# AEM 8423 Project Final Report

Ahmet Semi Asarkaya

***Abstract*— In this study, I focus on an algorithm for generating collision-free quadrocopter trajectories. Starting from some initial position, the quadrcopters are desired to go to the final position within a specified time under dynamic constraints. Given initial/final information of position, velocity, and acceleration, the problem is formulated as a convex optimization problem. I utilize an iterative approach to estimate the non-convex constraints replacing them with their linear approximations and show simulation results for different scenarios.**

## I. Introduction

Trajectory/path planing has obtained more attention as the interest in multiple flying machines increased. These machines may interact with each other and the environment. Therefore, the need for collision-free trajectories while optimizing some other parameters such as fuel consumption, trajectory length etc.. has emerged.

There are plenty of studies in the literature regarding the subject. Different types of convex optimization methods are utilized to find collision-free trajectories with some performance criteria [1], [2], [3], [4].

In [3], the problem is cast as a model predictive control problem. Trajectories are generated by formulating the trajectory of the quadrocopter in its jerk and then solving a convex optimization problem on each decoupled axis. The authors consider a very specific type of quadrocopter dynamics and derive the constraints accordingly.

In [1], the authors develop a method generating trajectories that account for simple dynamics constraints. Hence, they claim it is independent of the vehicle's type. My work is very similar to [1]. Although the main ideas are very similar, I use different constraints at some parts of the problem and I am using a different approximation approach for non-convex constraints. Also, the authors in [1] use a different initialization approach.

This study is outlined in the following order: In section II, trajectory dynamics and construction of the constraints are explained in detail. Also, the formulation of the problem as a convex optimization problem and how to use the iterative approach for approximating non-convex constraints are discussed. In section III, simulation results are discussed.

## II. Problem Formulation

### A. Relations for Trajectory Generation

Velocity and the position of every agent at each time step, $V_i^t$ and $P_i^t$, can be related using simple discrete time equations. Equations 1 and 2 show these equations.

Ahmet S Asarkaya is with Department of Aerospace Engineering and Mechanics, University of Minnesota, Minneapolis, MN 55414, USA, asark001@umn.edu

$$P_i^{t+1} = P_i^t + \Delta t V_i^t + \frac{\Delta t^2}{2} a_i^t \tag{1}$$

$$V_i^{t+1} = V_i^t + \Delta t a_i^t \tag{2}$$

where $\Delta t$ is the size of the time steps, $t = 0, 1, ..., T$ is the current time step, and $i = 1, ..., N$ is the agent number. T and N represent the final time step and the total number of agents. From equations 1 and 2, the position and the velocity can be expressed as affine functions of acceleration. Then, the equations 3 and 4 can be obtained as follows:

$$P_i^t = P_i^0 + t\Delta t V_i^0 + \frac{\Delta t^2}{2}\Big[\sum_{k=1}^{t} a_i^k (2(t-k)+1))\Big] \tag{3}$$

$$V_i^t = V_i^0 + \Delta t\Big[\sum_{k=1}^{t} a_i^k\Big] \tag{4}$$

where $t = 1, ..., T-1$ and $i = 1, ..., N$. Using the equations 3 and 4, an optimization problem can be formulated with the optimization parameter $X$, which is ] the acceleration of the every agents at each time steps, i.e, $X \in \mathbb{R}^{3NT}$

### B. Objective Function

For a quadrocopter, the control inputs of the systems are the total thrust produced,($f = \|T\|/m$) and the angular velocities, ($\Omega = [\omega_1\ \omega_2\ \omega_3]^T$). In this study, I will consider a normalized thrust force which has the acceleration units for simplicity. Another approach might be considering the jerk as the only input to the system. A relation for jerk can be obtained using the total thrust produced,($f = \|T\|/m$) and the and angular velocities.

$$a = R\begin{bmatrix}0\\0\\f\end{bmatrix} + \begin{bmatrix}0\\0\\-g\end{bmatrix} \tag{5}$$

R is the proper rotation matrix between the inertial and the body fixed frame. Knowing that the norm of the rotation matrix is 1, the total thrust produced can be calculated using the equation 5 as follows:

$$f = \|a - g\| \tag{6}$$

It is required to obtain $\dot{R}$ in terms of known parameters to calculate the jerk. $\dot{R}$ can be expressed by rotation matrix, $R$ and the skew symmetric matrix of body rates, as follows:

$$\dot{R} = R \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \tag{7}$$

Using the equations 5 and 7, simply taking the derivative of the acceleration, the jerk can be defined as follows :

$$j = \dot{R} \begin{bmatrix} 0 \\ 0 \\ f \end{bmatrix} + R \begin{bmatrix} 0 \\ 0 \\ \dot{f} \end{bmatrix} \tag{8}$$

$$j = R( \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ f \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dot{f} \end{bmatrix} ) \tag{9}$$

Therefore, the sum of the total thrust of agents at every time step can be chosen as the cost.

$$f_o = \Big[ \sum_{t=1}^{T} \sum_{i=1}^{N} \left\| \vec{a}_i^t + \vec{g} \right\|^2 \Big] \tag{10}$$

The equation 10 can be expressed in a quadratic form where $G$ represents the gravity vector of each agent at every time step, i.e, $G \in \mathbb{R}^{3NT}$.

$$f_o = X^T X + 2G^T X + G^T G \tag{11}$$

Equation 11 is in a quadratic form. Therefore, it is possible to use this cost function with proper affine equality/inequality constraints. Therefore, this problem can easily be solved by utilizing available quadratic programming solvers.

### C. Body Rates

After finding the acceleration of the agents at each time step, the total thrust force can be found using the equation 6. Rearranging the equation 8, following equation can be obtained:

$$\begin{bmatrix} \omega_2 \\ -\omega_1 \\ 0 \end{bmatrix} = \frac{1}{f} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} R^T j \tag{12}$$

The rotation about the 3 axis, does not have any effect on the acceleration. Therefore, it can be assumed that $\omega_3 = 0$ for simplicity. Equation 12 can be used for a real life implementation of the algorithm.

To put it simply, appropriate inputs, body rates/thrust can readily be calculated using the output of the optimization problem, which is the acceleration of the agents at each time step. However, I do not include such analyze as it is not the main goal of this study.

### D. Equality Constraints

It is required to choose initial velocity and the position values to start the algorithm. Also, the desired final values of the acceleration, velocity and the position need to be specified. It is possible to impose the constraints on initial velocity and position using the equations 3 and 4. To impose the constraints on the final acceleration, velocity and position, equality constraints can be used. Considering that the constraints are imposed in each axis for all final acceleration, velocity, and the position, there must be 9 equality constraints in total for every agent. Therefore, there needs to be $9N$ boundary constraints in the system. These constraints can be expressed as equality constraints in the following form:

$$A_{eq} X = b_{eq} \tag{13}$$

where $A_{eq} \in \mathbb{R}^{9N \times 3N(T+1)}$ and $b_{eq} \in \mathbb{R}^{9N}$.

### E. Inequality Constraints for feasibility

Constraining the jerk $j_i$, which is the first derivative of the acceleration is necessary for the continuity in the acceleration of the quadrocopter [2], [3]. Jerk should be bounded for smooth and feasible trajectories. For simplicity, the jerk can be constrained in each axis separately.

$$j_{min} \leq j_i^t \leq j_{max} \tag{14}$$

In discrete time, jerk of any agent at any time, $j_i^t$ can be expressed by the equation 15.

$$j_i^t = \frac{a_i^t - a_i^{t-1}}{\Delta t} \tag{15}$$

Bounding the jerk by $j_{max}$ and $j_{min}$ requires $6TN$ inequality constraints of which $3TN$ is for upper bound, and $3TN$ is for lower bound. To obtain a feasible solution set for real applications, dynamical constraints on acceleration should be taken into account.

$$a_{min} \leq a_i^t \leq a_{max} \tag{16}$$

Similar to the jerk constraints, $6(T+1)N$ inequality constraints are required to keep the acceleration input in feasible regions.

$$[a_{min}] \leq X \leq [a_{max}] \tag{17}$$

Putting all these constraints together, the following inequality can be obtained:

$$A_{feasibility} X \leq b_{feasibility} \tag{18}$$

where $A_{feasibility} \in \mathbb{R}^{6N(2T+1) \times 3N(T+1)}$ and $b_{feasibility} \in \mathbb{R}^{6N(2T+1)}$ . In other words, $A_{feasibility}$ includes the feasibility constraints for both jerk and acceleration. It is also possible to include constraints to simulate physical environment with certain size limitations. For example, a lab environment with limited space can be represented by constraints on position. This could be achieved by using the equation 3. Depending on the dynamics of the vehicle and the mission, velocity can also be constrained by using the equation 4.

### F. Collision Avoidance Constraints

To avoid the collision between the drones, a minimum distance between them must be kept at each time step.

$$\left\|P_i^t - P_j^t\right\| \geq d_{min} \;\; \forall i,j \;\; i \neq j, \;\; \forall t \tag{19}$$

Equation 19 is a non-convex constraint. Therefore, it cannot directly be included in the optimization problem. An iterative approach is used to solve this issue. However, it is first required to linearize the equation 19 around an equilibrium point, $\hat{X}$. This method is also called sequantial convex programming [5]. It is a local optimization algorithm especially for non-convex problems. The aim is to change non-convex constraints by convex estimations around a previous guess, $\hat{X}$. Then, the problem is solved iteratively until a desired convergence criteria is satisfied. Linearized version of 19 can be written as follows:

$$\begin{aligned} \left\|P_i^t - P_j^t\right\|^2 &= (P_i^t - P_j^t)^T(P_i^t - P_j^t) \\ \geq 2(\hat{P}_i^t - \hat{P}_j^t)^T(P_i^t - P_j^t) &- (\hat{P}_i^t - \hat{P}_j^t)^T(\hat{P}_i^t - \hat{P}_j^t) \\ &\geq d_{min}^2 \;\; \forall i,j \;\; i \neq j, \;\; \forall t \end{aligned} \tag{20}$$

This particular linearization is done using iterative convex overbounding concept presented by the authors in [6]. The main idea of using this linearization is that it overbounds the true constraint and hence, the conservatism is small. Equation 20 can be expressed as affine functions of acceleration, $X$. This requires $TN(N-1)/2$ inequality constraints.

$$A_{collision}X \leq b_{collision} \tag{21}$$

where $A_{collision} \in \mathbb{R}^{TN(N-1)/2)\times 3N(T+1)}$ and $b_{collision} \in \mathbb{R}^{TN(N-1)/2}$. Putting equations 18 and 21 together, the following inequality can be obtained:

$$AX \leq b \tag{22}$$

where $A \in \mathbb{R}^{(0.5TN^2+11.5TN+6N)\times 3N(T+1)}$ and $b \in \mathbb{R}^{0.5TN^2+11.5TN+6N}$. $A$ matrix consists of feasibility and collision avoidance constraints. However, it is required to have an initial guess of $\hat{X}$ because of the terms in the linearized version of the collision avoidance constraint.

### G. Optimization Problem and Proposed Algorithm

Optimization problem can be solved in two steps. At the first step, an initial optimization without including collision avoidance constraints is done. The result $\hat{X}$ is then used to initiate the second step.

1) Find an initial guess, using the feasibility constraints.

$$\begin{aligned} &\min_X f_0 \\ \text{subjected to} \;\; &A_{eq}X = b_{eq} \\ &A_{feasibility}X \leq b_{feasibility} \end{aligned} \tag{23}$$

Obtain $\hat{X}$

2) Use $\hat{X}$ to form new inequality matrices, A and b.

$$\begin{aligned} &\min_X f_0 \\ \text{subjected to} \;\; &A_{eq}X = b_{eq} \\ &AX \leq b \end{aligned} \tag{24}$$

In the second step, an iterative approach is used where at each iteration, the latest estimate of X, $\hat{X}$ is utilized. Given a threshold, $f_{threshold}$, If $f_0$ - $(f_0)_{prev} \geq f_{threshold}$, then the step two is repeated.

Since all the constraints are convex, or convexified, quadratic programming tools can be used to solve the optimization problem.

## III. Results and Discussion

In this section, I present a case study as well as simulation results showing the performance of the proposed method. The algorithm is implemented on MATLAB R2019a, and performed on a PC with an Intel i7-7700HQ CPU at 2.8 GHz processor and 16.0 GB RAM. Specifically, I use MATLAB's quadratic programming function(quadprop). In the case study, a fleet of 8 quadrocopters are considered in a 6x6x4 three dimensional environment. I first run the first step of the simulation, in other words, there is no any distance constraint imposed yet. Figure 1 shows that the algorithm generate the optimal trajectories which are the shortest distance between the initial and final points. Since there is no constraints regarding the collision, the trajectories are intersecting each other at some points.

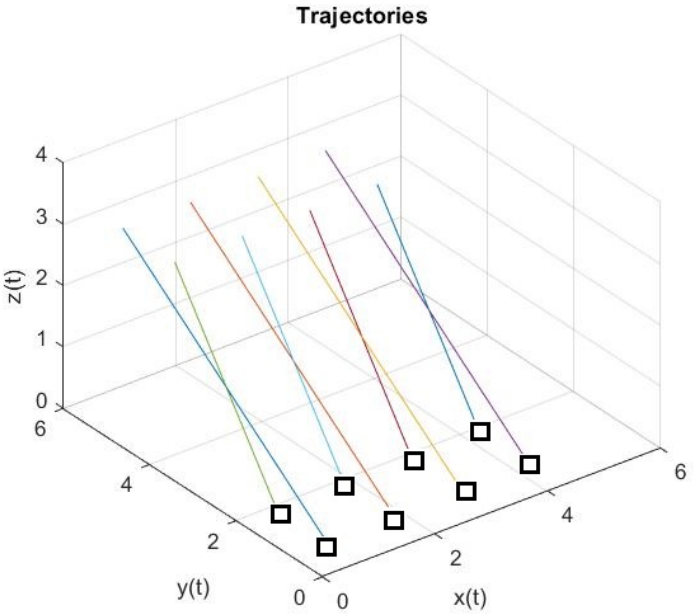


Fig. 1. Trajectories for 8 quadrocopters without imposing minimum distance constraints. Rectangles indicate the initial positions of the quadrocopters.

The parameters used for case study are chosen as follows:

- Number of agents = 8
- Simulation length = 30 seconds

- Discretization step-size, dt = 0.2 seconds
- Number of inequality constraints without distance constraints = 7200
- Total number of inequality constraints with distance constraints = 11400
- Cost Threshold, $f_{threshold}$ = 0.05
- Number of iterations for convergence = 2
- Run time for first step of the algorithm = 1.85 seconds
- Total run time = 35 seconds

When the second step of the algorithm is run, trajectories are automatically modified by the algorithm so that the collision is avoided. Figure 3 shows that the change in the trajectories is minimum while it is guaranteed that the trajectories are collision-free. In this case study, the minimum distance is taken as 1 meter, $d_{min} = 1$. Depending on the quadrocopter's size and the size of the environment, this parameter can be chosen accordingly.

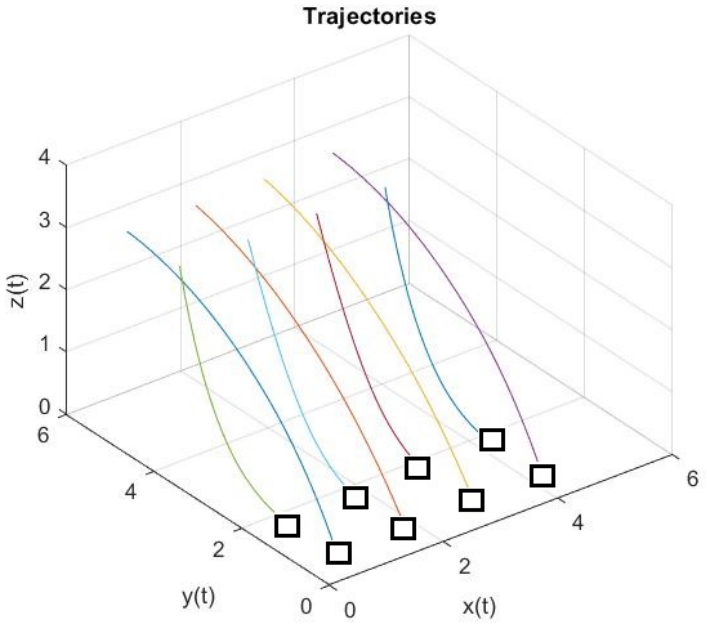


Fig. 2. Collision-free trajectories for 8 quadrocopters. Rectangles indicate the initial positions of the quadrocopters.

Run time of the algorithm depends on many different factors. The main factor is the size of the inequality matrix which can be found by $A \in \mathbb{R}^{(0.5TN^2+11.5TN+6N)\times 3N(T+1)}$. Number of agents, simulation time, the step-size of the discretization are the parameters deciding the size of this matrix. There are also other factors affecting the run time. For example, the cost threshold set for convergence of the second step of the algorithm is directly affecting the run time. If a large threshold is chosen, it may converge in a few steps. On the other hand, a small threshold cause a huge number of iteration which results in a longer run time.

Figure 3 shows the computation time for different scenarios. Different cases may result in the same number of inequality constraints. For example, increasing the number of

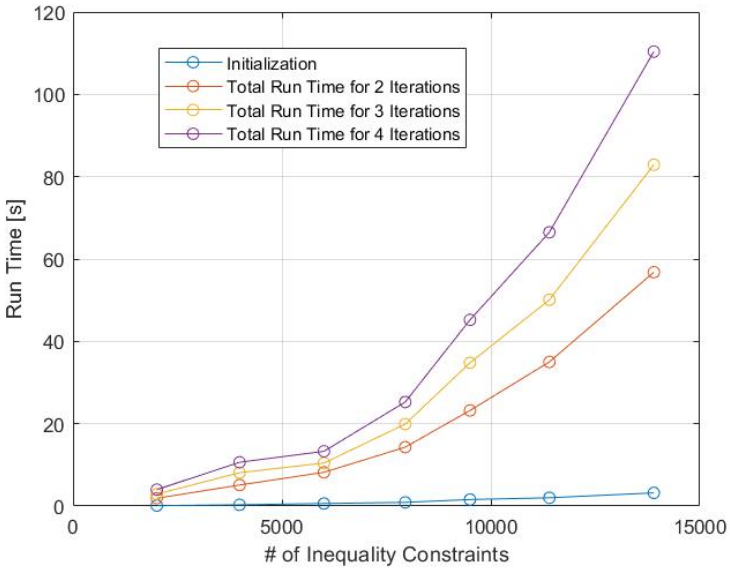


Fig. 3. The average run time for different number of iterations and different number of inequality constraints (10 trials are done for each data point).

agents while decreasing the simulation time accordingly may not change the number of required inequality constraints. Such trade-offs can be taken into consideration for different types of missions, environments, vehicles.

## IV. Conclusion

In this study, I present an algorithm that enables the generation of collision-free trajectories for a multi-agent quadrocopter fleet. This method allows for quadrocopters to take-off from a given initial position and fly to the desired final position within a specified flight time. It is guaranteed that the quadrocopters are not going to collide with each other. Since the control effort is minimized, these trajectories are the shortest possible trajectories under the collision avoidance constraints. Simulation results showing the computational effort and the performance indicates that this algorithm can be used in a real-life application with well-tuned hyper-parameters. In other words, these collision-free trajectories can be generated online.

## V. Acknowledgments

The author thanks Prof. Ryan James Caverly for his continues feedback during the project.

## References

[1] F. Augugliaro, A. P. Schoellig, and R. D'Andrea, "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach," in *2012 IEEE/RSJ international conference on Intelligent Robots and Systems*, pp. 1917–1922, IEEE, 2012.

[2] J. Alonso-Mora, T. Naegeli, R. Siegwart, and P. Beardsley, "Collision avoidance for aerial vehicles in multi-agent scenarios," *Autonomous Robots*, vol. 39, no. 1, pp. 101–121, 2015.

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

[4] X. Sun, S. Chai, and B. Zhang, "Trajectory planning of the unmanned aerial vehicles with adaptive convex optimization method," *IFAC-PapersOnLine*, vol. 52, no. 12, pp. 67–72, 2019.
[5] S. Boyd, "Sequential convex programming," *Lecture Notes, Stanford University*, 2008.
[6] R. J. Caverly and J. R. Forbes, "Lmi properties and applications in systems, stability, and control theory," *arXiv preprint arXiv:1903.08599*, 2019.