Control of a Three Degree of Freedom Quadcopter Stand Using Linear Quadratic Integral Based on the Differential Game Theory

Hadi Nobahari
Department of Aerospace Engineering
Sharif University of Technology
Tehran, Iran
nobahari@sharif.edu

Ali BaniAsad
Department of Aerospace Engineering
Sharif University of Technology
Tehran, Iran
alibaniasad1999@yahoo.com

***Abstract*—In this paper, a quadcopter stand with three degrees of freedom was controlled using game theory-based control. The first player tracks a desired input, and the second player creates a disturbance in the tracking of the first player to cause an error in the tracking. The control effort is chosen using the Nash equilibrium, which presupposes that the other player made the worst move. In addition to being resistant to input interruptions, this method may also be resilient to modeling system uncertainty. This method evaluated the performance through simulation in the Simulink environment and implementation on a three-degree-of-freedom stand.**

***Index Terms*—Quadcopter, Differential Game, Game Theory, Nash Equilibrium, Three Degree of Freedom Stand, Model Base Design, Linear Quadratic Regulator**

I. Introduction

Quadcopter is a type of helicopter with four rotors. Quadcopters have extensive applications due to their excellent maneuverability and the possibility of hover flight with high balance. In recent years, companies, universities, and research centers have attracted more to this type of UAV. In this way, the facilities and the flight of these UAVs are continuously improving. Quadcopters are widely used in research, military, imaging, recreation, and agriculture. Mathematical models are used in game theory to examine how rational, intelligent beings cooperate or compete. Game theory can be applied to pursuit and evasion as one of its broad applications. There can be two [1] or more players [2] involved in the pursuit-evasion. Pursuit-evasion can occur indoors as well [3]. In some cases, machine learning and differential games pursuit-evade [4]. Players may play different roles in differential games, such as protecting some targets [5]. The differential game's ability to examine the actions of two or more players makes it powerful. Player cooperation can be used through swarm platooning [6]. Multi-agent [7] and self-driving automobiles [8] motion planning are two other applications of player cooperation.

Due to the widespread use of quadrotors, their control has become an important issue. In order to control quadrotors, neural networks [9] and machine learning [10] methods have been used.

In partnership with BaniAsad Industries

II. Mathematical Modeling

In this section, first, a nonlinear dynamic model of the quadrotor is presented for control purposes. Then, the system linearized for three SISO systems. In dynamic modeling of the quadrotor, the vehicle is assumed to be rigid. The space state

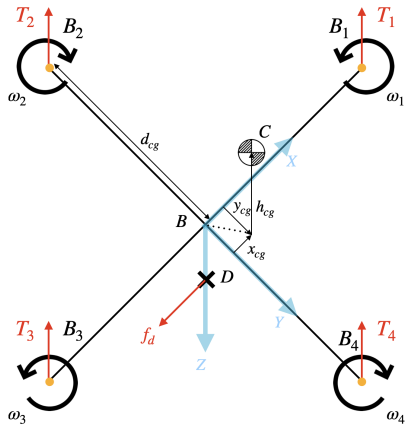


Fig. 1. Configuration of the quadrotor and the conventions

of a three-degree-of-freedom quadcopter is defined as:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} \phi \\ \theta \\ \psi \\ p \\ q \\ r \end{bmatrix} \tag{1}$$

For simplicity, the system's inputs have been changed from rotational speed to influential forces in roll, pitch, and yaw modes. This change makes the problem from multi-input and

multi-output to three single-input problems. Then, the input vector is defined as

$$\mathbf{u} = \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix}^{\mathrm{T}} \tag{2}$$

where

$$u_1 = \omega_2^2 - \omega_4^2,\ u_2 = \omega_1^2 - \omega_3^2,\ u_3 = \omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2 \tag{3}$$

The dynamic model of the three-degree-of-freedom quadcopter stand can be described as

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \tag{4}$$

where elements of $\mathbf{f}$ are nonlinear functions of the state vector $\mathbf{x}$ and the input vector $\mathbf{u}$.

$$\mathbf{f} = \begin{bmatrix} x_4 + x_5 \sin(x_1)\tan(x_2) + x_6 \cos(x_1)\tan(x_2) \\ x_5 \cos(x_1) - x_6 \sin(x_1) \\ (x_5 \sin(x_1) + x_6 \cos(x_1))\sec(x_2) \\ A_1 \cos(x_2)\sin(x_1) + A_2 x_5 x_6 + A_3 u_1 \\ B_1 \sin(x_2) + B_2 x_4 x_6 + B_3 u_2 \\ C_1 x_4 x_5 + C_2 u_3 \end{bmatrix} \tag{5}$$

A. Linearization

Following are the steps of linearization of the system.

$$\delta\dot{\mathbf{x}} = \mathbf{A}\delta\mathbf{x} + \mathbf{B}\delta\mathbf{u} \tag{6}$$

Where $(*)$ term represents incremental variations around this point.

$$\begin{aligned} \mathbf{x}^* &= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}} \\ \mathbf{u}^* &= \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}} \end{aligned} \tag{7}$$

Three single-input systems for each mode are presented here. The Space state matrices for the roll channel are shown below.

$$\begin{aligned} \mathbf{A}_{\text{roll}} &= \begin{bmatrix} \dfrac{\partial f_1}{\partial x_1} & \dfrac{\partial f_1}{\partial x_4} \\ \dfrac{\partial f_4}{\partial x_1} & \dfrac{\partial f_4}{\partial x_4} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ A_1 & 0 \end{bmatrix} \\ \mathbf{B}_{\text{roll}} &= \begin{bmatrix} \dfrac{\partial f_1}{\partial u_1} \\ \dfrac{\partial f_4}{\partial u_1} \end{bmatrix} = \begin{bmatrix} 0 \\ A_3 \end{bmatrix} \end{aligned} \tag{8}$$

The Space state matrices for the pitch channel are shown below.

$$\begin{aligned} \mathbf{A}_{\text{pitch}} &= \begin{bmatrix} \dfrac{\partial f_2}{\partial x_2} & \dfrac{\partial f_2}{\partial x_5} \\ \dfrac{\partial f_5}{\partial x_2} & \dfrac{\partial f_5}{\partial x_5} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ B_1 & 0 \end{bmatrix} \\ \mathbf{B}_{\text{pitch}} &= \begin{bmatrix} \dfrac{\partial f_2}{\partial u_2} \\ \dfrac{\partial f_5}{\partial u_2} \end{bmatrix} = \begin{bmatrix} 0 \\ B_3 \end{bmatrix} \end{aligned} \tag{9}$$

The Space state matrices for the yaw channel are shown below.

$$\begin{aligned} \mathbf{A}_{\text{yaw}} &= \begin{bmatrix} \dfrac{\partial f_3}{\partial x_3} & \dfrac{\partial f_3}{\partial x_6} \\ \dfrac{\partial f_6}{\partial x_3} & \dfrac{\partial f_6}{\partial x_6} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \\ \mathbf{B}_{\text{yaw}} &= \begin{bmatrix} \dfrac{\partial f_3}{\partial u_3} \\ \dfrac{\partial f_6}{\partial u_3} \end{bmatrix} = \begin{bmatrix} 0 \\ C_2 \end{bmatrix} \end{aligned} \tag{10}$$

B. Parameter Estimation

This section modifies the quadrotor stand parameters using the Simulink environment's simulation of different quadrotor channels and the stand's output data. The quadrotor stand parameters have been modified using the Parameter Estimator toolbox available in the Simulink environment. In order to perform the test, the quadrotor stand was released from various initial conditions and inputs, and data was collected using the output from the sensor. Then, Parameter Estimator takes the model and the recorded data of the sensor (stand states). Here is a comparison between the states of the quadrotor in simulation and reality after modifying various parameters.

TABLE I
PARAMETER ESTIMATION RESULTS

Parameter	***Initial Value***	***Value After Estimation***
A_1	7.312	4.152
A_3	1.1×10^{-4}	5.47×10^{-5}
B_1	4.53	4.36
B_3	1.1×10^{-4}	7.13×10^{-5}
C_2	5.45×10^{-5}	1.3×10^{-5}

III. DIFFERENTIAL GAME

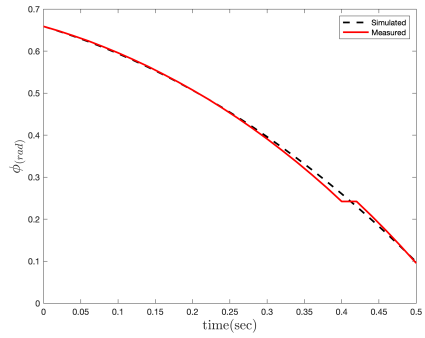
Differential games are a series of problems that arise while examining and simulating dynamic systems in game theory. Differential equations simulate how a state variable or set of state variables changes over time.

A. An introduction to the differential game

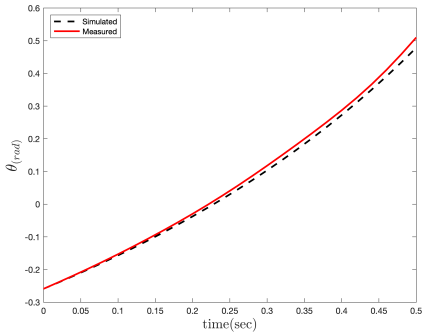
It is considered that two players are involved in this research. The space states of a continuous linear system are shown below.

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B_1}\mathbf{u_1}(t) + \mathbf{B_2}\mathbf{u_2}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{D_1}\mathbf{u_1}(t) + \mathbf{D_2}\mathbf{u_2}(t) \end{aligned} \tag{11}$$

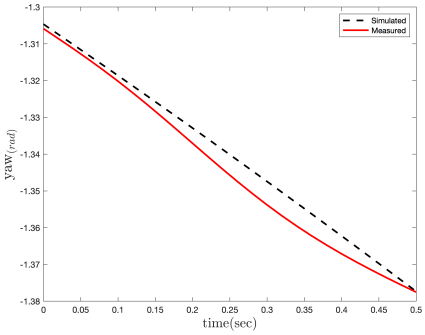
Where $\mathbf{x}$ is the vector of the state variables, $\dot{\mathbf{x}}$ is the time derivative of the state vector, $\mathbf{u_1}$ is the first player (controller) input vector, $\mathbf{u_2}$ is the second player (disturbance) input vector, $\mathbf{y}$ is the output vector, $\mathbf{A}$ is the state matrix, $\mathbf{B_1}$ is the first player input matrix, $\mathbf{B_2}$ is the second player input matrix, $\mathbf{C}$ is the output matrix, $\mathbf{D_1}$ is first player the output matrix and $\mathbf{D_2}$ is second player the output matrix. Equation (11) demonstrates how both participants have an impact on the



(a) Roll Channel



(b) Pitch Channel



(c) Yaw Channel

Fig. 2. Comparison of quadrotor states in simulation and reality.

system's dynamics. The second player may progress toward the goal as a result of the first player's exertion, or vice versa. This paper considers the case that players do not cooperate in order to realize their goals. The situation where players do not work together (non-cooperative) to achieve their objectives is examined in the paper. In this case, every player knows at time $t \in [0, T]$ just the initial state $\mathbf{x_0}$ and the model structure. For the game (1,2), we will use the set of Nash equilibria. Formal Nash equilibrium is defined as follows. An admissible set of actions $(\boldsymbol{u_1}^*, \boldsymbol{u_2}^*)$ is a Nash equilibrium for the game (1,2); if for all admissible $(\boldsymbol{u_1}, \boldsymbol{u_2})$, the following inequalities hold:

$$J_1(\mathbf{u_1}^*, \mathbf{u_2}^*) \leq J_1(\mathbf{u_1}, \mathbf{u_2}^*) \ , \ J_2(\mathbf{u_1}^*, \mathbf{u_2}^*) \leq J_2(\mathbf{u_1}^*, \mathbf{u_2}) \tag{12}$$

B. LQDG controller

For the system described in equation (11), LQDG optimum control effort calculates from equation (13).

$$\mathbf{u_i}(t) = -\mathbf{R_{ii}}^{-1}\mathbf{B_i}^{\mathrm{T}}\mathbf{P_i}(t)\mathbf{x}(t) = -\mathbf{K_i}(t)\mathbf{x}(t), \quad i = 1, 2 \tag{13}$$

In equation (13), $\mathbf{K_i}$ is the optimal feedback gain. Assuming that the other players will make their worst move, this gain is calculated to minimize the quadratic cost function equation (14) of player number i.

$$J_i(\mathbf{u_1}, \mathbf{u_2}) = \int_0^T \Big(\mathbf{x}^{\mathrm{T}}(t)\mathbf{Q_i}\mathbf{x}(t) + \mathbf{u_i}^{\mathrm{T}}(t)\mathbf{R_{ii}}\mathbf{u_i}(t) + \mathbf{u_j}^{\mathrm{T}}(t)\mathbf{R_{ij}}\mathbf{u_j}(t) \Big) \mathrm{d}t \tag{14}$$

Here the matrices $\mathbf{Q_i}$ and $\mathbf{R_{ii}}$ are assumed to be symmetric and $\mathbf{R_{ii}}$ positive definite. $\mathbf{P_i}$ is found by solving the continuous time couple Riccati differential equation:

$$\begin{aligned} \dot{\mathbf{P}}_\mathbf{1}(t) &= -\mathbf{A}^{\mathrm{T}}\mathbf{P_1}(t) - \mathbf{P_1}(t)\mathbf{A} - \mathbf{Q_1} + \mathbf{P_1}(t)\mathbf{S_1}(t)\mathbf{P_1}(t) + \\ &\quad \mathbf{P_1}(t)\mathbf{S_2}(t)\mathbf{P_2}(t) \\ \dot{\mathbf{P}}_\mathbf{2}(t) &= -\mathbf{A}^{\mathrm{T}}\mathbf{P_2}(t) - \mathbf{P_2}(t)\mathbf{A} - \mathbf{Q_2} + \mathbf{P_2}(t)\mathbf{S_2}(t)\mathbf{P_2}(t) + \\ &\quad \mathbf{P_2}(t)\mathbf{S_1}(t)\mathbf{P_1}(t) \end{aligned} \tag{15}$$

Using the shorthand notation $\boldsymbol{S_i} := \boldsymbol{B_i}\boldsymbol{R_{ii}}^{-1}\boldsymbol{B_i}^{\mathrm{T}}$.

C. LQIDG controller

The absence of an integrator in the LQDG controller may result in steady-state errors due to disturbances or modeling errors. The LQIDG controller is based on the LQDG controller to eliminate this error.

The LQIDG controller adds the integral of the difference between the system output and the desired value to the state vector. Therefore, The augmented space states of a continuous linear system are shown below.

$$\mathbf{x_a} = \begin{bmatrix} \mathbf{x_d} - \mathbf{x} \\ \int (\mathbf{y_d} - \mathbf{y}) \end{bmatrix} \tag{16}$$

Where $\mathbf{x_a}$ is the vector of augmented state variables, $\mathbf{x_d}$ is the vector of the desired state variables, and $\mathbf{y_d}$ is the desired output vector. As a result, the state vector and the output vector are equal.

$$\mathbf{y} = \mathbf{x} \tag{17}$$

The following represents the system dynamics in the augmented state space.

$$\dot{\mathbf{x}}_{\boldsymbol{a}}(t) = \mathbf{A_a}\mathbf{x_a}(t) + \mathbf{B_{a_1}}\mathbf{u_{a_1}}(t) + \mathbf{B_{a_2}}\mathbf{u_{a_2}}(t) \tag{18}$$

Where matrices $\mathbf{A_a}$ and $\mathbf{B_a}$ are defined as follows:

$$\mathbf{A_a} = \begin{bmatrix} \mathbf{A} & 0 \\ \mathbf{C} & 0 \end{bmatrix}, \quad \mathbf{B_a} = \begin{bmatrix} \mathbf{B} \\ 0 \end{bmatrix} \tag{19}$$

By introducing a new space state for the system, the remaining design phases of the LQIDG controller are comparable to

those of the LQDG controller. LQIDG optimum control effort calculates from equation (20).

$$\mathbf{u_i}(t) = -{\mathbf{R_{ii}}}^{-1}{\mathbf{B_{a_i}}}^{\mathrm{T}}\mathbf{P_{a_i}}(t)\mathbf{x_a}(t) = -\mathbf{K_{a_i}}(t)\mathbf{x_a}(t),\ i = 1, 2 \tag{20}$$

In equation (20), $\mathbf{K_{a_i}}$ is the optimal feedback gain. Assuming that the other players will make their worst move, this gain is calculated to minimize the quadratic cost function, equation (21), of player number i.

$$J_i(\mathbf{u_1}, \mathbf{u_2}) = \int_0^T \Big({\mathbf{x_a}}^{\mathrm{T}}(t)\mathbf{Q_i}\mathbf{x_a}(t) + {\mathbf{u_i}}^{\mathrm{T}}(t)\mathbf{R_{ii}}\mathbf{u_i}(t) + {\mathbf{u_j}}^{\mathrm{T}}(t)\mathbf{R_{ij}}\mathbf{u_j}(t) \Big) \mathrm{d}t \tag{21}$$

$\dot{\mathbf{P}}_{\boldsymbol{a_i}}$ is found by solving the continuous time couple Riccati differential equation:

$$\begin{aligned} \dot{\mathbf{P}}_{\boldsymbol{a_1}}(t) &= -{\mathbf{A_a}}^{\mathrm{T}}\mathbf{P_{a_1}}(t) - \mathbf{P_{a_1}}(t)\mathbf{A_a} - \mathbf{Q_1} + \\ &\quad \mathbf{P_{a_1}}(t)\mathbf{S_{a_1}}(t)\mathbf{P_{a_1}}(t) + \mathbf{P_{a_1}}(t)\mathbf{S_{a_2}}(t)\mathbf{P_{a_2}}(t) \\ \dot{\mathbf{P}}_{\boldsymbol{a_2}}(t) &= -{\mathbf{A_a}}^{\mathrm{T}}\mathbf{P_{a_2}}(t) - \mathbf{P_{a_2}}(t)\mathbf{A_a} - \mathbf{Q_2} + \\ &\quad \mathbf{P_{a_2}}(t)\mathbf{S_{a_2}}(t)\mathbf{P_{a_2}}(t) + \mathbf{P_{a_2}}(t)\mathbf{S_{a_1}}(t)\mathbf{P_{a_1}}(t) \end{aligned} \tag{22}$$

Using the shorthand notation $\boldsymbol{S_{a_i}} := \boldsymbol{B_{a_i}}{\boldsymbol{R_{ii}}}^{-1}{\boldsymbol{B_{a_i}}}^{\mathrm{T}}$.

IV. Simulation

used Simulink

A. LQR

LQR weighting matrices are optimized using the TCACS optimization method in the simulation. ITSE is considered for the TCACS input cost function. Here are the weighting matrices for the optimized output.

$$\mathbf{Q}_{\mathrm{LQR}} = \begin{bmatrix} 0.5215 & 0 \\ 0 & 0.0745 \end{bmatrix}, \quad R_{\mathrm{LQR}} = 0.0001 \tag{23}$$

B. LQDG

The weighting matrices used in the LQDG portion are chosen like that of the LOR.

$$\mathbf{Q}_{\mathrm{LQDG}} = \begin{bmatrix} 100 & 0 \\ 0 & 0.078 \end{bmatrix}, \quad R_{1_{\mathrm{LQDG}}} = 1, \quad R_{2_{\mathrm{LQDG}}} = 99.96 \tag{24}$$

$$\mathbf{K_1} = \begin{bmatrix} 39.1188 & 8.8510 \end{bmatrix} \tag{25}$$

C. LQIDG

LQIDG weighting matrices are chosen like the method used in the LQR and LQDG sections.

Roll:

$$\mathbf{Q}_{\mathrm{LQIDG}} = \begin{bmatrix} 0.1707 & 0 & 0 & 0 \\ 0 & 0.12 & 0 & 0 \\ 0 & 0 & 837.8606 & 0 \\ 0 & 0 & 0 & 756.1341 \end{bmatrix} \tag{26}$$

$$R_{1_{\mathrm{LQIDG}}} = 1, \quad R_{2_{\mathrm{LQIDG}}} = 7.7422$$

$$\mathbf{K_{a_1}} = \begin{bmatrix} 28.1410 & 8.4017 & 27.2223 & 11.6894 \end{bmatrix} \tag{27}$$

Roll-Pitch:

$$\mathbf{Q}_{\mathrm{LQIDG_{roll}}} = \begin{bmatrix} 585.9 & 0 & 0 & 0 \\ 0 & 31.1 & 0 & 0 \\ 0 & 0 & 83.8 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{Q}_{\mathrm{LQIDG_{pitch}}} = \begin{bmatrix} 546.5 & 0 & 0 & 0 \\ 0 & 311.4 & 0 & 0 \\ 0 & 0 & 2.22 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{28}$$

$$R_{1_{\mathrm{LQIDG}}} = 1, \quad R_{2_{\mathrm{LQIDG}}} = 7.7422$$

Roll-Pitch-Yaw:

$$\mathbf{Q}_{\mathrm{LQIDG_{roll}}} = \begin{bmatrix} 631.85 & 0 & 0 & 0 \\ 0 & 214.28 & 0 & 0 \\ 0 & 0 & 7.91 & 0 \\ 0 & 0 & 0 & 0.01 \end{bmatrix}$$

$$\mathbf{Q}_{\mathrm{LQIDG_{pitch}}} = \begin{bmatrix} 0.01 & 0 & 0 & 0 \\ 0 & 873.93 & 0 & 0 \\ 0 & 0 & 9853.09 & 0 \\ 0 & 0 & 0 & 0.12 \end{bmatrix} \tag{29}$$

$$\mathbf{Q}_{\mathrm{LQIDG_{yaw}}} = \begin{bmatrix} 0.03 & 0 & 0 & 0 \\ 0 & 0.17 & 0 & 0 \\ 0 & 0 & 1.81 & 0 \\ 0 & 0 & 0 & 0.45 \end{bmatrix} \times 10^{-4}$$

$$R_{1_{\mathrm{LQIDG}}} = 1, \quad R_{2_{\mathrm{LQIDG}}} = 1.2577$$

References

[1] Weintraub, I. E., Pachter, M., Garcia, E. (2020). An Introduction to Pursuit-evasion Differential Games. 2020 American Control Conference (ACC), 1049–1066. https://doi.org/10.23919/ACC45564.2020.9147205

[2] Garcia, E., Casbeer, D. W., Pachter, M. (2020). Optimal Strategies for a Class of Multi-Player Reach-Avoid Differential Games in 3D Space. IEEE Robotics and Automation Letters, 5(3), 4257–4264. https://doi.org/10.1109/LRA.2020.2994023

[3] [1]H. Lai, W. Liang, R. Yan, Z. Shi, and Y. Zhong, "LiDAR-Inertial based Localization and Perception for Indoor Pursuit-Evasion Differential Games," in 2021 40th Chinese Control Conference (CCC), 2021, pp. 7468–7473. doi: 10.23919/CCC52363.2021.9549330.

[4] F. Jiang, X. Guo, X. Zhang, Z. Zhang, and D. Dong, "Approximate Soft Policy Iteration Based Reinforcement Learning for Differential Games with Two Pursuers versus One Evader," in 2020 5th International Conference on Advanced Robotics and Mechatronics (ICARM), 2020, pp. 471–476. doi: 10.1109/ICARM49381.2020.9195328.

[5] M. He and X. Wang, "Nonlinear Differential Game Guidance Law for Guarding a Target," in 2020 6th International Conference on Control, Automation and Robotics (ICCAR), 2020, pp. 713–721. doi: 10.1109/ICCAR49639.2020.9108001.

[6] A. Yildiz and H. B. Jond, "Vehicle Swarm Platooning as Differential Game," in 2021 20th International Conference on Advanced Robotics (ICAR), 2021, pp. 885–890. doi: 10.1109/ICAR53236.2021.9659431.

[7] D. Fridovich-Keil, V. Rubies-Royo, and C. J. Tomlin, "An Iterative Quadratic Method for General-Sum Differential Games with Feedback Linearizable Dynamics," in 2020 IEEE International Conference on Robotics and Automation (ICRA), 2020, pp. 2216–2222. doi: 10.1109/ICRA40945.2020.9196517.

[8] T. Kessler, K. Esterle, and A. Knoll, “Linear Differential Games for Cooperative Behavior Planning of Autonomous Vehicles Using Mixed-Integer Programming,” in 2020 59th IEEE Conference on Decision and Control (CDC), 2020, pp. 4060–4066. doi: 10.1109/CDC42340.2020.9304495.

[9] S. Edhah, S. Mohamed, A. Rehan, M. AlDhaheri, A. AlKhaja, and Y. Zweiri, “Deep Learning Based Neural Network Controller for Quad Copter: Application to Hovering Mode,” in 2019 International Conference on Electrical and Computing Technologies and Applications (ICECTA), 2019, pp. 1–5. doi: 10.1109/ICECTA48151.2019.8959776.

[10] R. G. do Nascimento, K. Fricke, and F. Viana, “Quadcopter Control Optimization through Machine Learning,” in AIAA Scitech 2020 Forum, doi: 10.2514/6.2020-1148.