2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

A Sequential Quadratic Programming Approach to the Solution of Open-Loop Generalized Nash Equilibria

Edward L. Zhu and Francesco Borrelli

Abstract— **In this work, we propose a numerical method for the solution of local generalized Nash equilibria (GNE) for the class of open-loop general-sum dynamic games for agents with nonlinear dynamics and constraints. In particular, we formulate a sequential quadratic programming (SQP) approach which requires only the solution of a single convex quadratic program at each iteration and is locally convergent. Central to the effectiveness of our approach is a non-monotonic line search method and a novel merit function for SQP step acceptance which helps to improve solver convergence beyond the local neighborhood of a GNE. We demonstrate the effectiveness of the algorithm in the context of car racing, where we see up to 32% improvement of success rate when comparing against a recent solution approach for dynamic games. We also make our code available at `https://github.com/zhu-edward/DGSQP`.**

I. Introduction

Real-world robotic systems must be able to operate safely in an environment which is inhabited by other intelligent agents who are each pursuing their own agenda. In the process of doing so, it is likely that interactions between the agents arise due to limitations on the shared workspace. How the robot handles such interactions is critical to its performance and safety as these scenarios typically involve inter-agent constraints becoming active. The primary challenge here is that of information availability, where agents, either due to technological limitations or to maintain a competitive advantage, do not share information with each other about their intentions or future plans. It is therefore crucial to model the behavior of the other agents in the environment when planning the actions of the robot.

Whereas traditional approaches adopt a pipeline architecture where an upstream module provides forecasts of the behavior of other agents [1], [2], dynamic non-cooperative games [3] provide a theoretical framework for simultaneous prediction and planning for multiple ego-centric agents and has seen many applications in trajectory optimization for robotic systems [4], [5], [6], [7]. This is done by formulating a set of coupled optimization problems which describe the behavior of an agent as a function of the others' in the environment. Two solution or equilibrium concepts are common for dynamic games, namely the Stackelberg and Nash equilibria, which make different assumptions on the structure of the game. A Stackelberg equilibrium can be found for a game with an explicit leader-follower hierarchy [8], whereas a Nash equilibrium models the case when agents make their decisions simultaneously. Our work focuses on the selection of generalized Nash equilibria (GNE) [9], which we believe to be a good fit for modeling the behavior of ego-centric agents with state and input constraints when no *a priori* structure is imposed on the order of the interactions. Specifically, we propose an iterative approach for finding GNE of a discrete-time dynamic game based on sequential quadratic programming (SQP), which builds upon ideas from [10]. In particular, we are able to handle general nonlinear game dynamics and constraints on both the game state and agent actions. Our primary contributions are the following:

1) A locally convergent solver for GNE of dynamic games with nonlinear dynamics and constraints.
2) A novel merit function, which when used in conjunction with a non-monotone line search strategy, greatly improves solver convergence in practice.
3) A simulation study in the context of car racing which shows up to 32% improvement in success rate when comparing our approach with a recent method.

Related Work: [5], [11], [12] take a differential dynamic programming approach to obtain a linear-quadratic approximation of the dynamic game. However, the approach is unable to explicitly account for inequality constraints and instead include them in the cost function via barrier functions. [13], [14] formulate a decomposition based method called Iterative Best Response where the agents improve their strategy in a sequential manner while holding the behavior of all other agents fixed. It is shown that fixed points of this algorithm correspond to GNE. However, the method requires the solution of the same number of optimization problems as agents and can be slow to converge in practice. In contrast, our approach only requires the solution of a single optimization problem at each iteration. The approach proposed in this work is inspired by and builds upon [4] and [10] with improvements which are demonstrated to be essential to effectively solve the class of problems we are interested in. [10] proposes a similar SQP approach for the computation of *feedback* Nash equilibria, but does not investigate its local behavior. Compared to [10], we also introduce a new merit function and line search strategy which improves solver convergence. [4] proposes a GNE solver based on an augmented Lagrangian approach. The solver, called ALGAMES, outperforms [5]. However, as will be shown in our comparison, ALGAMES appears to struggle with convergence in the context of car racing where more complex dynamics and environments are introduced.

This work was supported by the National Science Foundation under Grant No. 1931853.

Edward L. Zhu, and Francesco Borrelli {edward.zhu, fborrelli}@berkeley.edu are with the Department of Mechanical Engineering at the University of California, Berkeley CA, USA

II. Problem Formulation

Consider an M-agent, finite-horizon, discrete-time, general-sum, open-loop, dynamic game whose state is characterized by the joint dynamical system:

$$x_{k+1} = f(x_k, u_k), \tag{1}$$

where $x_k^i \in \mathcal{X}^i$ and $u_k^i \in \mathcal{U}^i$ are the state and input of agent i at time step k and $x_k = [{x_k^1}^\top, \dots, {x_k^M}^\top]^\top \in \mathbb{R}^n$, $u_k = [{u_k^1}^\top, \dots, {u_k^M}^\top]^\top \in \mathbb{R}^m$ are the concatenated states and inputs of all agents. In this work, we will use the notation $x_k^{\neg i}$ and $u_k^{\neg i}$ to denote the vector of states and inputs of all but the i-th agent.

Each agent i attempts to minimize its own cost function, which is comprised of stage costs l_k^i and terminal cost l_N^i, over a horizon of length N:

$$\bar{J}^i(\mathbf{x}, \mathbf{u}^i) = \sum_{k=0}^{N-1} l_k^i(x_k, u_k^i) + l_N^i(x_N) \tag{2a}$$

$$= J^i(\mathbf{u}^1, \dots, \mathbf{u}^M, x_0), \tag{2b}$$

where $\mathbf{x} = \{x_0, \dots, x_N\}$ and $\mathbf{u}^i = \{u_0^i, \dots, u_{N-1}^i\}$ denote state and input sequences over the horizon. Note that the cost in (2a) for agent i depends on its *own* inputs and the *joint* state. We arrive at (2b) by recursively substituting in the dynamics (1) to the cost function, which are naturally a function of the open-loop input sequences for all agents. The agents are additionally subject to n_c constraints

$$C(\mathbf{u}^1, \dots, \mathbf{u}^M, x_0) \leq 0, \tag{3}$$

which can be used to describe individual constraints as well as coupling between agents and where we have once again made the dependence on the joint dynamics implicit. For the sake of brevity, when focusing on agent i, we omit the inital state x_0 and write the cost and constraint functions as $J^i(\mathbf{u}^i, \mathbf{u}^{\neg i})$ and $C(\mathbf{u}^i, \mathbf{u}^{\neg i})$. Let us now define the conditional constraint set

$$\mathcal{U}^i(\mathbf{u}^{\neg i}) = \{\mathbf{u}^i \mid C(\mathbf{u}^i, \mathbf{u}^{\neg i}) \leq 0\},$$

which can be interpreted as a restriction of the joint constraint set for agent i given some $\mathbf{u}^{\neg i}$. We assume that the sets $\mathcal{X}^i$ and $\mathcal{U}^i$ are compact and the functions f, J^i, and C are twice continuously differentiable on $\mathcal{X}$ and $\mathcal{U}$.

A. Generalized Nash Equilibrium

We define the constrained dynamic game as the tuple:

$$\Gamma = (N, \mathcal{X}, \mathcal{U}, f, \{J^i\}_{i=1}^M, C). \tag{4}$$

For such a game, a GNE is attained at the set of feasible input sequences $\mathbf{u} = \{\mathbf{u}^i\}_{i=1}^M$ which minimize (2) for all agents i. Formally, we define this solution concept as follows:

Definition 1. A generalized Nash equilibrium (GNE) [9] for the dynamic game Γ is the set of open-loop solutions $\mathbf{u}^\star = \{\mathbf{u}^{i,\star}\}_{i=1}^M$ such that for each agent i:

$$J^i(\mathbf{u}^{i,\star}, \mathbf{u}^{\neg i,\star}) \leq J^i(\mathbf{u}^i, \mathbf{u}^{\neg i,\star}),\ \forall \mathbf{u}^i \in \mathcal{U}^i(\mathbf{u}^{\neg i,\star}).$$

If the condition holds only in some local neighborhood of $\mathbf{u}^{i,\star}$, then $\mathbf{u}^\star$ is denoted as a local GNE.

In other words, at a local GNE, agents cannot improve their cost by unilaterally perturbing their open-loop solution in a locally feasible direction. The local GNE for agent i can be obtained equivalently by solving the following constrained finite horizon optimal control problems (FHOCP):

$$\mathbf{u}^{i,\star}(\mathbf{u}^{\neg i,\star}) = \arg\min_{\mathbf{u}^i} \ J^i(\mathbf{u}^i, \mathbf{u}^{\neg i,\star}) \quad \text{subject to} \ \ C(\mathbf{u}^i, \mathbf{u}^{\neg i,\star}) \leq 0. \tag{5}$$

where $\mathbf{u}^{\neg i,\star}$ correspond to local GNE solutions for the other agents. Note that we are assuming uniqueness of the local GNE of (4). This will be made formal in the next section. A distinct advantage of using (5) to model agent interactions is that a dynamic game allows for a direct representation of agents with competing objectives as the M objectives are considered separately instead of being summed together, which is typical in cooperative multi-agent approaches [15].

III. An SQP Approach to Dynamic Games

In this section, we propose a method which iteratively solves for open-loop local GNE of dynamic games using sequential quadratic approximations. In particular, we will derive the algorithm and present guarantees on local convergence, which is based on established SQP theory. We begin by defining the Lagrangian functions for the M coupled FHOCPs in (5):

$$\mathcal{L}^i(\mathbf{u}^i, \mathbf{u}^{\neg i,\star}, \lambda^i) = J^i(\mathbf{u}^i, \mathbf{u}^{\neg i,\star}) + C(\mathbf{u}^i, \mathbf{u}^{\neg i,\star})^\top \lambda^i,$$

where we have again omitted the dependence on the initial state x_0 for brevity. As in [4], we require that the Lagrange multipliers $\lambda^i \geq 0$ are equal over all agents, i.e $\lambda^i = \lambda^j = \lambda$, $\forall i, j \in \{1, \dots, M\}$. Under this condition, the GNE from (5) are also known as normalized Nash equilibria [16].

A direct consequence of writing the constrained dynamic game in the coupled nonlinear optimization form of (5) is that, subject to regularity conditions, solutions of (5) must satisfy the KKT conditions below:

$$\nabla_{\mathbf{u}^i} \mathcal{L}^i(\mathbf{u}^{i,\star}, \mathbf{u}^{\neg i,\star}, \lambda^\star) = 0,\ \forall i = 1, \dots, M, \tag{6a}$$

$$C(\mathbf{u}^{1,\star}, \dots, \mathbf{u}^{M,\star}) \leq 0, \tag{6b}$$

$$C(\mathbf{u}^{1,\star}, \dots, \mathbf{u}^{M,\star})^\top \lambda^\star = 0, \tag{6c}$$

$$\lambda^\star \geq 0. \tag{6d}$$

We therefore propose to find a local GNE as a solution to the KKT system (6) in an iterative fashion starting from an initial guess for the primal and dual solution, which we denote as $\mathbf{u}_0^i$ and $\lambda_0 \geq 0$ respectively, and taking steps p_q^i and p_q^λ, at iteration q, to obtain the sequence of iterates:

$$\mathbf{u}_{q+1}^i = \mathbf{u}_q^i + p_q^i,\ \lambda_{q+1} = \lambda_q + p_q^\lambda. \tag{7}$$

In particular, we form a quadratic approximation of (6a) and linearize the constraints in (6b) about the primal and dual solution at iteration q in a SQP manner [17] as follows:

$$L_q = \begin{bmatrix} \nabla^2_{\mathbf{u}^1} \mathcal{L}_q^1 & \nabla_{\mathbf{u}^2,\mathbf{u}^1} \mathcal{L}_q^1 & \dots & \nabla_{\mathbf{u}^M,\mathbf{u}^1} \mathcal{L}_q^1 \\ \nabla_{\mathbf{u}^1,\mathbf{u}^2} \mathcal{L}_q^2 & \nabla^2_{\mathbf{u}^2} \mathcal{L}_q^2 & \dots & \nabla_{\mathbf{u}^M,\mathbf{u}^2} \mathcal{L}_q^2 \\ \vdots & \vdots & \ddots & \vdots \\ \nabla_{\mathbf{u}^1,\mathbf{u}^M} \mathcal{L}_q^M & \nabla_{\mathbf{u}^2,\mathbf{u}^M} \mathcal{L}_q^M & \dots & \nabla^2_{\mathbf{u}^M} \mathcal{L}_q^M \end{bmatrix},$$

$$h_q = \begin{bmatrix} \nabla_{\mathbf{u}^1} J_q^1 & \nabla_{\mathbf{u}^2} J_q^2 & \dots & \nabla_{\mathbf{u}^M} J_q^M \end{bmatrix}^\top,$$

$$G_q = \begin{bmatrix} \nabla_{\mathbf{u}^1} C_q & \nabla_{\mathbf{u}^2} C_q & \dots & \nabla_{\mathbf{u}^M} C_q \end{bmatrix},$$

$$B_q = \text{proj}_{\succeq 0}((L_q + L_q^\top)/2) + \epsilon I, \tag{8}$$

where the subscript q indicates that the corresponding quantity is evaluated at the primal and dual iterate $\mathbf{u}_q$ and λ_q. Here, $\epsilon \geq 0$ is a regularization coefficient, I is the identity matrix of appropriate size, and $\text{proj}_{\succeq 0} = \sum_{i=1}^{n} \max\{0, s_i\} v_i v_i^\top$ denotes the operation which projects the symmetric matrix $X \in \mathbb{R}^{n \times n}$ onto the positive semi-definite cone, where s_i and v_i denote the i-th eigenvalue and eigenvector of X respectively. Using this approximation, we solve for the step in the primal variables via the following convex quadratic program (QP):

$$p_q^{\mathbf{u}} = \arg \min_{p^1, \ldots, p^M} \quad \frac{1}{2} p^{\mathbf{u}\top} B_q p^{\mathbf{u}} + h_q^\top p^{\mathbf{u}} \tag{9a}$$

$$\text{subject to} \quad C_q + G_q p^{\mathbf{u}} \leq 0. \tag{9b}$$

where we denote $p^{\mathbf{u}} = [{p^1}^\top, \ldots, {p^M}^\top]^\top$. Denote the Lagrange multipliers corresponding to the solution of (9) as d_q. We then define the step in the dual variables as

$$p_q^\lambda = d_q - \lambda_q. \tag{10}$$

We note that in contrast to the approach used in [13] and [14], which require the solution of M optimization problems, our SQP procedure requires the solution of only a single QP at each iteration.

A. Local Behavior of Dynamic Game SQP

We make the following assumptions about the primal and dual solutions of (5):

Assumption 1. Solutions $\{\mathbf{u}^{i,\star}\}_{i=1}^M$ and $\lambda^\star$ of (5) satisfy the following, for each $i \in \{1, \ldots, M\}$:

- The rows of the Jacobian of the active constraints at the local GNE, i.e. $\nabla_{\mathbf{u}^i} \bar{C}(\mathbf{u}^{i,\star}, \mathbf{u}^{\neg i,\star})$, are linearly independent (LICQ).
- $d^\top \nabla^2_{\mathbf{u}^i} \mathcal{L}^i(\mathbf{u}^{i,\star}, \mathbf{u}^{\neg i,\star}, \lambda^\star) d > 0$, $\forall d \neq 0$ such that $\nabla_{\mathbf{u}^i} \bar{C}(\mathbf{u}^{i,\star}, \mathbf{u}^{\neg i,\star})^\top d = 0$.

It is straightforward to see that (6) and Assumption 1 together constitute necessary and sufficient conditions for a primal and dual solution of (5) for agent i to be locally optimal and unique. When these conditions hold for the solutions over all agents, satisfaction of the requirements for a unique local GNE follow immediately. This result was proven formally in [10]. Note that, as in [18] and [10], Assumption 1 is standard and can be verified a posteriori.

To analyze the local behavior of the iterative procedure as defined by (7), (8), and (9), let us assume that $\mathbf{u}_0$ and λ_0 are close to the optimal solution and the subset of active constraints at the local GNE, which we denote as $\bar{C}$, with Jacobian $\bar{G}$, is known and constant at each iteration q. Therefore, for the purposes of this section, we can replace the inequality constraint in (9b) with the equality constraint:

$$\bar{C}_q + \bar{G}_q p^{\mathbf{u}} = 0. \tag{11}$$

We refer to the QP constructed from (9a) and (11) as EQP.

It was shown in [18], [17] that under the aforementioned assumptions, the traditional SQP step is identical to a Newton step for the corresponding KKT system. The SQP step therefore inherits the quadratic convergence rate of Newton's method in a local neighborhood of the optimal solution [18, Theorem 3.1]. However, in the case of dynamic games, the equivalence between the SQP procedure and Newton's method is no longer exact since the matrix L_q is not symmetric in general. To see this, let us first state the joint KKT system for the equality constrained version of (5):

$$F(\mathbf{u}^\star, \lambda^\star) = \begin{bmatrix} \nabla_{\mathbf{u}} \mathcal{L}(\mathbf{u}^\star, \lambda^\star) \\ \bar{C}(\mathbf{u}^\star) \end{bmatrix} = 0, \tag{12}$$

Where $\nabla_{\mathbf{u}} \mathcal{L}(\mathbf{u}^\star, \lambda^\star)$ denotes the concatenation of (6a) for all agents. For the system of equations (12), the Newton step at iteration q is the solution of the linear system:

$$\begin{bmatrix} L_q & \bar{G}_q^\top \\ \bar{G}_q & 0 \end{bmatrix} \begin{bmatrix} \bar{p}_q^{\mathbf{u}} \\ \bar{p}_q^\lambda \end{bmatrix} = - \begin{bmatrix} h_q + \bar{G}_q^\top \lambda_q \\ \bar{C}_q \end{bmatrix}. \tag{13}$$

On the other hand, by the first order optimality conditions for EQP, we have that the SQP step must satisfy

$$\begin{bmatrix} B_q & \bar{G}_q^\top \\ \bar{G}_q & 0 \end{bmatrix} \begin{bmatrix} p_q^{\mathbf{u}} \\ p_q^\lambda \end{bmatrix} = - \begin{bmatrix} h_q + \bar{G}_q^\top \lambda_q \\ \bar{C}_q \end{bmatrix}. \tag{14}$$

When the matrix L_q is positive definite and $\epsilon = 0$, (14) and (13) are equivalent. This corresponds to the special case of potential games [19]. However, this is not true in general for our SQP step, which implies that we cannot inherit the quadratic convergence of Newton's method. Instead, the SQP step from (9a) and (11) can be seen as a symmetric approximation to the Newton step. As such, we can establish local linear convergence for our SQP procedure via established theory for SQP with approximate Hessians. The proof for Theorem 1 can be found in [20].

Theorem 1. *Consider the dynamic game defined by* (4)*. Let Assumption 1 hold. Then there exist positive constants* ϵ_1 *and* ϵ_2 *such that if*

$$\|\mathbf{u}_0 - \mathbf{u}^\star\| \leq \epsilon_1, \ \|B_0 - L^\star\| \leq \epsilon_2,$$

and $\lambda_0 = -(\bar{G}_0 \bar{G}_0^\top)^{-1} \bar{G}_0 h_0$*, then the sequence* $(\mathbf{u}_q, \lambda_q)$ *generated by the SQP procedure* (7) *and* (9) *converges linearly to* $(\mathbf{u}^\star, \lambda^\star)$*.*

IV. A Novel Merit Function and Non-Monotone Line Search Strategy for Dynamic Game SQP

We have shown that our proposed SQP approach exhibits linear convergence when close to a local GNE. However, as is commonly seen with numerical methods for nonlinear optimization, a naïve implementation of the procedure defined by (7), (8), and (9) often performs poorly due to overly aggressive steps leading to diverging iterates. In this section, we introduce a merit function and line search method which will help address this problem in practice. These components will be used to determine how much of the SQP step $p_q^{\mathbf{u}}$ and p_q^λ can be taken to make progress towards a local GNE while remaining in a region about the current iterate where the QP approximation (9) is valid.

A. A Novel Merit Function

In traditional constrained optimization, merit functions typically track a combination of cost value and constraint violation. In the context of dynamic games, this is not as straightforward as the agents may have conflicting objectives and a proposed step may result in an increase in the objectives of some agents along with a decrease in others'. We also cannot just simply sum the objectives as minimizers

of the combined cost function may not be local GNE. We therefore propose the following merit function:

$$\phi(\mathbf{u},\lambda,s;\mu) = \frac{1}{2}\|\nabla\mathcal{L}(\mathbf{u},\lambda)\|_2^2 + \mu\|C(\mathbf{u}) - s\|_1, \quad (15)$$

where $\nabla\mathcal{L}(\mathbf{u},\lambda)$ is the concatenation of (6a) for all agents and define $\gamma(\mathbf{u},\lambda) = (1/2)\|\nabla\mathcal{L}(\mathbf{u},\lambda)\|_2^2$. The slack variable $s = \min(0, C(\mathbf{u}))$ is defined element-wise such that $C - s$ captures violation of the inequality constraints and we define the step $p^s = C(\mathbf{u}) + G(\mathbf{u})p^{\mathbf{u}} - s$. Compared to the merit function from [10], which only included the first term of (15), ours includes the l^1 norm term, which is a common choice in nonlinear optimization [17] and whose purpose will be described shortly. Clearly, the merit function attains a minimum of zero at any local GNE. However, we note that this merit function is not *exact* [17] since the first order conditions are only necessary for optimality.

Since we would like the sequence of iterates to converge to the minimizers of ϕ, it follows that at each iteration we would like the step to achieve a decrease in ϕ. In other words, we would like the directional derivative of ϕ evaluated along the step to be negative. Following a simple derivation in [20], the directional derivative of ϕ can be written as:

$$D(\phi(\mathbf{u},\lambda,s;\mu), p^{\mathbf{u}}, p^{\lambda}) = (\nabla_{\mathbf{u},\lambda}\gamma)\mathbf{p} - \mu\|C - s\|_1, \quad (16)$$

where $\mathbf{p} = [{p^{\mathbf{u}}}^\top, {p^{\lambda}}^\top]^\top$. From (16) it follows that that given $C - s \neq 0$, we can select a value for $\mu > 0$ such that the directional derivative is negative for the step $p^{\mathbf{u}}$ and p^{λ}. As such, we propose the following expression to compute μ, given some $\rho \in (0,1)$:

$$\mu \geq (\nabla_{\mathbf{u},\lambda}\gamma)\mathbf{p}/((1-\rho)\|C - s\|_1), \quad (17)$$

which results in $D(\phi(\mathbf{u},\lambda;\mu), p^{\mathbf{u}}, p^{\lambda}) \leq -\rho\mu\|C - s\|_1$. In the case when $C - s = 0$, the directional derivative is not guaranteed to be negative as its sign is now fully dependent on the residual between the KKT matrix L and its symmetric approximation B. This fact can be verified by deriving an expression for the first term of (16). For dynamic games where the agents have highly coupled and differing objectives, i.e. when L is highly non-symmetric, it is likely that the approximation would suffer and that we may not be able to achieve a decrease in the merit function. For this reason, we utilize a non-monotone strategy for the line search step, which will be discussed in the following.

B. A Non-Monotone Line Search Strategy

Line search methods are used in conjunction with merit functions to achieve a compromise between the goals of making rapid progress towards the optimal solution and keeping the iterates from diverging. This is done by finding the largest step size $\alpha \in (0,1]$ such that the following standard decrease condition is satisfied [18]:

$$\begin{aligned}&\phi(\mathbf{u}_q + \alpha p_q^{\mathbf{u}}, \lambda_q + \alpha p_q^{\lambda}, s_q + \alpha p_q^{s}; \mu) \quad (18)\\ &\quad \leq \phi(\mathbf{u}_q, \lambda_q, s_q; \mu) + \zeta\alpha D(\phi(\mathbf{u}_q, \lambda_q, s_q; \mu), p_q^{\mathbf{u}}, p_q^{\lambda}),\end{aligned}$$

where $\zeta \in (0, 0.5)$. However, since our merit function is not exact, the line search procedure can be susceptible to poor local minima which do not correspond to local GNE. We therefore include in our approach a non-monotone approach to line search called the *watchdog* strategy [21]. Instead of insisting on a sufficient decrease in the merit function at every iteration, this approach allows for relaxed steps to be taken for a certain number of iterations, which can lead to increases in the merit function. The decrease requirement (18) is then enforced after the prescribed number of relaxed iterations. The rationale behind this strategy is that we can use the relaxed steps as a way to escape regions where it is difficult to make progress w.r.t. the merit function. The algorithm is presented in detail in [20].

Algorithm 1: Dynamic Game SQP (DG-SQP)

Input: $\mathbf{u}_0$, ρ, ζ

1 $q \leftarrow 0$, $\lambda_0 \leftarrow \max(0, -(G_0 G_0^\top)^{-1} G_0 h_0)$;
2 **while** *not converged* **do**
3 $\quad B_q,\ h_q,\ G_q,\ C_q \leftarrow$ (8);
4 $\quad p_q^{\mathbf{u}},\ p_q^{\lambda} \leftarrow$ (9), (10);
5 $\quad s_q \leftarrow \min(0, C_q)$, $p_q^s \leftarrow C_q + G_q p_q^{\mathbf{u}} - s_q$;
6 $\quad$ **if** $C_q - s_q \neq 0$ **then**
7 $\quad\quad$ Compute μ from (17);
8 $\quad$ **else**
9 $\quad\quad \mu \leftarrow 0$;
10 $\quad$ **end**
11 $\quad \mathbf{u}_{q+1},\ \lambda_{q+1} \leftarrow$ watchdog line search;
12 $\quad q \leftarrow q + 1$
13 **end**
14 **return** $\mathbf{u}^\star \leftarrow \mathbf{u}_q$, $\lambda^\star \leftarrow \lambda_q$;

C. The Dynamic Game SQP Algorithm

By combining the elements previously discussed, we arrive at the dynamic game SQP (DG-SQP) algorithm presented in Algorithm 1. The algorithm requires as input initial guesses of open-loop input sequences for each agent. Line 1 initializes the primal and dual iterates, where the dual variables are initialized as the least squares solution to (6a). Lines 2 to 13 perform the SQP iteration which has been described in Sections III and IV. An iterate is said to have converged to a local GNE if it satisfies the KKT conditions described in (6) up to some user specified tolerance. Namely, for some given $\epsilon_1,\ \epsilon_2,\ \epsilon_3 > 0$, we require the conditions $\|\nabla\mathcal{L}(\mathbf{u}_q, \lambda_q)\|_\infty \leq \epsilon_1,\ \|C(\mathbf{u}_q)\|_\infty \leq \epsilon_2,\ |\lambda_q^\top C(\mathbf{u}_q)| \leq \epsilon_3$ be satisfied in order for the algorithm to terminate successfully. We allow the algorithm to terminate with relative tolerance if the difference between successive primal and dual iterates are below a given threshold for a given number of iterations. Note that when this occurs, the solution may not be a local GNE. The algorithm outputs the open-loop strategies for the M agents and the corresponding dual multipliers.

V. Simulation Study

In this section, we use simulation studies to demonstrate the performance of our DG-SQP (DS) algorithm and to compare our approach with the recently proposed GNE solver ALGAMES (AL) [4]. The DG-SQP algorithm was implemented in Python All results were obtained on a desktop with a 2.5 GHz 11th-Gen Intel Core i7 CPU.

We demonstrate our approach in the context of head-to-head car racing, where our approach would be used by a vehicle to simultaneously obtain an open-loop control

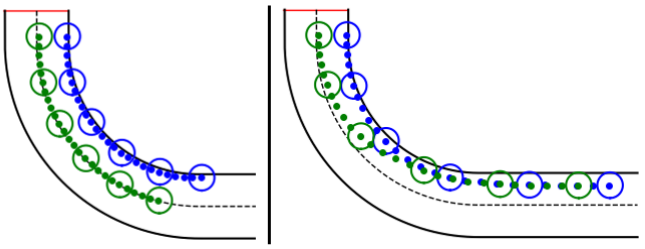

Fig. 1: Example of the initial guess (left) and a corresponding local GNE (right) of horizon length $N = 25$ for two agents on a curved track segment with a 90° turn. The circles represent the collision avoidance constraints.

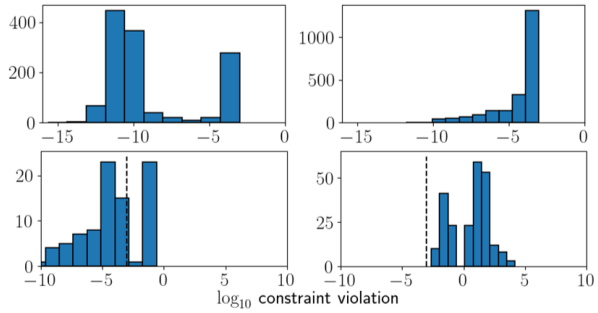

Fig. 2: Largest non-zero constraint violation at solver convergence (top) and failure (bottom) of DG-SQP (left) and ALGAMES (right). The dashed line corresponds to constraint violation of 10^{-3}.

sequence and predictions of its opponent's behavior. We assume that no information about future plans is shared between agents and that they must avoid collisions with each other while also remaining within the boundaries of the track. In our examples, the agents are described by the kinematic bicycle model [22] with the following state and input vectors: $x = [p_x, p_y, v_x, e_\psi, s, e_y]^\top \in \mathbb{R}^6$, $u = [a, \delta]^\top \in \mathbb{R}^2$, where $p = (p_x, p_y)$ are the Cartesian position and v_x is the longitudinal velocity of the vehicle's center of gravity (CoG). The remaining states are expressed in a Frenet reference frame [23] which is defined w.r.t. the centerline of the track. s and e_y denote the distance travelled along the track and lateral deviation from the centerline, of the CoG, respectively. e_ψ is the deviation between the vehicle's heading and the tangent angle of the track centerline at s [24]. We obtain the discrete-time dynamics using Euler discretization with a time step of $T_s = 0.1$ s. The inputs to the vehicle are the longitudinal acceleration a and front wheel steering angle δ. We define the collision avoidance and track boundary constraints using the expressions $(r^i + r^j)^2 - \|p^i - p^j\|_2^2 \leq 0$ and $-W/2 \leq e_y^i \leq W/2$ where r^i and r^j are the radii of the circular collision buffers for agents i and j respectively and W is the width of the track. We subject the input magnitude and rate to identical box constraints for all agents.

A. Comparison with ALGAMES

We compare DS against a custom Python implementation of AL. In this comparison study, all agents have identical dynamics and use the following cost functions:

$$l_k^i(x_k, u_k^i) = \frac{1}{2} {u_k^i}^\top R^i u_k^i + \frac{1}{2} \Delta {u_k^i}^\top R_d^i \Delta u_k^i \tag{19a}$$

$$l_N^i(x_N) = -c_p s_N^i + c_c \textstyle\sum_{j \neq i} \arctan(s_N^i - s_N^j), \tag{19b}$$

where $\Delta u_k^i = u_k^i - u_{k-1}^i$. The stage cost (19a) penalizes the input magnitude and rate with $R^i, R_d^i \succ 0$. The terminal cost (19b) captures the competitive nature of racing with

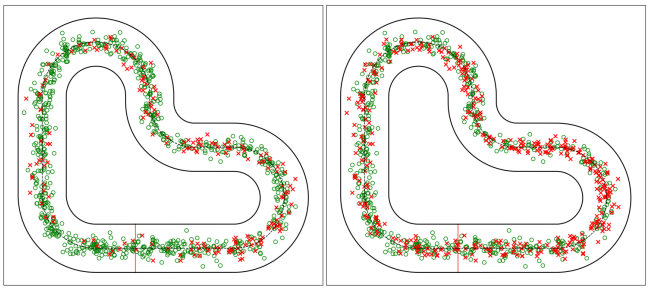

Fig. 3: Monte Carlo results on a racetrack for DG-SQP (left) and AL-GAMES (right). Each point denotes the average sampled initial position for the two agents. ○, × denote successful and failed trials respectively.

$c_p, c_c > 0$, where the first term encourages progress along the track and the second term is made small when agent i is ahead of all other agents.

We conduct the first comparison on a segment of curved track to examine the effect of track curvature and game horizon length on the performance of the GNE solvers. In particular, we perform a Monte Carlo analysis by randomly sampling feasible initial states near the start of the track segment and roll out the initial guess via a PID controller which maintains the car's speed and lateral deviation from the centerline, as seen in the left plot of Fig. 1. This guess is then used to initialize both algorithms. We do 200 trials each on tracks with 45, 75, and 90 degree turns for horizon lengths of 10, 15, 20, and 25. The results are presented in Tab. I. A trial is said to be successful if the iterates converge to a point where the KKT conditions (6) are satisfied with tolerance 10^{-3}. In the top plots of Fig. 2, it can be seen that when successful, both algorithms return solutions which satisfy the constraints.

Looking at the number of successful trials in the first row of Tab. I, it is clear that DS and AL perform similarly well for track segments with low curvature. These scenarios are similar to those investigated in [4], where dynamic games are played out on straight sections of road, and our results appear to further corroborate the high success rate of AL in these situations. However, as the curvature of the track or horizon length increases, we see that, while both solvers experience failures, DS outperforms AL in terms of success rate, showing a 21% improvement in the case of $\theta = 90°$ and $N = 25$. In row four of Tab. I we report the average number of solves for the two approaches. For DS and AL, this corresponds to the number of QPs and linear systems that are solved, respectively. The average run time of successful trials of DS (in seconds) is shown in the fifth row. From this, we see that for horizons of moderate length, our Python implementation can achieve near real-time performance. We believe that an efficient implementation of DS in a compiled language would achieve a similar level of real-time performance as reported for AL in [4].

We turn next to the failure cases for the two solvers, where we have split them into the two categories *# Fail* in row two and *# Max Iters. Reached* in row three. *# Fail* counts the trials of AL, where the iterations diverged, which is defined as $\|\nabla \mathcal{L}(\mathbf{u}_q, \lambda_q)\|_\infty > 10^5$. For DG-SQP it additionally counts trials where the QP in (9) returned infeasible. *# Max Iters. Reached* counts the trials where a solver reached

TABLE I: Monte Carlo results for $M = 2$ agents on a curved track with θ degree turns and game horizon length N. The first and second row for the # Conv, # Fail, # Max Iters. Reached, and # Solves statistics correspond to DG-SQP and ALGAMES respectively. The Time statistic is reported only for DG-SQP. For the Time statistic, the mean and (standard deviation) in seconds over all successful trials are reported.

θ	45				75				90			
N	10	15	20	25	10	15	20	25	10	15	20	25
# Conv	**200** **200**	**199** **199**	**199** 189	**185** 172	**200** **200**	**199** 192	**195** 180	**169** 136	**200** 198	**197** 193	**193** 167	**172** 142
# Fail	0 0	0 0	0 9	0 16	0 0	0 6	0 16	0 39	0 1	0 4	0 24	0 43
# Max Iters. Reached	0 0	1 1	1 2	15 12	0 0	1 2	5 4	31 25	0 1	3 3	7 9	28 15
# Solves	3 9	6 16	14 28	17 36	3 12	9 23	21 39	25 64	4 12	9 28	25 57	17 77
Time [s]	0.02(0.01)	0.06(0.02)	0.24(0.14)	0.49(0.46)	0.03(0.01)	0.09(0.08)	0.44(0.26)	0.89(0.51)	0.03(0.02)	0.10(0.07)	0.41(0.19)	0.47(0.30)

50 iterations without converging. From these results, it is clear that the majority of the AL failure cases (68.1%) were due to the solver diverging, whereas DS fails primarily due to reaching the maximum allowable iterations. We observe that in many of the DS failure cases, the iterates exhibit oscillatory behavior and fail the stationarity requirement of (6a). However, it is important to note that even in failure, most of the DS iterates remain feasible at termination (up to a tolerance of 10^{-3}), whereas for AL, they do not. This is seen in the bottom plots of Fig. 2, which shows the distribution of constraint violation for the iterates returned by failed trials. We believe that this is due to the explicit linearized inequality constraints in DS. In AL, the iterates may not satisfy any form of the inequality constraints (linearized or otherwise), especially when the estimate of the Lagrange multipliers is poor. In practice, this is important as it is likely that in the case of failure, the outputs of DS, though not local GNE, can still be used as feasible solutions to the dynamic game.

We conduct a second comparison on the racetrack shown in Fig. 3 to investigate the performance of the solvers in a more diverse set of racing conditions. For this study, we fix the game horizon to $N = 15$ and randomly sample the initial states of two agents such that an interaction is likely to occur over the horizon. Specifically, the agents start within 1.2 car lengths of each other and are traveling in the CCW direction at velocities that exhibit at most a 25% difference. Out of the 1000 trials, we observe that DS was successful in 814 trials whereas AL was successful in 618 trials. This is a 31.7% improvement in success rate. Of the trials where the solvers failed, DS (AL) saw 13 (203) instances of divergence or QP failure and 185 (179) instances of reaching max iterations. Fig. 3 illustrates the distribution of solver outcomes w.r.t. the averaged initial positions of the two agents. We note that failure cases for DS occur more frequently in sections of the racetrack with high curvature. This mirrors the trend observed in the previous study and highlights a limitation of our approach which will be investigated in future work.

B. Effect of Merit Function and Non-Monotone Strategy

We next investigate the effect of the merit function and non-monotone line search strategy proposed in Section IV. We are especially interested in the case where the agent objectives are different as this induces asymmetry in the matrix L_q, which adversely affects the quality of our symmetric approximation B_q in (9). For this study, we consider the two agent scenario where the car in front (Agent 1) would like to block the one behind (Agent 2) and impede their progress. As such, we add the term $(1/2)c_b(e^1_{y,k} - e^2_{y,k})^2$, with $c_b > 0$, to the stage and terminal cost functions in (19) for Agent 1 and keep the cost function for Agent 2 the same. This additional term encourages Agent 1 to match the lateral position of Agent 2 on the track. We compare the terminal value of $\|\nabla\mathcal{L}(\mathbf{u}_q, \lambda_q)\|_\infty$ for our approach with a variant of DS which uses a traditional backstepping line search and the merit function from [10]. The results are shown in Fig. 4 for 200 trials on a track segment with $\theta = 90°$ and $N = 25$, where it is clear that, for this track segment, our proposed modifications greatly improve the quality of the solution at termination of the DG-SQP algorithm. We observe a median stationarity of 9.369×10^{-4}, compared to 19.76 when none of the proposed modifications are used.

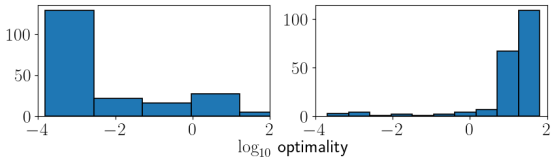

Fig. 4: Stationarity, i.e. $\|\nabla\mathcal{L}(\mathbf{u}_q, \lambda_q)\|_\infty$, of iterate at termination of DG-SQP using the non-monotone line search with the novel merit function from Section IV (left) and standard backstepping line search with the merit function from [10] (right).

VI. Future Work

In this work, we have presented an SQP based approach to the solution of GNE for open-loop dynamic games, which has provable local convergence and good practical performance stemming from novel modifications to the standard SQP algorithm from [10]. We have shown that, in the context of head-to-head car racing, our approach outperforms the recent ALGAMES solver in terms of success rate. However, we observe that our approach can still suffer from convergence issues when dealing with problems with long horizons, highly nonlinear dynamics, or highly asymmetric cost. In future work, we hope to address these shortcomings by further investigating merit functions and line search strategies which can be tailored to dynamic games. Finally, we note that the GNE finding problem for open-loop dynamic games can be cast as a nonlinear mixed complementarity problem (MCP) for which there exists the notable PATH solver [25], which exhibits structural similarities to our approach. The key difference being that we form local approximations of the GNE finding problem via QPs instead of the linear MCPs in [25]. A detailed comparison between our approach and the PATH solver is forthcoming in future work.

REFERENCES

[1] J. Betz, H. Zheng, A. Liniger, U. Rosolia, P. Karle, M. Behl, V. Krovi, and R. Mangharam, "Autonomous vehicles on the edge: A survey on autonomous vehicle racing," *IEEE Open Journal of Intelligent Transportation Systems*, 2022.

[2] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke *et al.*, "Junior: The stanford entry in the urban challenge," *Journal of field Robotics*, vol. 25, no. 9, pp. 569–597, 2008.

[3] T. Başar and G. J. Olsder, *Dynamic noncooperative game theory*. SIAM, 1998.

[4] L. Cleac'h, M. Schwager, Z. Manchester *et al.*, "Algames: A fast solver for constrained dynamic games," in *Proceedings of Robotics: Science and Systems*, 2020.

[5] D. Fridovich-Keil, E. Ratner, L. Peters, A. D. Dragan, and C. J. Tomlin, "Efficient iterative linear-quadratic approximations for nonlinear multi-player general-sum differential games," in *2020 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2020, pp. 1475–1481.

[6] W. Schwarting, A. Pierson, J. Alonso-Mora, S. Karaman, and D. Rus, "Social behavior for autonomous vehicles," *Proceedings of the National Academy of Sciences*, vol. 116, no. 50, pp. 24 972–24 978, 2019.

[7] F. Laine, D. Fridovich-Keil, C.-Y. Chiu, and C. Tomlin, "Multi-hypothesis interactions in game-theoretic motion planning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 8016–8023.

[8] A. Liniger and J. Lygeros, "A noncooperative game approach to autonomous racing," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 3, pp. 884–897, 2019.

[9] F. Facchinei and C. Kanzow, "Generalized nash equilibrium problems," *Annals of Operations Research*, vol. 175, no. 1, pp. 177–211, 2010.

[10] F. Laine, D. Fridovich-Keil, C.-Y. Chiu, and C. Tomlin, "The computation of approximate generalized feedback nash equilibria," *arXiv preprint arXiv:2101.02900*, 2021.

[11] W. Schwarting, A. Pierson, S. Karaman, and D. Rus, "Stochastic dynamic games in belief space," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 2157–2172, 2021.

[12] T. Kavuncu, A. Yaraneri, and N. Mehr, "Potential ilqr: A potential-minimizing controller for planning multi-agent interactive trajectories," *arXiv preprint arXiv:2107.04926*, 2021.

[13] R. Spica, E. Cristofalo, Z. Wang, E. Montijano, and M. Schwager, "A real-time game theoretic planner for autonomous two-player drone racing," *IEEE Transactions on Robotics*, vol. 36, no. 5, pp. 1389–1403, 2020.

[14] M. Wang, Z. Wang, J. Talbot, J. C. Gerdes, and M. Schwager, "Game-theoretic planning for self-driving cars in multivehicle competitive scenarios," *IEEE Transactions on Robotics*, vol. 37, no. 4, pp. 1313–1325, 2021.

[15] E. L. Zhu, Y. R. Stürz, U. Rosolia, and F. Borrelli, "Trajectory optimization for nonlinear multi-agent systems using decentralized learning model predictive control," in *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE, 2020, pp. 6198–6203.

[16] J. B. Rosen, "Existence and uniqueness of equilibrium points for concave n-person games," *Econometrica: Journal of the Econometric Society*, pp. 520–534, 1965.

[17] S. Wright, J. Nocedal *et al.*, "Numerical optimization," *Springer Science*, vol. 35, no. 67-68, p. 7, 1999.

[18] P. T. Boggs and J. W. Tolle, "Sequential quadratic programming," *Acta numerica*, vol. 4, pp. 1–51, 1995.

[19] Q. Zhu, "A lagrangian approach to constrained potential games: Theory and examples," in *2008 47th IEEE Conference on Decision and Control*. IEEE, 2008, pp. 2420–2425.

[20] E. L. Zhu and F. Borrelli, "A sequential quadratic programming approach to the solution of open-loop generalized nash equilibria," *arXiv preprint arXiv:2203.16478*, 2022.

[21] A. R. Conn, N. I. Gould, and P. L. Toint, *Trust region methods*. SIAM, 2000.

[22] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, "Kinematic and dynamic vehicle models for autonomous driving control design," in *2015 IEEE intelligent vehicles symposium (IV)*. IEEE, 2015, pp. 1094–1099.

[23] A. Micaelli and C. Samson, "Trajectory tracking for unicycle-type and two-steering-wheels mobile robots," Ph.D. dissertation, INRIA, 1993.

[24] U. Rosolia and F. Borrelli, "Learning how to autonomously race a car: a predictive control approach," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 6, pp. 2713–2719, 2019.

[25] S. P. Dirkse and M. C. Ferris, "The path solver: a nommonotone stabilization scheme for mixed complementarity problems," *Optimization methods and software*, vol. 5, no. 2, pp. 123–156, 1995.