Project Report

Analysis of 4th Order UAV Multi-agent Systems

Sushant Trivedi
(Masters in Computer Engineering)
School of Computing, Informatics, & Decision Systems Engineering, Arizona State University

Ninad Jadhav
(Masters in Computer Science)
School of Computing, Informatics, & Decision Systems Engineering, Arizona State University

***Abstract*—This report has attempted implementation and analysis of collision avoidance between two teams of multi-robot systems [1]. The objective has been divided into 3 main subproblems – Formation Flight, Trajectory Planning, Collision Avoidance. For these, the author has referred us to his previous works in order [2][3][4]. Here, we successfully modeled Formation Control using paper [2] and it has been comprehensively analysed in this report by changing various tuning parameters like sampling time, lattice structure.**

The software framework is flexible to simulate formation control for large count of UAVs and various formation with minimal effort.

***Index Terms*—MPC, UAV, Formation Control, Leader Follower, Consensus**

a.) INTRODUCTION

The paper **[1]** we selected proposed collision avoidance algorithm among two multi–robot teams. This paper is built upon the work of his previous papers [2][3][4]. In paper [2], they defined 4th order UAV dynamics and implemented consensus based formation control algorithm. Using this paper, they published paper [3][4], where they tackled collision avoidance while trajectory planning is being done using MPC. To tackle collision avoidance they modified MPC optimization problem's cost function. Furthermore they used Kalman Filter to estimate obstacles and leader positions for this MPC problem.

In this report, we start with a detailed analysis of [2]. Section II gives an overview of the fourth order flight dynamics for a UAV system and algorithm for the formation control. In section III we present the results obtained by tweaking various parameters. Section IV gives an improvement to the algorithm that solves failure in *chain connection* when follower is connected to all the other UAVs including the leader. Section V gives details about the framework implementation and steps for using it. Finally in Section VII we discuss current work undergoing testing to be presented during class presentations [1].

b.) FOURTH ORDER FLIGHT DYNAMICS FOR UAV SYSTEM

In [2], the authors deal with challenges in cooperative control for a multi-UAV team expressed as a fourth-order system using a consensus-based algorithm. A linearized model of UAVs (quadrotors) as a fourth-order system is described in the paper, and then a formation control algorithm for this model has been proposed. The proposed algorithm uses a consensus algorithm based control law to maintain formation lattice. Their study shows that proposed control algorithm can guarantee accurate formation following under certain assumptions about the network of the multi-agent system. The proposed algorithm has been validated by some simulations. A detailed representation of the single system model are equations in [2]. This has been simplified in paper [1] and mentioned below. The formation lattice has been defined in (4) using Adjacency matrix - A_{ij}.

$$X(k+1) = AhX(k) + Bhu(k) \tag{1}$$

with

$$A_h = \begin{bmatrix} \bar{A} & 0 \\ 0 & \bar{A} \end{bmatrix}, B_h = \begin{bmatrix} \bar{B} & 0 \\ 0 & \bar{B} \end{bmatrix}, \tag{2}$$

and

$$\bar{A} = \begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} & \frac{\Delta t^3}{6} \\ 0 & 1 & \Delta t & \frac{\Delta t^2}{2} \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix}, \bar{B} = \begin{bmatrix} \frac{\Delta t^4}{24} \\ \frac{\Delta t^3}{6} \\ \frac{\Delta t^2}{2} \\ \Delta t \end{bmatrix}. \tag{3}$$

$Xk = [x\ u\ \theta\ q\ y\ v\ \phi\ p]^T$

They represent [position velocity roll angular velocity]

Control Law for each agent:

$$M_i(t) = -\sum_{j=1}^{N+1} a_{ij} \left[\sum_{k=0}^{3} \beta_k (\hat{r}_i^{(k)} - \hat{r}_j^{(k)}) \right], \quad i \in \{1, 2, \cdots, N\},$$

Constraints on Tuning Parameters (β_k):

$$\beta_k > 0, \quad \forall k \in \{0, 1, 2, 3\},$$

$$\lambda_{min} > \frac{\beta_1^2}{\beta_1\beta_2\beta_3 - \beta_0\beta_3^2},$$

$$\beta_1\beta_2 > \beta_0\beta_3.$$

In the next section we analyze this algorithm by modifying various parameters and *present results that have not been explored or discussed by the authors in* [1] [2] [3].

c.) ROBUSTNESS ANALYSIS OF FORMATION CONTROL

In this section, in addition to detailed analysis, we highlight challenges faced and scenarios where the algorithm fails and suggest a modification to algorithm.

A] Analysis of Formation Control with different Lattice structure.

i) Using the lattice structure (Fig 1), our simulation has shown successful formation following of the leader as shown in Fig 2.

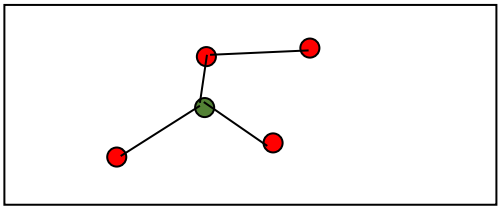


Fig1

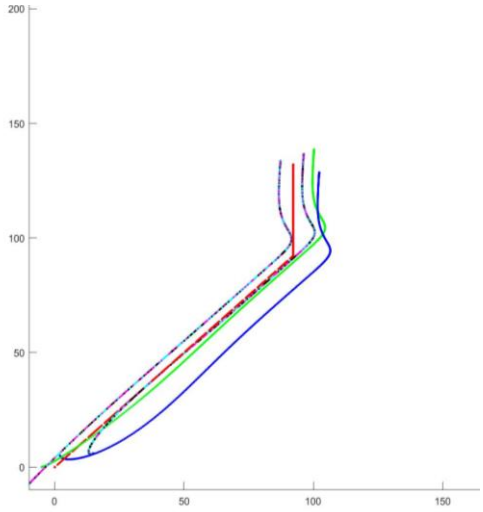
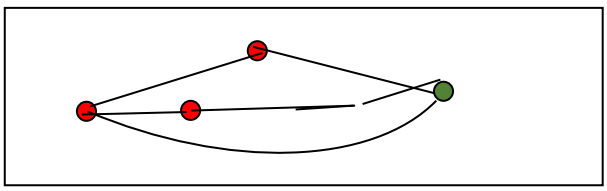


Fig2 : A_{ij} = [0,1,0,0,0; 0,0,0,0,1; 0,0,0,1,0; 0,0,0,0,1; 0,0,0,0,0];

ii) Under the assumptions of algorithm in paper [2], *we found that in lattice when one of the UAVs is connected to every other UAV and if not all of the other UAV are connected to the leader directly, the trajectory for this UAV agent performs erratically* .i.e. it does a random walk. In Fig3, below this is UAV1. Rest of the agents maintain their structure.

Fig4



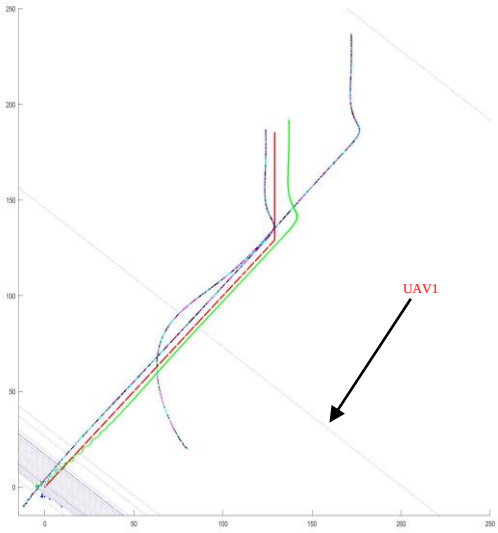


Fig5 : A_{ij} = *[0,1,1,1,1; 0,0,1,0,0; 0,0,0,1,0; 0,0,0,0,1; 0,0,0,0,0];*

iii) When the lattice is a spanning tree with each agent connect only to 2 other agents (Fig4), the trajectories of the UAV which is farthest from the leader is oscillating when formation starts to move or changes directions as shown in Fig5. If the direction remains unchanged for a considerable time, the trajectory stabilizes as shown by UAV1 in figure.

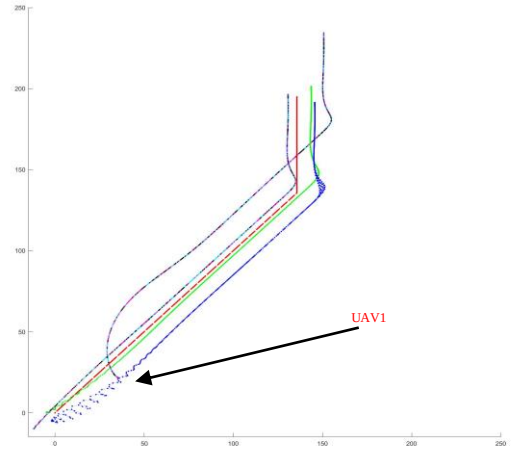
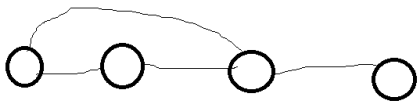


Fig6 : *Adj* = *[0,1,0,0,0; 0,0,1,0,0; 0,0,0,1,0; 0,0,0,0,1; 0,0,0,0,0];*

Interestingly if the *lattice is modified such that the end agents of the spanning tree are connected to the agent before the one it is connected to currently (Fig 6) then there are improved trajectory behavior.(Fig 7).* This behavior is seen in Fig 8 as UAV1 and UAV2.



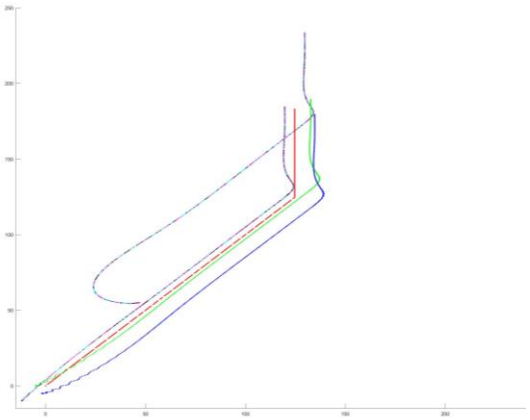


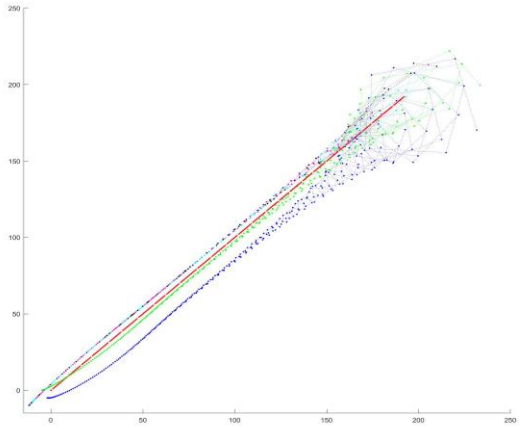
Fig8 : = *[0,1,1,0,0; 0,0,1,0,0; 0,0,0,1,0; 0,0,0,0,1; 0,0,0,0,0];*

Some other simulations involved using 10 UAVs, single or 2 connections per UAV. The trajectories were found to converge in these scenarios (convergence time ~ 50 secs).

B] Sampling Time Analysis of the Model.

i) In implementation, in order to reduce computation we stop our control law calculation for any agent whose position has converged to desired position with accuracy = 0.001 with the Sampling time of the model (Td) = 0.2. But if this convergence condition is kept in the same order as our sampling time (= 0.1). The algorithm stops on the first passing of the desired position.
From this we observed that the convergence doesn't assure that the agents will remain at that position. This will be defined by the processing computing vs sensor acquisition sampling rate. This has been demonstrated in Fig8, 9.

ii) As the time-step values increase, our model become inaccurate and the algorithm fails. This is analogous to having a low cost sensor. Hence, there is a lower bound on the sensor capacity. Fig 10 shows such behavior for t=0.3.



Plot 5: *t=0.3, no convergence*

C] Formation Simulation with Various Beta Parameters

In the control law, there are β_k parameter each state of the agent. They define the state contribution to figuring how the UAV should move with respect to position. This behavior is shown below (Fig 11,12) 1D simulation where the agent overshoots when the weightage on roll and pitch angles are higher and it acts reactively and stops at desired position when β_k parameters weigh all states equally.

Fig8,9

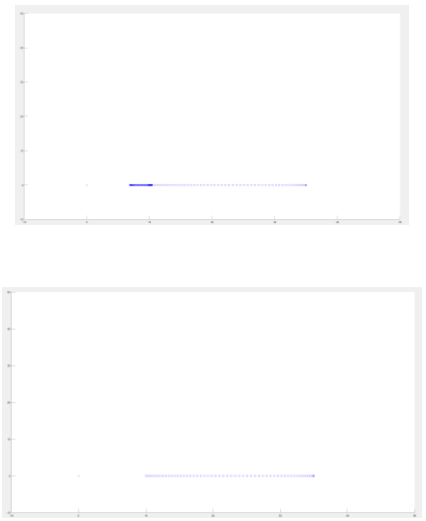
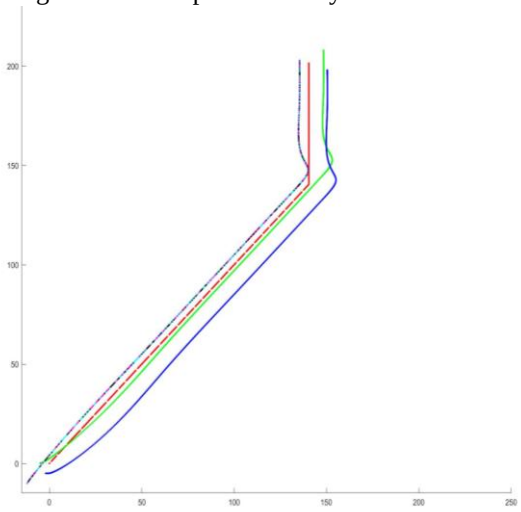


Fig11,12

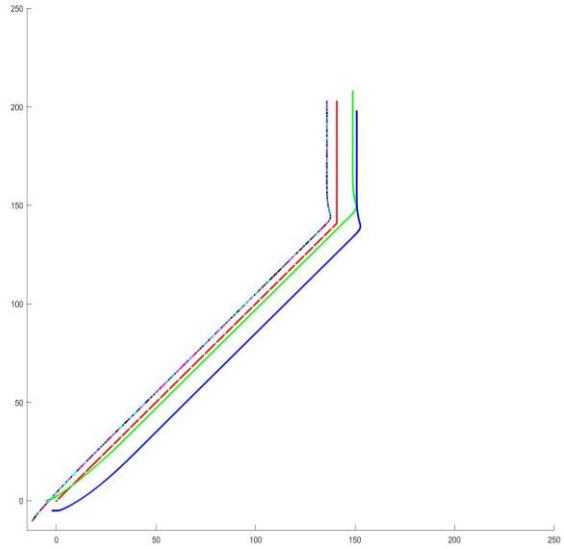
a) Changing only β_2 values.

This parameter controls the orientation of the UAVs (roll, pitch).

i) Using the default values, we get the desired behavior (with some possible path collision) as shown below because there is more chances of collision when there is frequent changes in roll and pitch of the system.



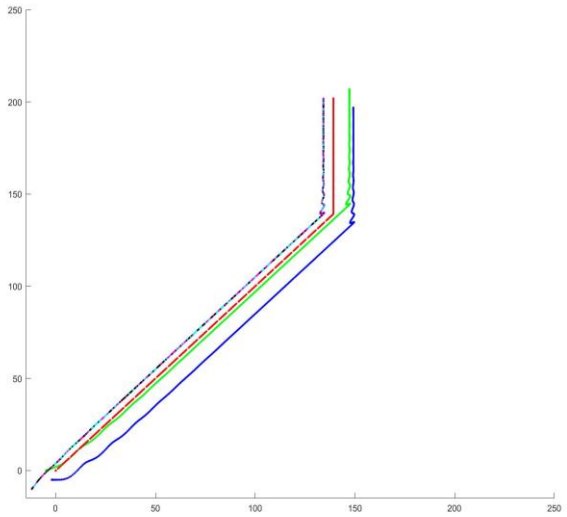
Plot 7: *Beta = [1,5,20,3], where Beta2 = 20, convergence = 47s*



Plot 8: *Beta = [1,5,10,3], where Beta2 = 10, convergence ~ 36s*

ii) Decreasing the β_2 value, but still satisfying the constraints as given in the equations, gives us faster convergence and more quicker turns (avoids path collision with leader).

iv) Simulating the λ_{min} constraint by a small margin for $\beta_2 = 3$, causes the UAV trajectories to get wiggly. Lowering the β_2 value further will cause the trajectory to go erratic.

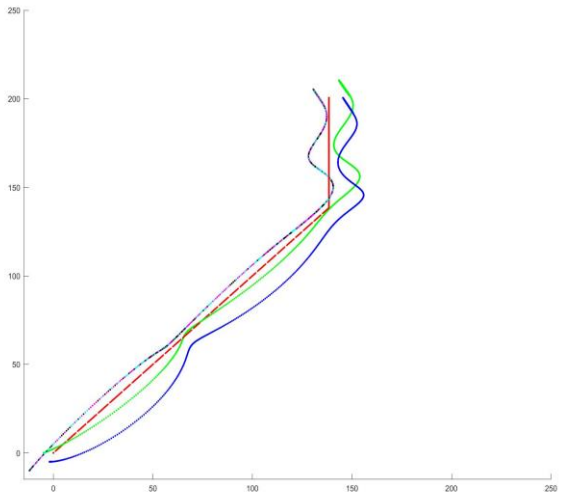


Plot 9: β_k= *[1,5,3,3], where Beta2 = 3,*

b.) Tweaking β_1 values

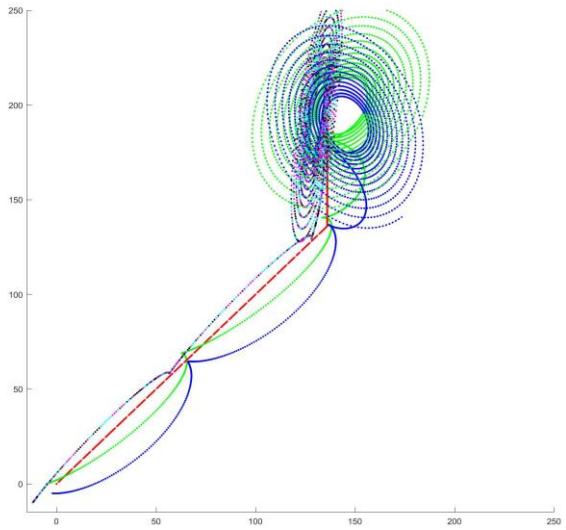
B_1 values determines the weight of the Velocity for the UAVs.

i) Giving smaller value of $\beta_1 = 1$ as compared to the default value cause the follower UAVs to move faster than the leader and slow down before traversing a specific distance so that the leader can catch up. The trajectories also get oscillating and we see a higher convergence time.



Plot 10: *Beta = [1,1,20,3], where Beta1 = 1,convergence = 132s,*

ii) Breaking the λ_{min} constraint using β_1=0, causes the UAVs to circle around the leader with increasing radius, once the leader stops. The formation never reaches convergence. The initial behavior was similar to the case when β_1 =1



Plot 11: *Beta = [1,0,20,3], where Beta1 = 0, No convergence*

This corroborates the mathematical proof given in paper [2] for the λ_{min} constraint.

c.) ALGORITHM MODIFICATION

In order to avoid *erratic and uncontrollable movement of agents* as seen in Section III Fig3, we propose a modification to the original algorithm. If an agent is directly connected to the leader, we discard communication information from other nodes solving the issue and reducing our communication loads on the network. This will be true for agents in vicinity of leader.

This has been tested as shown in Fig , and source code is (*UAVMultiAgent_AlgorithmModified.m*).

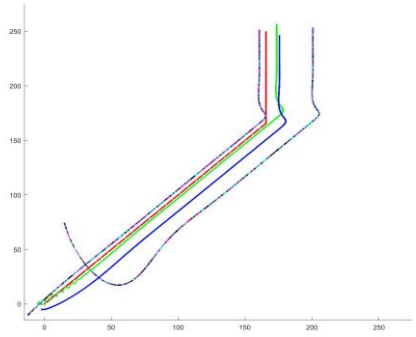


Fig : *Adj = [0,1,1,1,1; 0,0,1,0,0; 0,0,0,1,0; 0,0,0,0,1; 0,0,0,0,0];*

VI. CURRENT WORK UNDERGOING

a.) Collision Avoidance using MPC

In Paper [3] and [4], they have used MPC based control for collision avoidance and Kalman Filter for Position prediction. ***We have implemented the code but the behavior encoded there is not seen in our code. Possible issues that we guess are either it's a logical error in the code implementation or the tuning parameter matrix used for MPC Cost function (P,Q,R) in paper [2] are incorrect for us. This behavior should be debugged by the class presentation.***

b.) Hardware Implementation of Robotarium

In order to test these ideas on hardware, we planned to use Robotarium Testbed. But as the model of UAV is 4th order and Robotarium bots 1st order differential drive based, we weren't getting consensus formation control in our implementation for more than 2 robots. *We plan to complete this by class presentation.*

VII. SOFTWARE FRAMEWORK

We not only implemented the modified algorithm but also developed a general purpose framework that enables a user to simulate the formation for *N* – followers and 1 leader, with minimal effort. Some of the features of the framework as follows:

- Quick simulation with only modification required is the total number of agents (no_agent) and the adjacency matrix that defines the connections between different UAVs in the formation
- Keep default positions for 3 follower UAVs and the leader, other start & desired positions for each new UAV are generated dynamically.

VIII. CONCLUSION

The challenges faced in the implementation has been that each paper didn't really refer to the previous paper about the mathematical explanation in a sequential manner. Thus, we had to understand all the paper but each paper have different notations for state matrices and other term which was not explicitly evident from the paper. Thus, in this paper we have analysed the consensus based algorithm for different agent count, tuning parameters.

IX. ACKNOWLEDGEMENT

We acknowledge and thank Prof. Stephanie Gil for all her help.

X. REFERENCES

[1] Collision Avoidance between Multi-UAV-Systems considering Formation Control using MPC

[2] Y. Kuriki and T. Namerikawa: Formation control of UAVs with a fourth-order flight dynamics, SICE Journal of Control, Measurement, and System Integration, Vol. 7, No. 2, pp. 74–81,

[3] Y. Kuriki and T. Namerikawa, Formation Control with Collision Avoidance for a Multi-UAV System Using Decentralized MPC and Consensus-Based Control, SICE Journal of Control, Measurement, and System Integration, Vol.8, pp.285-294, 2015

[4] Formation Control of Multi-Agent System Considering Obstacle Avoidance.

Ryo Toyota