta u_j(t) + 2r_{ij}(t)\right), \end{aligned} \tag{5}$$

Algorithm 1: Iterative LQ Games

Input: initial state $x(0)$, control strategies $\{\gamma_i^0\}_{i\in[N]}$, time horizon T, running costs $\{g_i\}_{i\in[N]}$
Output: converged control strategies $\{\gamma_i^*\}_{i\in[N]}$

1 **for** *iteration* $k = 1, 2, \ldots$ **do**
2 $\quad \xi^k \equiv \{\hat{x}(t), \hat{u}_{1:N}(t)\}_{t\in[0,t]} \leftarrow$
3 $\quad\quad$ getTrajectory$\big(x(0), \{\gamma_i^{k-1}\}\big)$;
4 $\quad \{A(t), B_i(t)\} \leftarrow$ linearizeDynamics$\big(\xi^k\big)$;
5 $\quad \{l_i(t), Q_i(t), r_{ij}(t), R_{ij}(t)\} \leftarrow$
$\quad\quad$ quadraticizeCost$\big(\xi^k\big)$;
6 $\quad \{\tilde{\gamma}_i^k\} \leftarrow$ solveLQGame$\big($
7 $\quad\quad \{A(t), B_i(t), l_i(t), Q_i(t), r_{ij}(t), R_{ij}(t)\}\big)$;
8 $\quad \{\gamma_i^k\} \leftarrow$ stepToward$\big(\{\gamma_i^{k-1}, \tilde{\gamma}_i^k\}\big)$;
9 $\quad$ **if** *converged* **then**
10 $\quad\quad$ **return** $\{\gamma_i^k\}$

where vector $l_i(t)$ is the gradient $D_{\hat{x}} g_i$, r_{ij} is $D_{\hat{u}_j} g_i$, and matrices Q_i and R_{ij} are Hessians $D^2_{\hat{x}\hat{x}} g_i$ and $D^2_{\hat{u}_j\hat{u}_j} g_i$, respectively. We neglect mixed partials $D^2_{\hat{u}_j\hat{u}_k} g_i$ and $D^2_{\hat{x}\hat{u}_j} g_i$ as they rarely appear in cost structures of practical interest, although they could be incorporated if needed.

Thus, we have constructed a finite-horizon continuous-time LQ game, which may be solved via coupled Riccati differential equations [12, 34]. This results in a new set of *candidate* feedback strategies $\{\tilde{\gamma}_i^k\}$ which constitute a feedback (global) Nash equilibrium of the LQ game [12]. In fact, these feedback strategies are affine maps of the form:

$$\tilde{\gamma}_i^k\big(t, x(t)\big) = \hat{u}_i(t) - P_i^k(t)\delta x(t) - \alpha_i^k(t)\,, \tag{6}$$

with gains $P_i^k(t) \in \mathbb{R}^{m_i\times n}$ and affine terms $\alpha_i^k(t) \in \mathbb{R}^{m_i}$.

However, we find that choosing $\gamma_i^k = \tilde{\gamma}_i^k$ often causes Algorithm 1 to diverge because the trajectory resulting from $\{\tilde{\gamma}_i\}$ is far enough from the current trajectory iterate ξ^k that the dynamics linearizations (Algorithm 1, line 4) and cost quadraticizations (line 5) no longer hold. As in ILQR [35], to improve convergence, we take only a small step in the "direction" of $\tilde{\gamma}_i^k$.[1] More precisely, for some choice of step size $\eta \in (0, 1]$, we set

$$\gamma_i^k\big(t, x(t)\big) = \hat{u}_i(t) - P_i^k(t)\delta x(t) - \eta\alpha_i^k(t)\,, \tag{7}$$

which corresponds to line 8 in Algorithm 1. Note that at $t = 0$, $\delta x(0) = 0$ and $\gamma_i^k\big(0, x(0)\big) = \hat{u}_i(0) - \eta\alpha_i^k(0)$. Thus, taking $\eta = 0$, we have $\gamma_i^k\big(t, x(t)\big) = \hat{u}_i(t)$ (which may be verified recursively). That is, when $\eta = 0$ we recover the open-loop controls from the previous iterate, and hence $x(t) = \hat{x}(t)$. Taking $\eta = 1$, we recover the LQ solution in (6). Similar logic implies the following lemma.

Lemma 1: Suppose that trajectory ξ^* is a fixed point of Algorithm 1, with $\eta \neq 0$. Then, the converged affine terms $\{\alpha_i^*(t)\}$ must all be identically zero for all time.

In ILQR, it is important to perform a line-search over step size η to ensure a sufficient decrease in cost at every

[1] We also note that, in practice, it is often helpful to "regularize" the problem by adding scaled identity matrices ϵI to Q_i and/or R_{ij}.

iteration, and thereby improve convergence (e.g., [35]). In the context of a noncooperative game, however, line-searching to decrease "cost" does not make sense, as costs $\{J_i\}$ may conflict. For this reason, like other local methods in games (e.g., [23]), our approach is not guaranteed to converge from arbitrary initializations. In practice, however, we find that our algorithm typically converges for a fixed, small step size (e.g. $\eta = 0.01$). Heuristically decaying step size with each iteration k or line-searching until $\|\xi^k - \xi^{k-1}\|$ is smaller than a threshold are also promising alternatives. Further investigation of line-search methods in games is a rich topic of future research.

Note: Although we have presented our algorithm in continuous-time, in practice, we solve the coupled Riccati equations analytically in discrete-time via dynamic programming. Please refer to [12, Corollary 6.1] for a full derivation. To discretize time at resolution Δt, we employ Runge-Kutta integration of nonlinear dynamics (1) with a zero-order hold for control input over each time interval Δt.

B. Characterizing fixed points

Suppose Algorithm 1 converges to a fixed point $(\gamma_1^*, \ldots, \gamma_N^*)$. These strategies are the *global* Nash equilibrium of a *local* LQ approximation of the original game about the limiting operating point ξ^*. While it is tempting to presume that such fixed points are also local Nash equilibria of the original game, this is not always true because converged strategies are only optimal for a LQ *approximation* of the game at every time rather than the original game. This approximation neglects higher order coupling effects between each player's running cost g_i and other players' inputs $u_j, j \neq i$. These coupling effects arise in the game setting but *not* in the optimal control setting, where ILQR converges to local minima.

C. Computational complexity and runtime

The per-iteration computational complexity of our approach is comparable to that of ILQR, and scales modestly with the number of players, N. Specifically, at each iteration, we first linearize system dynamics about ξ^k. Presuming that the state dimension n is larger than the control dimension m_i for each player, linearization requires computing $\mathcal{O}(n^2)$ partial derivatives at each time step (which also holds for ILQR). We also quadraticize costs, which requires $\mathcal{O}(Nn^2)$ partial derivatives at each time step (compared to $\mathcal{O}(n^2)$ for ILQR). Finally, solving the coupled Riccati equations of the resulting LQ game at each time step has complexity $\mathcal{O}(N^3n^3)$, which may be verified by inspecting [12, Corollary 6.1] (for ILQR, this complexity is $\mathcal{O}(n^3)$).

Total algorithmic complexity depends upon the number of iterations, which we currently have no theory to bound. However, empirical results are extremely promising. For the three-player 14-state game described in Section V-B, each iteration takes $< 8\,\text{ms}$ and the entire game can be solved from a zero initialization ($P_i^0(\cdot) = 0, \alpha_i^0(\cdot) = 0$) in $< 0.25\,\text{s}$. Moreover, receding horizon invocations in a hardware collision-avoidance test can be solved in $< 50\,\text{ms}$ (Section V-C). All computation times are reported for single-threaded operation on a 2017 MacBook Pro with a 2.8 GHz Intel Core i7 CPU. For reference, the iterative best response scheme of [32] reports solving a receding horizon two-player zero-sum racing game at 2 Hz, and the method of [30] reportedly takes several minutes to converge for a different two-player zero-sum example. The dynamics and costs in both cases differ from those in Section V (or are not clearly reported); nonetheless, the runtime of our approach compares favorably.

V. Examples

In this section, we demonstrate our algorithm experimentally in three-player noncooperative settings, both in software simulation and hardware.[2]

A. Monte Carlo study

We begin by presenting a Monte Carlo study of the convergence properties of Algorithm 1. As we shall see, the solution to which Algorithm 1 converges depends upon the initial strategy of each player, γ_i^0. For clarity, we study this sensitivity in a game with simplified cost structure so that differences in solution are more easily attributable to coupling between players.

Concretely, we consider a three-player "hallway navigation" game with time horizon $10\,\text{s}$ and discretization $0.1\,\text{s}$. Here, three people wish to interchange positions in a narrow hallway while maintaining at least $1\,\text{m}$ clearance between one another. We model each player i's motion as:

$$\begin{aligned} \dot{p}_{x,i} &= v_i \cos(\theta_i)\,, \quad \dot{\theta}_i = \omega_i\,, \\ \dot{p}_{y,i} &= v_i \sin(\theta_i)\,, \quad \dot{v}_i = a_i\,, \end{aligned} \tag{8}$$

where $p_i := (p_{x,i}, p_{y,i})$ denotes player i's position, θ_i heading angle, v_i speed, and input $u_i := (\omega_i, a_i)$ yaw rate and longitudinal acceleration. Concatenating all players' states into a global state vector $x := (p_{x,i}, p_{y,i}, \theta_i, v_i)_{i=1}^3$, the game has 12 state dimensions and six input dimensions.

We encode this problem with running costs g_i (2) expressed as weighted sums of the following:

$$\text{wall: } \mathbf{1}\{|p_{y,i}| > d_{\text{hall}}\}(|p_{y,i}| - d_{\text{hall}})^2 \tag{9}$$

$$\text{proximity: } \mathbf{1}\{\|p_i - p_j\| < d_{\text{prox}}\}(d_{\text{prox}} - \|p_i - p_j\|)^2 \tag{10}$$

$$\text{goal: } \mathbf{1}\{t > T - t_{\text{goal}}\}\|p_i - p_{\text{goal},i}\|^2 \tag{11}$$

$$\text{input: } u_i^T R_{ii} u_i \tag{12}$$

Here, $\mathbf{1}\{\cdot\}$ is the indicator function, i.e., it takes the value 1 if the given condition holds, and 0 otherwise. d_{hall} and d_{prox} denote threshold distances from hallway center and between players, which we set to 0.75 and $1\,\text{m}$, respectively. The goal cost is active only for the last t_{goal} seconds, and the goal position is given by $p_{\text{goal},i}$ for each player i. Control inputs are penalized quadratically, with R_{ii} a diagonal matrix. The hallway is too narrow for all players to cross simultaneously without incurring a large proximity cost; hence, this proximity cost induces strong coupling between players' strategies.

[2]Video summary available at `https://youtu.be/KPEPk-QrkQ8`.

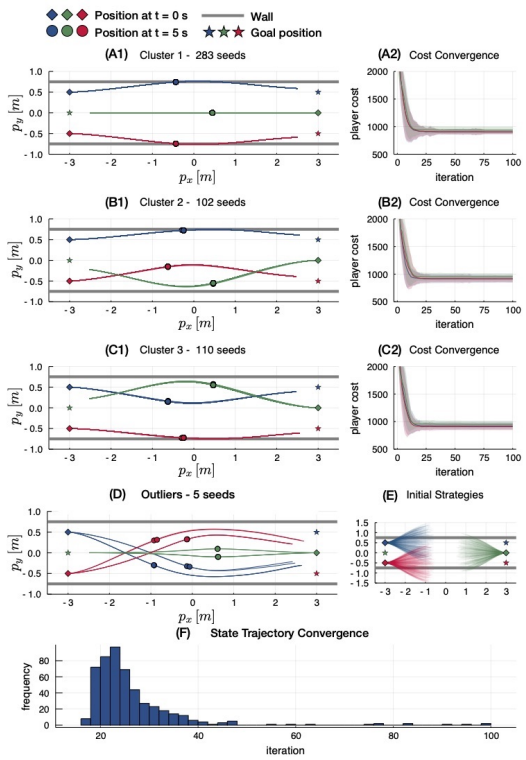

Fig. 2: Monte Carlo results for a three-player hallway navigation game. (A1, B1, C1) Converged trajectories clustered by total Euclidean distance; each cluster corresponds to a qualitatively distinct mode of interaction. (A2, B2, C2) Costs for each player at each solver iteration. The shaded region corresponds to one standard deviation. (D) Several converged trajectories did not match a cluster (A-C). (E) Trajectories resulting from 500 random initial strategies. (F) Histogram of iterations until state trajectory has converged.

Fig. 2 displays the results of our Monte Carlo study. We seed Algorithm 1 with 500 random sinusoidal open-loop initial strategies, which correspond to the trajectories shown in Fig. 2(E). From each of these initializations, we run Algorithm 1 for 100 iterations and cluster the resulting trajectories by Euclidean distance. As shown in Fig. 2(A1, B1, C1), these clusters correspond to plausible modes of interaction; in each case, one or more players incur slightly higher cost to make room for the others to pass. Beside each of these clusters in Fig. 2(A2, B2, C2), we also report the mean and standard deviation of each player's cost at each solver iteration. As shown in Fig. 2(F), state trajectories converge within an ℓ_∞ tolerance of 0.01 in well under 100 iterations.

In these 500 random samples, only 6 did not converge and had to be resampled, and 5 converged to trajectories which were outliers from the clusters depicted in Fig. 2(A-C). These outliers are shown in Fig. 2(D). We observe that, in these 5 cases, the players come within $0.5\,\text{m}$ of one another.

B. Three-player intersection

Next, we consider a more complicated game intended to model traffic at an intersection. As shown in Fig. 3, we consider an intersection with two cars and one pedestrian, all of which must cross paths to reach desired goal locations. We use a time horizon of $5\,\text{s}$ with discretization of $0.1\,\text{s}$, and Algorithm 1 terminates in under $0.25\,\text{s}$.

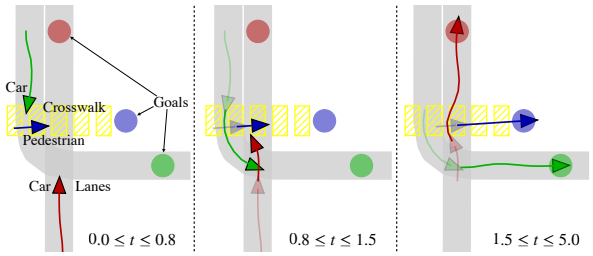

Fig. 3: Three-player intersection game. (Left) Green car seeks the lane center and then swerves slightly to avoid the pedestrian. (Center) Red car accelerates in front of the green car and slows slightly to allow the pedestrian to pass. (Right) Red car swerves left to give pedestrian a wide berth.

We model the pedestrian's dynamics as in (8) and each cars i's dynamics as follows:

$$\begin{aligned} \dot{p}_{x,i} &= v_i \cos(\theta_i)\,, \quad \dot{\theta}_i = v_i \tan(\phi_i)/L_i, \quad \dot{\phi}_i = \psi_i \\ \dot{p}_{y,i} &= v_i \sin(\theta_i)\,, \quad \dot{v}_i = a_i\,, \end{aligned} \tag{13}$$

where the state variables are as before (8) except for front wheel angle ϕ_i. L_i is the inter-axle distance, and input $u_i := (\psi_i, a_i)$ is the front wheel angular rate and longitudinal acceleration, respectively. Together, the state of this game is 14-dimensional.

The running cost for each player i are specified as weighted sums of (10)–(12), and the following:

$$\text{lane center: } d_\ell(p_i)^2 \tag{14}$$

$$\text{lane boundary: } \mathbf{1}\{d_\ell(p_i) > d_{\text{lane}}\}(d_{\text{lane}} - d_\ell(p_i))^2 \tag{15}$$

$$\text{nominal speed: } (v_i - v_{\text{ref},i})^2 \tag{16}$$

$$\text{speed bounds: } \mathbf{1}\{v_i > \overline{v}_i\}(v_i - \overline{v}_i)^2 + \mathbf{1}\{v_i < \underline{v}_i\}(\underline{v}_i - v_i)^2 \tag{17}$$

Here, d_{lane} denotes the lane half-width, and $d_\ell(p_i) := \min_{p_\ell \in \ell} \|p_\ell - p_i\|$ measures player i's distance to lane centerline ℓ. Speed v_i is penalized quadratically away from a fixed reference $v_{\text{ref},i}$ also outside limits $\underline{v}_i$ and $\overline{v}_i$.

Fig. 3 shows a time-lapse of the converged solution identified by Algorithm 1. These strategies exhibit non-trivial coordination among the players as they compete to reach their goals efficiently while sharing responsibility for collision-avoidance. Such competitive behavior would be difficult for any single agent to recover from a decoupled, optimal control formulation. Observe how, between $0 \le t \le 0.8\,\text{s}$ (left), the green car initially seeks the lane center to minimize its cost, but then turns slightly to avoid the pedestrian (blue). Between $0.8 \le t \le 1.5\,\text{s}$ (center), the red car turns right to pass in front of the green car, and then slows and begins to turn left to give the pedestrian time to cross. Finally (right), the red car turns left to give the pedestrian a wide berth.

C. Receding horizon motion planning

Differential games are appropriate in a variety of applications including multi-agent modeling and coordinated planning. Here we present a proof-of-concept for their use in single-agent planning in a dynamic environment. In this setting, a single robot operates amongst multiple other agents whose true objectives are unknown. The robot models these objectives and formulates the interaction as a differential game. Then, crucially, the robot re-solves the differential

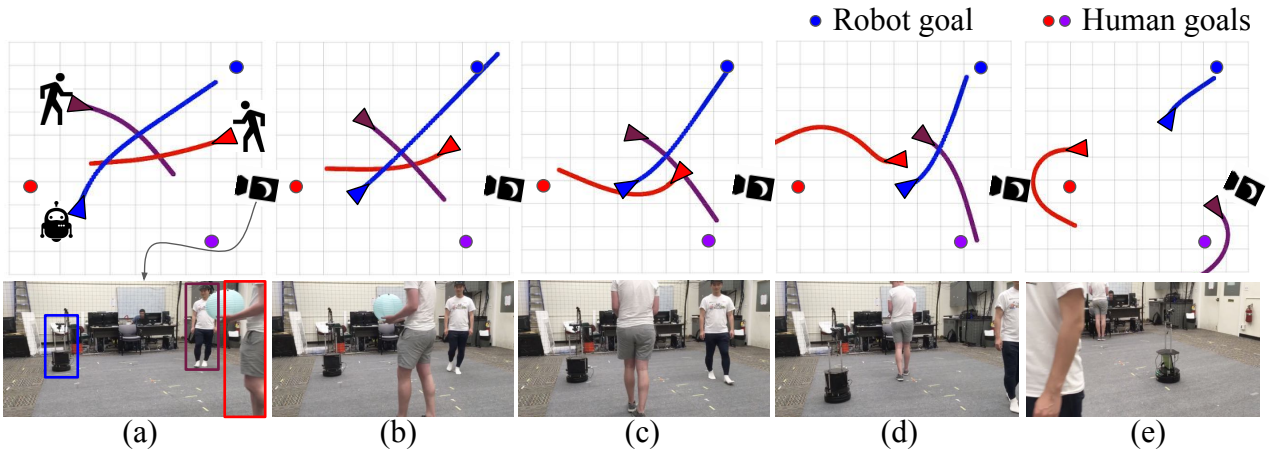

Fig. 4: Time-lapse of a hardware demonstration of Algorithm 1. We model the interaction of a ground robot (blue triangle) and two humans (purple and red triangles) using a differential game in which each agent wishes to reach a goal location while maintaining sufficient distance from other agents. Our algorithm solves receding horizon instantiations of this game in real-time, and successfully plans and executes interactive collision-avoiding maneuvers. Planned (and predicted) trajectories are shown in blue (robot), purple, and red (humans).

game along a receding time horizon to account for possible deviations between the other agents' decisions and those which result from the game solution.

We implement Algorithm 1 in C++[3] within the Robot Operating System (ROS) framework, and evaluate it in a real-time hardware test onboard a TurtleBot 2 ground robot, in a motion capture room with two human participants. The TurtleBot wishes to cross the room while maintaining $> 1\,\text{m}$ clearance to the humans, and it models the humans likewise. We model the TurtleBot dynamics as (8) and humans likewise but with constant speed v_i, i.e.:

$$\dot{p}_{x,i} = v_i \cos(\theta_i)\,, \quad \dot{p}_{y,i} = v_i \sin(\theta_i)\,, \quad \dot{\theta}_i = \omega_i\,. \tag{18}$$

We use a similar cost structure as in Section V-B, and initialize Algorithm 1 with all agents' strategies identically zero (i.e., $P_i^0(\cdot), \alpha_i^0(\cdot) \equiv 0$). We re-solve the game in a $10\,\text{s}$ receding horizon with time discretization of $0.1\,\text{s}$, and warm-start each successive receding horizon invocation with the previous solution. Replanning every $0.25\,\text{s}$, Algorithm 1 reliably converges in under $50\,\text{ms}$. We gather state information for all agents using a motion capture system. Fig. 4 shows a time-lapse of a typical interaction.

Initially, in frame (a) Algorithm 1 identifies a set of strategies which steer each agent to their respective goals while maintaining a large separation. Of course, the human participants do not actually follow these precise trajectories; hence later receding horizon invocations converge to slightly different strategies. In fact, between frames (c) and (d) the red participant makes an unanticipated sharp right-hand turn, which forces the (blue) robot to stay to the right of its previous plan and then turn left in order to maintain sufficient separation between itself and both humans. We note that our assumed cost structure models all agents as wishing to avoid collision. Thus, the resulting strategies may be less conservative than those that would arise from a non-game-theoretic motion planning approach.

[3]Code available at: `github.com/HJReachability/ilqgames`

VI. Discussion

We have presented a novel algorithm for finding local solutions in multi-player general-sum differential games. Our approach is closely related to the iterative linear-quadratic regulator (ILQR) [8], and offers a straightforward way for optimal control practitioners to directly account for multi-agent interactions via differential games. We performed a Monte Carlo study which demonstrated the reliability of our algorithm and its ability to identify complex interactive strategies for multiple agents. These solutions display the competitive behavior associated with local Nash equilibria, although there are subtle differences. We also showcased our method in a three-player 14-dimensional traffic example, and tested it in real-time operation in a hardware robot navigation scenario, following a receding time horizon.

There are several other approaches to identifying local solutions in differential games, such as iterative best response [23]. We have shown the computational efficiency of our approach. However, quantitatively comparing the solutions identified by different algorithms is challenging due to differences in equilibrium concept, information structure (feedback vs. open loop), and implementation details. Furthermore, in arbitrary general-sum games, different players may prefer different equilibria. Studying the qualitative differences in these equilibria is an important direction of future research.

Although our experiments show that our algorithm converges reliably, we have no *a priori* theoretical guarantee of convergence from arbitrary initializations. Future work will seek a theoretical explanation of this empirical property.

Acknowledgments

The authors would like to thank Andrew Packard for his helpful insights on LQ games, as well as Forrest Laine, Chih-Yuan Chiu, Somil Bansal, Jaime Fisac, Tyler Westenbroek, and Eric Mazumdar for helpful discussions.

References

[1] R. Bellman. *Dynamic programming*. Tech. rep. RAND CORP SANTA MONICA CA, 1956.

[2] J. Ding, J. Sprinkle, S. S. Sastry, and C. J. Tomlin. "Reachability calculations for automated aerial refueling". *47th Conference on Decision and Control (CDC)*. IEEE. 2008.

[3] S. L. Herbert*, M. Chen*, S. Han, S. Bansal, J. F. Fisac, and C. J. Tomlin. "FaSTrack: a Modular Framework for Fast and Guaranteed Safe Motion Planning". *56th Conference on Decision and Control (CDC)* (2017).

[4] J. F. Fisac, E. Bronstein, E. Stefansson, D. Sadigh, S. S. Sastry, and A. D. Dragan. "Hierarchical game-theoretic planning for autonomous vehicles". *International Conference on Robotics and Automation (ICRA)*. IEEE. 2019.

[5] B. D. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. A. Bagnell, M. Hebert, A. K. Dey, and S. Srinivasa. "Planning-based prediction for pedestrians". *International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2009.

[6] H. Bai, S. Cai, N. Ye, D. Hsu, and W. S. Lee. "Intention-aware online POMDP planning for autonomous driving in a crowd". *International Conference on Robotics and Automation (ICRA)*. IEEE. 2015.

[7] E. Schmerling, K. Leung, W. Vollprecht, and M. Pavone. "Multimodal probabilistic model-based planning for human-robot interaction". *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018.

[8] W. Li and E. Todorov. "Iterative linear quadratic regulator design for nonlinear biological movement systems." *ICINCO*. 2004.

[9] J. Koenemann, A. Del Prete, Y. Tassa, E. Todorov, O. Stasse, M. Bennewitz, and N. Mansard. "Whole-body model-predictive control applied to the HRP-2 humanoid". *International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2015.

[10] N. Kitaev, I. Mordatch, S. Patil, and P. Abbeel. "Physics-based trajectory optimization for grasping in cluttered environments". *International Conference on Robotics and Automation (ICRA)*. IEEE. 2015.

[11] J. Chen, W. Zhan, and M. Tomizuka. "Constrained iterative LQR for on-road autonomous driving motion planning". *International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2017.

[12] T. Başar and G. J. Olsder. *Dynamic noncooperative game theory*. SIAM, 1999.

[13] A. W. Starr and Y.-C. Ho. "Nonzero-sum differential games". *Journal of Optimization Theory and Applications* 3.3 (1969).

[14] A. Starr and Y.-C. Ho. "Further properties of nonzero-sum differential games". *Journal of Optimization Theory and Applications* 3.4 (1969).

[15] T. Li and Z. Gajic. "Lyapunov iterations for solving coupled algebraic Riccati equations of Nash differential games and algebraic Riccati equations of zero-sum games". *New trends in dynamic games and applications*. Springer, 1995.

[16] E. Dockner, G. Feichtinger, and S. Jørgensen. "Tractable classes of nonzero-sum open-loop Nash differential games: theory and examples". *Journal of Optimization Theory and Applications* 45.2 (1985).

[17] A. B. Kurzhanski and P. Varaiya. "Ellipsoidal techniques for reachability analysis: internal approximation". *Systems & Control Letters* 41.3 (2000).

[18] A. B. Kurzhanski and P. Varaiya. "On ellipsoidal techniques for reachability analysis. part ii: Internal approximations box-valued constraints". *Optimization Methods and Software* 17.2 (2002).

[19] M. R. Greenstreet and I. Mitchell. "Reachability analysis using polygonal projections". *International Workshop on Hybrid Systems: Computation and Control*. Springer. 1999.

[20] J. N. Maidens, S. Kaynama, I. M. Mitchell, M. M. Oishi, and G. A. Dumont. "Lagrangian methods for approximating the viability kernel in high-dimensional systems". *Automatica* 49.7 (2013).

[21] M. Chen, S. L. Herbert, M. S. Vashishtha, S. Bansal, and C. J. Tomlin. "Decomposition of reachable sets and tubes for a class of nonlinear systems". *Transactions on Automatic Control* 63.11 (2018).

[22] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-dynamic programming*. Athena Scientific Belmont, MA, 1996.

[23] Z. Wang, R. Spica, and M. Schwager. "Game Theoretic Motion Planning for Multi-robot Racing". *Distributed Autonomous Robotic Systems*. Springer, 2019.

[24] M. Chen, J. F. Fisac, S. Sastry, and C. J. Tomlin. "Safe sequential path planning of multi-vehicle systems via double-obstacle Hamilton-Jacobi-Isaacs variational inequality". *2015 European Control Conference (ECC)*. IEEE. 2015.

[25] Z. Zhou, R. Takei, H. Huang, and C. J. Tomlin. "A general, open-loop formulation for reach-avoid games". *51st Conference on Decision and Control (CDC)*. IEEE. 2012.

[26] S.-Y. Liu, Z. Zhou, C. Tomlin, and J. K. Hedrick. "Evasion of a team of dubins vehicles from a hidden pursuer". *International Conference on Robotics and Automation (ICRA)*. IEEE. 2014.

[27] J. van den Berg. "Iterated LQR smoothing for locally-optimal feedback control of systems with non-linear dynamics and non-quadratic cost". *American Control Conference (ACC)*. IEEE. 2014.

[28] E. Todorov and W. Li. "A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems". *American Control Conference (ACC)*. IEEE. 2005.

[29] H. Mukai, A. Tanikawa, I. Tunay, I. Katz, H. Schättler, P. Rinaldi, I. Ozcan, G. Wang, L. Yang, and Y. Sawada. "Sequential linear quadratic method for differential games". *Proc. 2nd DARPA-JFACC Symposium on Advances in Enterprise Control*. Citeseer. 2000.

[30] A. Tanikawa, H. Mukai, and M. Xu. "Local Convergence of the Sequential Quadratic Method for Differential Games". *Transactions of the Institute of Systems, Control and Information Engineers* 25.12 (2012).

[31] E. V. Mazumdar, M. I. Jordan, and S. S. Sastry. "On finding local nash equilibria (and only local nash equilibria) in zero-sum games". *arXiv preprint arXiv:1901.00838* (2019).

[32] M. Wang, Z. Wang, J. Talbot, J. C. Gerdes, and M. Schwager. "Game Theoretic Planning for Self-Driving Cars in Competitive Scenarios". *Robotics: Science & Systems*. 2019.

[33] L. J. Ratliff, S. A. Burden, and S. S. Sastry. "On the characterization of local Nash equilibria in continuous games". *Transactions on Automatic Control* 61.8 (2016).

[34] M. Green and D. J. Limebeer. *Linear robust control*. Courier Corporation, 2012.

[35] Y. Tassa, N. Mansard, and E. Todorov. "Control-limited differential dynamic programming". *International Conference on Robotics and Automation (ICRA)*. IEEE. 2014.