EVALUATION OF GUIDANCE METHODS FOR A SWARM OF MUNITIONS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ABDULLAH ALP MUHIDDINOĞLU

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
AEROSPACE ENGINEERING

FEBRUARY 2021

Approval of the thesis:

EVALUATION OF GUIDANCE METHODS FOR A SWARM OF MUNITIONS

submitted by **ABDULLAH ALP MUHIDDINOĞLU** in partial fulfillment of the requirements for the degree of **Master of Science in Aerospace Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. İsmail Hakkı Tuncer
Head of Department, **Aerospace Engineering** _____

Prof. Dr. Ozan Tekinalp
Supervisor, **Aerospace Engineering, METU** _____

Examining Committee Members:

Prof. Dr. Kemal Leblebicioğlu
Electrical and Electronics Engineering, METU _____

Prof. Dr. Ozan Tekinalp
Aerospace Engineering, METU _____

Assist. Prof. Dr. Ali Türker Kutay
Aerospace Engineering, METU _____

Assist. Prof. Dr. Ali Emre Turgut
Mechanical Engineering, METU _____

Assist. Prof. Dr. Kutluk Bilge Arıkan
Mechanical Engineering, TEDU _____

Date: 10.02.2021

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Abdullah Alp Muhiddinoğlu

Signature :

ABSTRACT

EVALUATION OF GUIDANCE METHODS FOR A SWARM OF MUNITIONS

Muhiddinoğlu, Abdullah Alp
M.S., Department of Aerospace Engineering
Supervisor: Prof. Dr. Ozan Tekinalp

February 2021, 99 pages

In this study, collective flight, mission execution and decision-making algorithm for airdrop micro munitions with two different swarm algorithm alternatives are examined. Through a concept munition design is created and an aerodynamic database is prepared with the Missile-Datcom software. Six degrees of freedom flight mechanics model, subsystems of munition and environment are modeled. A linear model is obtained and controller is designed. Pole placement method is used in the controller design. Full state feedback assumption may not suitable for the actual problem. For this reason, the autopilot gains of the designed controller are converted into gain values suitable for measurements by using the projection control method. The designed autopilot tracks the commands of the swarm and proportional navigation guidance. The swarm algorithms coefficients are optimized for the nominal state by particle swarm optimization. Alternative algorithms obtained as a result of optimization are tested under various distortions by Monte Carlo analysis. Sensitivity analysis and performance evaluation are examined as well.

Keywords: Missile Dynamic, Swarm Algorithm, Six Degrees of Freedom Simulation, Servo Systems Design, Projective Control, Decision Algoritihm, Sensitivity Analysis, Particle Swarm Optimization, A Computer Simulation

ÖZ

MÜHİMMAT SÜRÜSÜ İÇİN GÜDÜM YÖNTEMLERİNİN DEĞERLENDİRİLMESİ

Muhiddinoğlu, Abdullah Alp
Yüksek Lisans, Havacılık ve Uzay Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Ozan Tekinalp

Şubat 2021 , 99 sayfa

Bu çalışmada iki farklı sürü algoritması alternatifi ile motorsuz mikro mühimmatlar için toplu uçuş, görev icrası ve karar verme algoritması incelenmiştir. Konsept mühimmat tasarımı oluşturulmuş ve Missile-Datcom yazılımı ile aerodinamik veri tabanı hazırlanmıştır. Altı serbestlik dereceli uçuş mekaniği modeli, mühimmata yer alan alt sistemler ve ortam modellenmiştir. Lineer model elde edilmiş ve kontrolcü tasarlanmıştır. Kontrolcü tasarımında kutup yerleştirme yöntemi kullanılmıştır. Tam durum geribesleme kabulü gerçek uygulamalar ile örtüşmemektedir. Bu nedenle tasarlanan kontrolcüye ait otopilot kazançları izdüşüm kontrol yönteminden faydalanılarak ölçümlere uygun kazanç değerlerine dönüştürülmüştür. Tasarlanan otopliot, sürü güdüm komutlarını ve oransal seyrüsefer güdüm koutlarını takip etmiştir. Sürü algoritmalarına ait katsayılar, nominal durum için parçacık sürüsü optimizasyonu ile eniyilenmiştir. Eniyileme sonucunda elde edilen alternatif algoritmalar çeşitli boznumlar ve hatalar altında Monte Carlo analizi ile sınanmıştır. Elde edilen çıktılara göre hasasiyet analizi ve performans değerlendirmesi yapılmıştır.

Anahtar Kelimeler: Füze Dinamiği, Sürü Algoritması, Altı Serbestlik Dereceli Model, Servo Otopilot Tasarımı, İzdüşüm Kontol, Karar Verme Algoritması, Hassasiyet Analizi, Parçacık Sürüsü Optimizasyonu, Bilgisayar Simülasyonu

To my family whose support I always feel,

To my love who always encouraged me.

ACKNOWLEDGMENTS

I would like to thank everyone who helped me directly or indirectly in completing my thesis in these days when the world is plagued with epidemics. First of all, I would like to thank to my advisor Prof. Dr. Ozan Tekinalp for his interest, support and patience towards me. I could not have acquired this much knowledge, without his experience, knowledge and guidance.

I would like to express my gratitude to Roketsan Missile Inc. for giving me the awareness of swarm systems in the early period of my engineering life, encouraging me in this regard, and supporting me in completing this thesis.

I would like to thank my colleagues Dr. Kaan Görür, Oğuz Han Altıntaş and our manager Dr. Özgür Ateşoğlu, who shared their support, wisdom and experiences with me during my thesis.

I would like to thank my friends and colleagues Berkan Yerlikaya, Koray Yayla, Mehmet Başaran and Onur Emre for their patience, friendship, kindness, and making me feel stronger. Without them, everything would be much more difficult. I would like to specially thank Koray Yayla for his full support in every phase of my thesis and my life, and for keeping me motivated.

I owe my deepest thanks to my fiance Hande Coşkun. Thanks to the happiness, courage, patience and friendship it gave me, I was able to complete my thesis by enduring the difficult days of 2020.

Finally, I owe my deepest gratitude to my father Mehmet Muhiddinoğlu, to my mother Gülten Muhiddinoğlu and my sister Merve Şimşek, and all of my family members for their love and their faith in my talents. They are the prominent architects of all my achievements and my career. Without them, I would not have achieved this dissertation like many other things that I have in my life.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF NOTATIONS

$[u\ v\ w]$	Velocity in Body-Fixed Frame
Φ	Roll Angle
Θ	Pitch Angle
Ψ	Yaw Angle
λ	Line of Sight
α	Angle of Attack
β	Sideslip Angle
δ_e	Deflection for Elevator Control Surfaces
δ_r	Deflection for Rudder Control Surfaces
δ_a	Deflection for Aileron Control Surfaces
X	Aerodynamic Force Along x-Body Axis
Y	Aerodynamic Force Along y-Body Axis
Z	Aerodynamic Force Along z-Body Axis
L	Aerodynamic Moment x-Body Axis (Roll Moment)
M	Aerodynamic Moment y-Body Axis (Pitch Moment)
N	Aerodynamic Moment z-Body Axis (Yaw Moment)
p	Roll Rate
q	Pitch Rate
r	Yaw Rate
C_A	Axial Force Coefficient
C_Y	Side Force Coefficient
C_{Y_r}	Side force coefficient derivative with yaw rate
C_N	Normal force coefficient
C_{N_q}	Normal force coefficient derivative with pitch rate

C_{LL}	Rolling Moment coefficient
C_{L_p}	Rolling moment coefficient derivative with roll rate
C_M	Pitching Moment coefficient
C_{M_q}	Pitching moment coefficient derivative with pitch rate
C_{LN}	Yawing Moment coefficient
C_{LN_r}	Yawing moment coefficient derivative with yaw rate
T	Temperature
a	Speed of Sound
P	Atmospheric Pressure
ρ	Air Density
C_{BI}	Direction Cosine Matrix Inertial Frame to Body Frame
$\vec{V}_B^I$	Body Velocity in Inertial Frame
$\vec{V}_W^I$	Wind Velocity in Inertial Frame
$\vec{X}_T^I$	Target Position in Inertial Frame
$\vec{X}_M^I$	Missile Position in Inertial Frame
ϵ_x	Look Angle For Body Lateral Plane
ϵ_y	Look Angle For Body Longitudinal Plane

CHAPTER 1

INTRODUCTION

1.1 Objective of Study

Technological improvements have always been inspired by the nature. Nowadays, most obvious example of this is making sense of swarm behaviour. Swarm behaviour is the mutual action of autonomous and self-organized systems, whether natural or artificial [1]. Compared to a single robotic system, multiple robotic systems provide convenience for performing difficult mission. This makes swarm systems more valuable.

In addition, swarm systems bring some advantages due to decentralized coordination mechanism [2]. These are;

- **Robustness** is the situation that the swarm keeps the capability in case of various defects. Any die out or malfunction of an agent can be tolerated by other swarm agents. Distributed sensing by each agent will improve the swarms' average signal-to-noise ratio.

- **Flexibility** is the ability of generating modularized approaches to various missions. In response to environmental changes, swarm systems should also have the flexibility to suggest alternatives to the work at hand through the use of various coordination strategies.

- **Scalability** is that swarm systems can continue their operations with different numbers of agents. Swarm systems should not be affected by the change in the number of agents.

There are various tradeoffs created by the decentralized structure. For robustness and easy adaptation to the environment, it is necessary to compromise on optimally and predictability. [3]

In this large area of research, the development of swarm system algorithms and the adaptation of technology is an important field of study, especially for flying objects. There are a number of applications, such as large area searches, border patrols, rescue operations and terrain mapping.

This thesis aims to create a modified algorithm; to compare it with existing algorithms and to perform a predefined task with a decision algorithm. The studies are examined under six degrees of freedom nonlinear simulation model. Small-scale low-cost tail controlled munitions are the agents of the swarm system to be modeled in the study.

1.2 Review of Literature

In 1986, Boids computer model [1] is developed by Craig Reynolds. It is related to the motion of bird flocks and fish schools.Model depends on three basic rule of thumbs. These are separation, alignment and cohesion. These rules point out how an individual boid maneuvers depending on positions and speeds according to its neighborhood [4]. Algorithms which are similar to the three rules that Reynolds bases on model flocking behavior are still preferred today. Helbing and Molnar investigated pedestrian dynamics [5]. In this study, social forces is defined to control motion of pedestrians. Three terms are described as acceleration towards the optimal velocity, cohesion and collision. Combination of these three terms converge nonlinear Langevin equations.

Similar to Helbing and Molnar, Tan also carried out a study with Langevin Equations that contained rules for the swarm system. Significant contribution of this study is that non-smooth potential energy equation is used because of local rules which are in communication between neighborhood and necessity of the local calculation on potential gradient. In other words, the local potential should not change when the communication distance limits are not satisfied [6]. The potential function is used to

[1] bird-oid object, bird-like object

obtain kinematic equation (1.1).

$$X = \begin{cases} \dot{x}_i(t) = v_i(t) \\ \dot{v}_i(t) = \frac{1}{m}\nabla U_i(t) \end{cases} \tag{1.1}$$

The potential equation (Langevin Equation) which is considered in the study is shown below shortly (1.2).

$$\ddot{x} = -\nabla U(x) - \gamma\dot{x} + \sqrt{2\gamma c_B T R(t)} \tag{1.2}$$

$U(x)$ is potential function, γ means the damping coefficient, c_B represents Boltzmann's constant, T denotes the temperature and $R(t)$ is a Brownian Motion.

Son, Ahn and Cha made use of Lennard-Jones Potential in order to defines the rules of swarm systems model [7]. Lennard-Jones potential(LJP) is generally preferred in order to model the motion of atoms or molecules [8]. According to the distance which is between two particles, repulsive or cohesive force is obtained from the potential function. This defines swarm systems algorithm. LJP function is mostly used between two swarm system agents. In case of the obstacle avoidance, another potential function is defined because cohesive force is not a desired factor for the obstacle. Therefore, barrier function is commonly preferred [7].

Beni and Wang stated that if thing is improbable and predictable, it is a non-intelligent machine. Also, if there is a probable and unpredictable thing, it is called Brownian Motion. However, swarm intelligence is both improbable and unpredictable [9].

Cheng, Page and Olsen carried on a study about the swarm algorithm which is based on rules and dynamic mission control [10]. In that study, agent based model is preferred. Each agent is able to communicate with others which are within its neighborhood. This gives the environmental awareness to the agents. Therefore, each swarm agent calculates the acceleration command in order to fulfill its duty and to get right position according to the swarm. This acceleration command is obtained by the swarm algorithm rules. In Cheng, Page and Olsen's study [10], six rules ensure that the agents are in a spread and loose form. These rules are cohesion, alignment, separation, collision avoidance, target avoidance and goal seek. These rules are pre-

sented and explained in Chapter 2. In addition, importance of task algorithm constant is evaluated by simulation results.

Another study is about comparison between centralized and decentralized systems [3]. Cheng, Page and Olsen point out the mission capability difference between centralized and decentralized systems. In order to show difference, a mission is defined. The mission is that certain targets being detected and destroyed by an Unmanned Air Vehicle(UAV). Task completion time is selected as the comparison metric. This study shows that as the number of UAV increases, decentralized systems become more important than the centralized system.

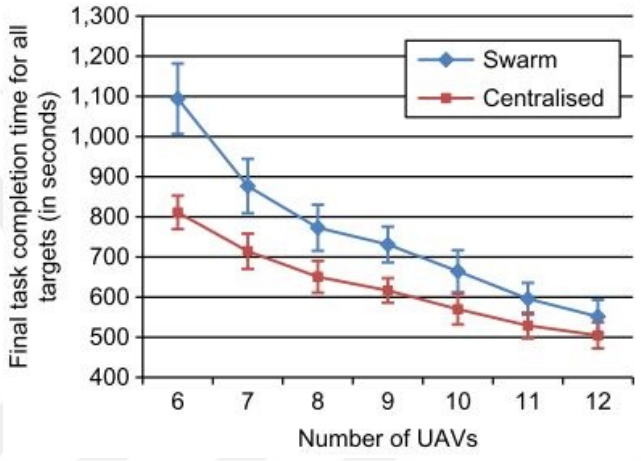

Figure 1.1: Comparison of Centralized and Decentralized Systems [11]

In Giles's study [11], a framework for a swarm UAV systems is given and a doctrine is defined. Unlike other studies, a new structure is introduced as orchestral network. It is based on temporary leader selection. The selection is related to the location of the agent, its state and the mission. This structure provides robustness but restricts scalability. It also brings necessity of an extra process power for each agent. Swarm mission, swarm tactic, swarm play, swarm algorithm and swarm data are defined to describe the swarm mission architecture.

- **Swarm mission** defines the overall goal for the all agents of a swarm.
- **Swarm tactic** defines the duties and responsibilities in order to perform a certain mission.
- **Swarm play** defines behavior of agents with specific triggers and constraints.

- **Swarm algorithm** defines how the agent should take action according to its behavior.
- **Swarm data** defines the data of agents.

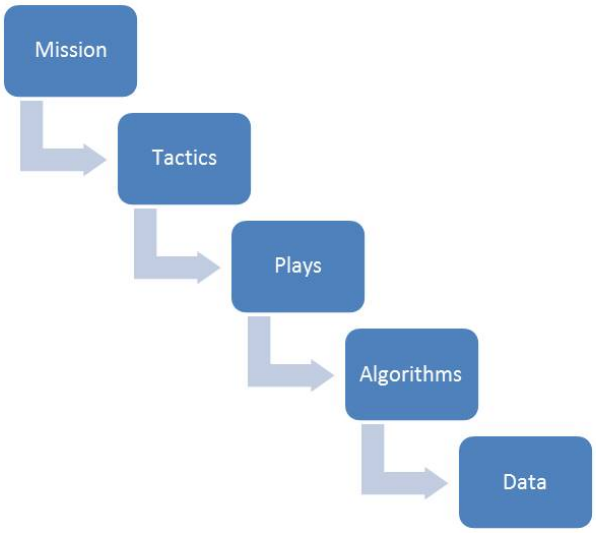

Figure 1.2: Swarm Mission Taxonomy

In an another study [12], self-organized, object avoiding, target tracking and collective motion are examined with ten quadrotors for outdoors. Decentralized structure is preferred with GPS dependency. Mk Basicset L4-Me platform and processes board which contains gyroscope, magnotometer, pressure sensor, GPS recevier, wireless communication unit (2.4GHz XBee) and mini-computer (Gumstix Overo Water) are used to build a swarm system. In the study, information is presented that could increase the loyalty level and inspire other simulation studies. The information is shared in the following chapter while simulation details is given.

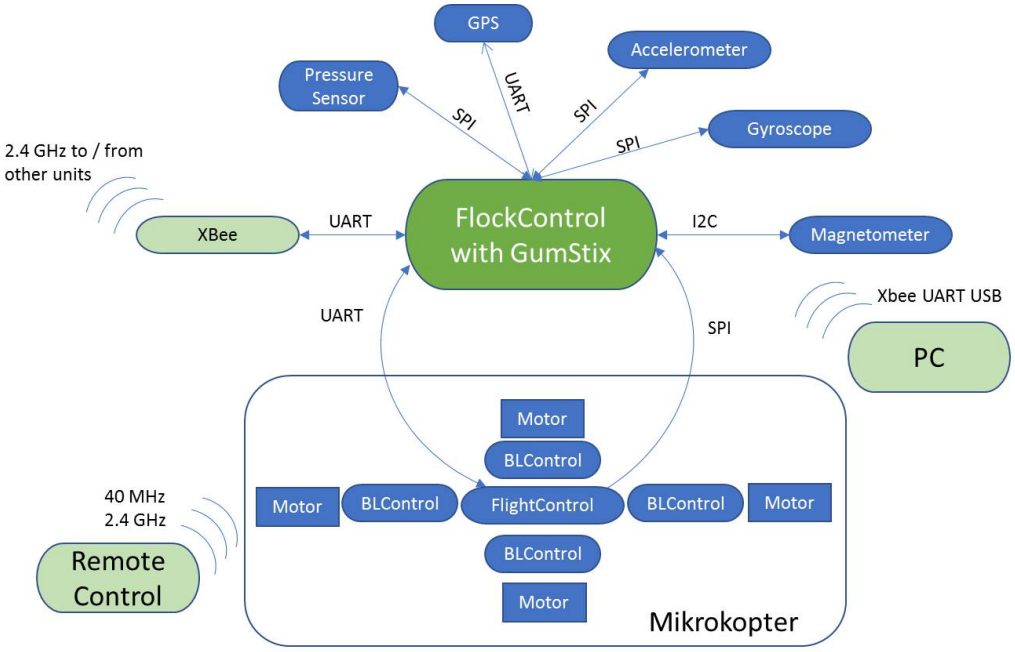

Figure 1.3: Main structure for one unit [12]

In addition, decentralized control algorithm was defined as rules and functions. Simulation frame was prepared to consider of that algorithm, position and velocity measurements errors as well as low level GPS update frequency [13]. Communication delays were negligible for swarm systems because smooth functions was preferred rather than sharp ones [12] [13]. Also in the study [12], the coherence of the flock is measured using the equation (1.3). In the equation, N is number of agents, superscript i and j mean i^{th} agent and j^{th} agent and $\vec{v}$ means velocity vector of swarm agent.

$$\Phi = \frac{2}{N(N-1)} \sum_{i<j} \frac{\vec{v}^i \vec{v}^j}{|\vec{v}^i||\vec{v}^j|} \tag{1.3}$$

In Virágh, Nagy, Gershenson, and Vásárhelyi's study [14], UAV traffic simulation cases are studied for 2D and 3D space under some disturbances such as sensor noise, communication errors and update rate of the data. Two different self-organized algorithms are implemented. These are constant direction and constant velocity algorithms to go to the desired point. Also evolutionary optimization method is used on them under different number of vehicle for traffic cases. In each optimization step, the

aim is to avoid from collision risk and maximize the flux. Collision risk is evaluated as in equation (1.4). Where θ is the Heaviside step function.

$$\psi^{coll}(t) = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j \neq i} \theta(r_c - |x_i(t) - x_j(t)|) \tag{1.4}$$

According to the study [14], the number of UAVs affects the algorithm and the optimization result. In fact, it is the most serious destabilizing environmental phenomenon that has a fundamental impact on efficiency and must be taken into consideration when developing algorithms to be used in real life is communication latency. To compare the effectiveness of the models, flux function given in equation (1.5) is used.

$$\Phi = \frac{v^{eff}}{MFP} \tag{1.5}$$

In equation (1.6), MFP is average linear distance between N distributed agents in Dim dimensional area with linear size L.

$$MFP = L/N^{\frac{1}{Dim}} \tag{1.6}$$

Furthermore, angular order, entropy, average angular velocity, average forward velocity and success rate are used for metrics of flocking behavior [15]. This metrics is detailed in the following chapter.

1.3 Contributions and Novelties

This thesis has the following contributions:

- A swarm of munitions is investigated using a nonlinear 6 DoF simulation
- The performance of a swarm for airdrop munitions is evaluated
- Rule based algorithms and potential function based algorithms in the context of swarm munitions robotics are presented and compared
- A target selection algorithm is implemented

1.4 The Outline of the Thesis

In the second part, mission taxonomy and algorithms from the literature are explained. In the third part, design and subsystem models specific information about modeling simulation are given. In the fourth part, the preferred control theory for the designed system is presented. Weighting matrices are optimized for the algorithms designed in the fifth part. In the last part, two different algorithms are tested and the results are evaluated.

CHAPTER 2

MISSION TAXONOMY AND ALGORITHMS

2.1 Swarm Mission Taxonomy

In this section, the taxonomy of the swarm mission and how it is simulated is determined.

In order to defend a friendly area under an assault, mission of a swarm munitions is investigated. These munitions are released from a UAV in the simulation. In particular the phase after they are released from the UAV is examined. Munitions fly as a flock from the point where the ammunitions are released from UAV to the prediction point where the asymmetric targets are located. When the targets are determined, the values of the targets and the probability of hitting the targets are calculated. Targets are shared based on the calculated probabilities and attack play is initiated. Algorithms are run with the data collected during all these actions. These algorithms are swarm algorithms, decision making algorithms and guidance algorithms.

2.2 Swarm Algorithms

Swarm algorithm design is the most essential part of this thesis. In this part, the algorithm which is used is explained. Two different algorithms are taken into account. These are based on rules and potential functions. Both algorithms aim to do similar basic maneuver and calculate the acceleration to drive the system to which they belong.

2.2.1 Modified Cheng's Rule Based Potential Model

In this section, five main rules which are based on Cheng's model are introduced and explained [10]. After working with the Cheng model, it became clear that additional terms that help the system to damp itself are needed. Due to this modification proposed in this study, it is called "modified" swarm algorithm model. Also, the speed for the kinematic model is limited, unlike the Cheng's model. It is used to prevent possible harmonic motion after reaching the target point.

2.2.1.1 Cohesion Rule

In the cohesion rule, the goal is the desire to go to the center of swarm agents which in the neighborhood of the agent. Cohesion acceleration command is calculated by using equation (2.1).

$$a_1^i = \left(\frac{\sum_{j \in NB_i} P_j}{N_i} - P_i \right) - k \cdot \frac{d}{dt} \left(\frac{\sum_{j \in NB_i} P_j}{N_i} - P_i \right) \tag{2.1}$$

P_i means the position of the agent for which the acceleration command is calculated. P_j is the position of other agents. N_i is the number of agent in the neighborhood. Note that the second term in Equation (2.1) is the modification proposed in this study. Thus, a proportional plus derivative type of acceleration feedback command is used. A neighborhood distance is limited by the communication distance of the munition. For this reason, an agent must be within this distance to be considered a neighbor. Throughout this section, where the rules are examined, a neighborhood distance of 5 meters is chosen. k is the coefficient of damping term. The effect of the damping coefficient on the change in distance is shown in Figure 2.1. To show the effect of damping a simulation is carried out. Figure 2.1 shows the results of the commanded accelerations with different damping coefficients. It may be observed from the simulation that as the damping coefficient is increased, the osscilations in the acceleration command damps out much faster.

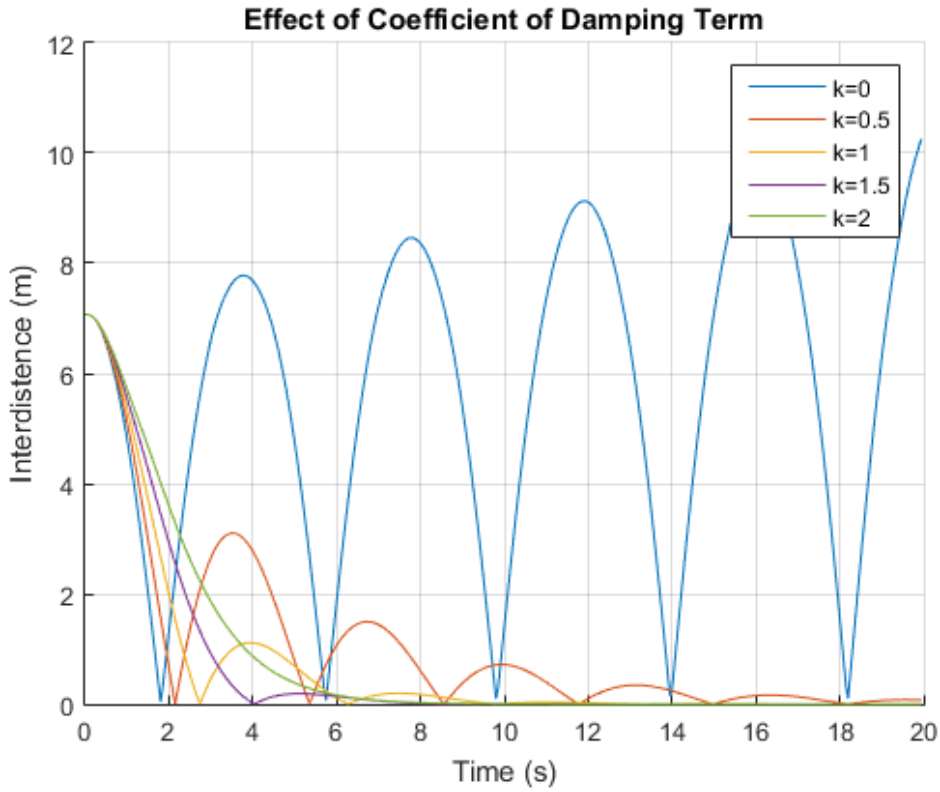

Figure 2.1: Effect of Coefficient of Damping Term

In Figures 2.2, 2.3, 2.4 and 2.5, solid arrows represent the velocity and dotted arrows symbolize the acceleration. In this simulation the value of damping term, k, is equal to 0.7. The rule is tested with the help of a kinematic model. In the kinematic model where the particles do not have initial velocity and initial acceleration, the states of the particles at certain times are shown in the figures. In the simulation, the first agent is not moved. Accelerations for the other agents are calculated. It may be observed from the simulation that the rule generates acceleration commands to the agents to move towards the center. However, after a while, agents at the corners are also included in the acceleration calculations as the enter into the 5 meters detection neighborhood, and the location of the center as well as direction of the acceleration command changes. From the Figure 2.5 it may be observed that agents generate acceleration in order to return after passing the target center position.

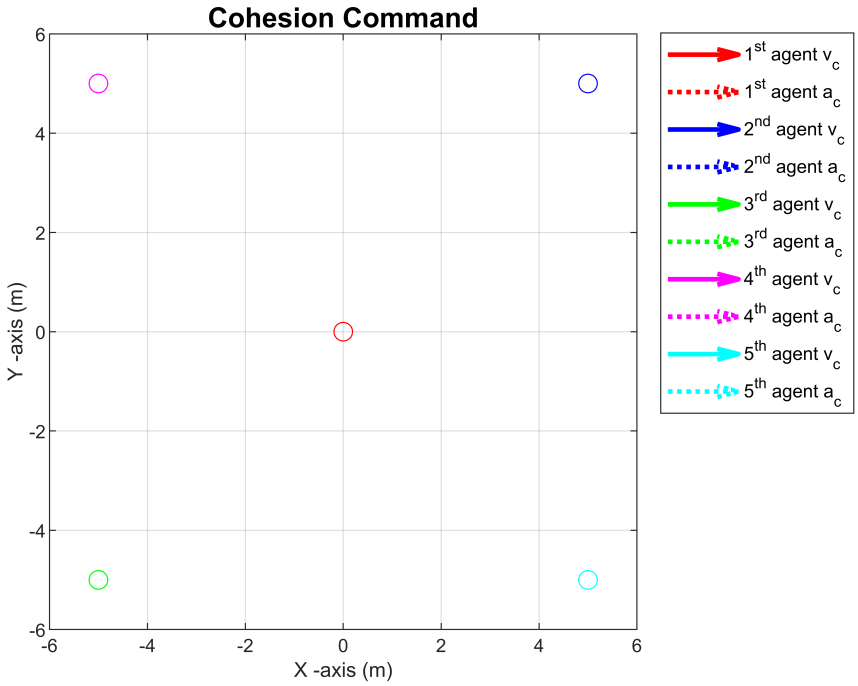

Figure 2.2: Position, velocity and acceleration at initial

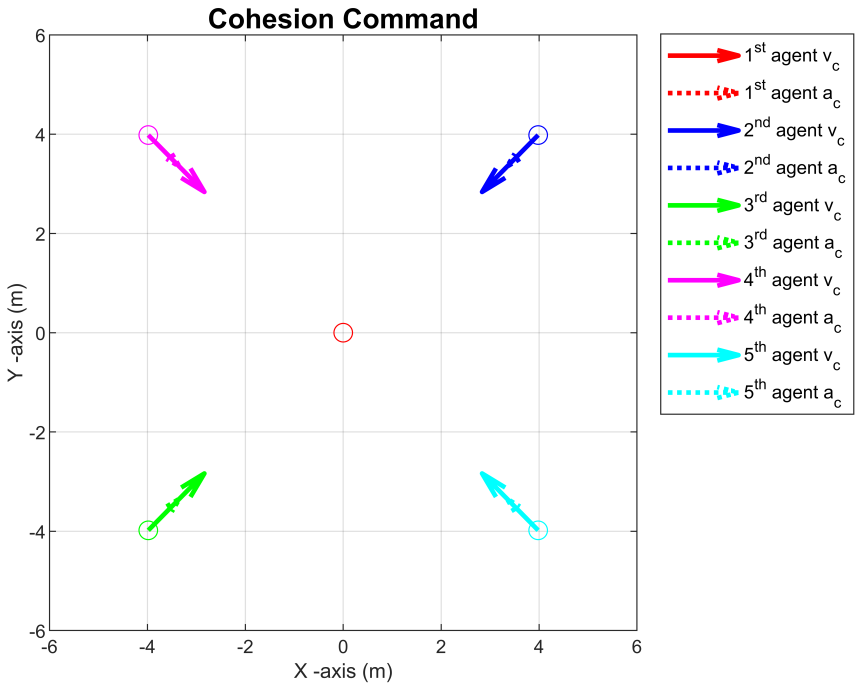

Figure 2.3: Position, velocity and acceleration at 1.8 seconds

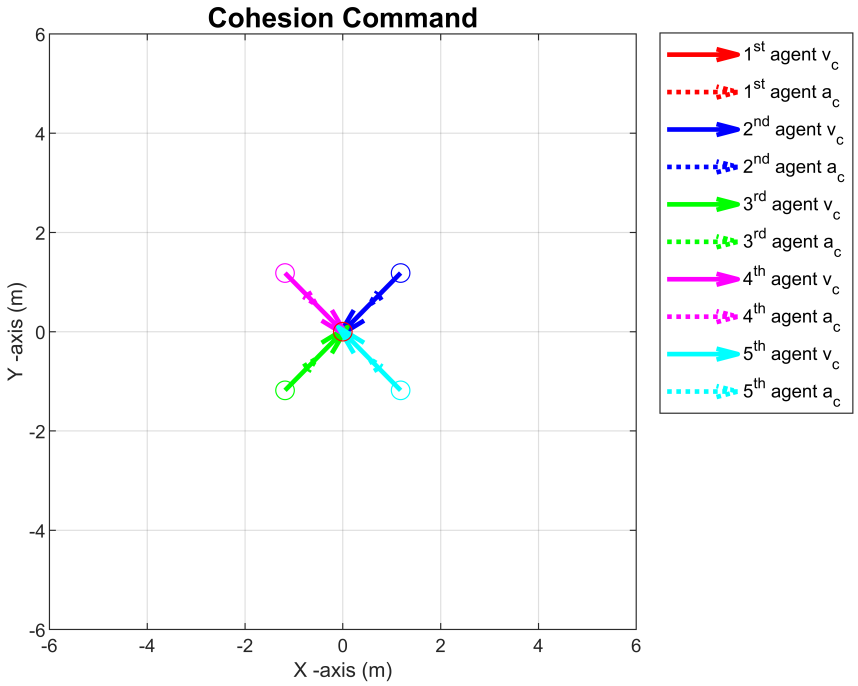

Figure 2.4: Position, velocity and acceleration at 3.8 seconds

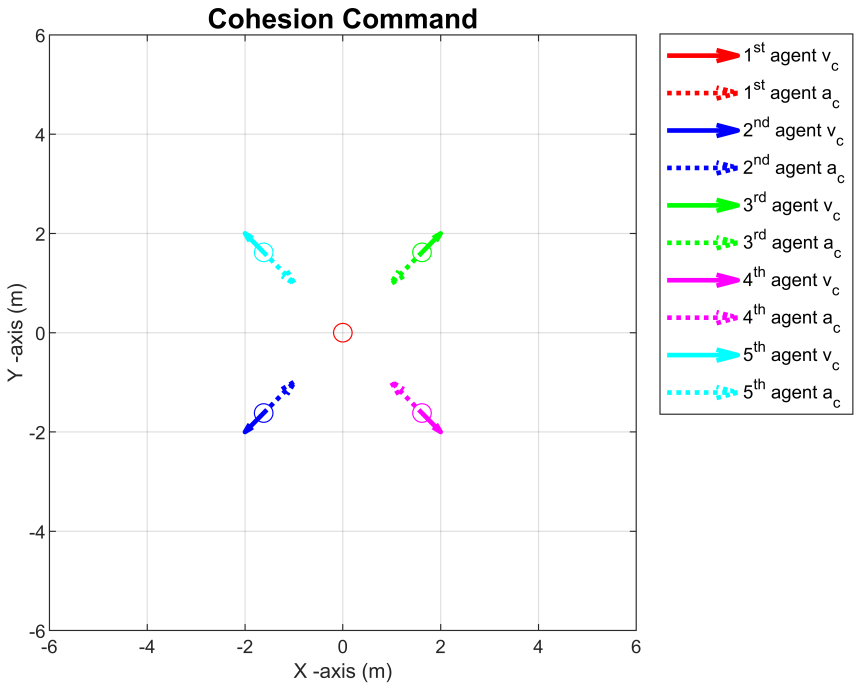

Figure 2.5: Position, velocity and acceleration at 6.2 seconds

2.2.1.2 Collision Avoidance Rule

It is desired to avoid collision with the other agents in the neighborhood of the agent which belongs to the swarm. Collision avoidance acceleration command is calculated by using equation (2.2) [10].

$$a_2^i = \begin{cases} \sum_{j \in NB_i} (P_j - P_i) \cdot \dfrac{1}{\left(\dfrac{\mid P_j - P_i \mid}{d_c}\right)^2}, & \mid P_j - P_i \mid \leq d_c \\ 0, & \mid P_j - P_i \mid > d_c \end{cases} \tag{2.2}$$

Where d_c is the collision distance.

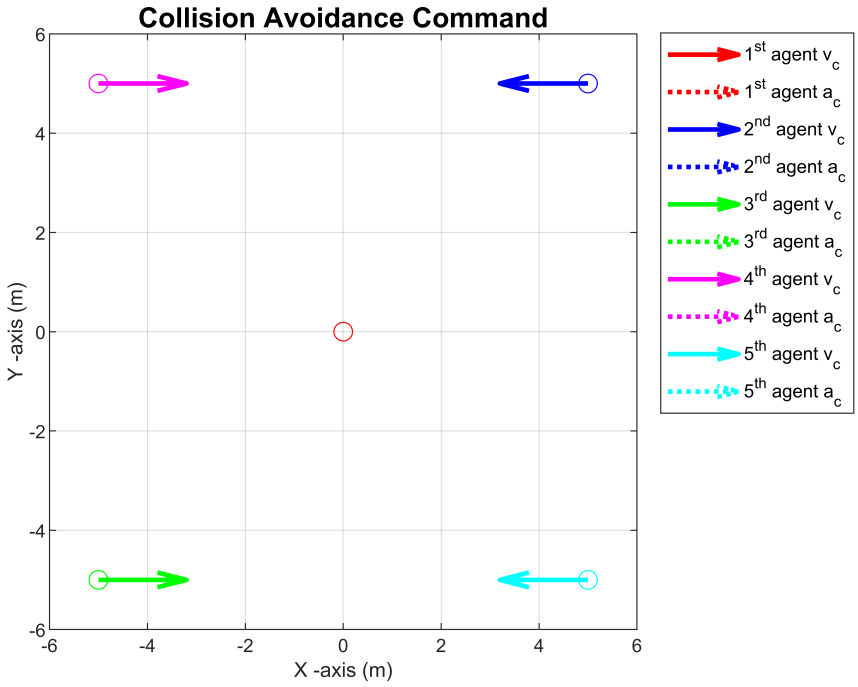

Figure 2.6: Position, velocity and acceleration at initial

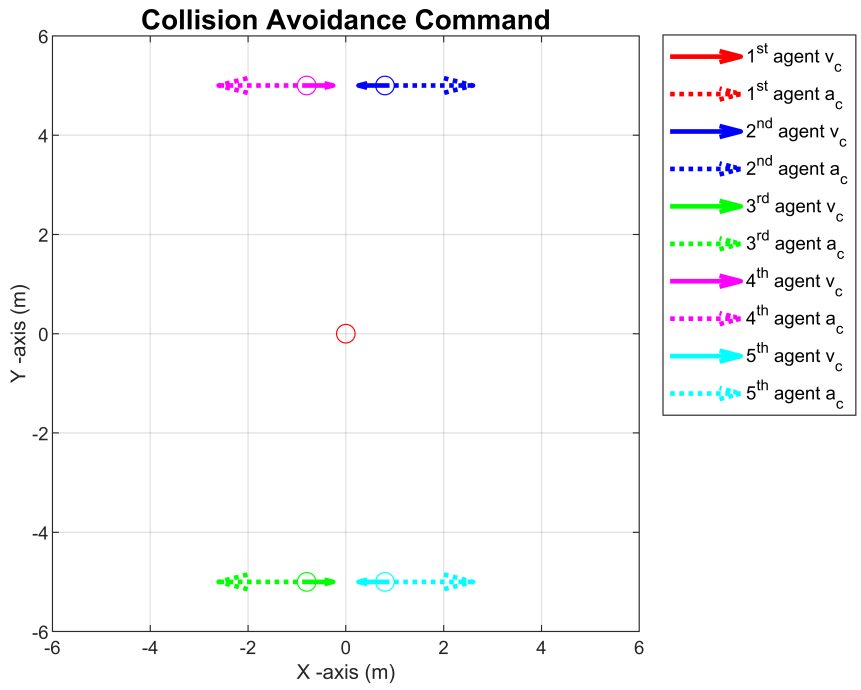

Figure 2.7: Position, velocity and acceleration at 3 seconds

Figure 2.6 and 2.7 are obtained with collision distance, d_c, equal to 3 meters. As the agents approach to each other, they repel each other and the acceleration occurs in the opposite direction with velocity.

2.2.1.3 Alignment Rule

If the swarm agent desires to align its speed vector with other agents that are in neighborhood, alignment acceleration command is generated by using the following equation (2.3).

$$a_3^i = \sum_{j \in NB_i} \frac{V_j}{\|P_i - P_j\|} \tag{2.3}$$

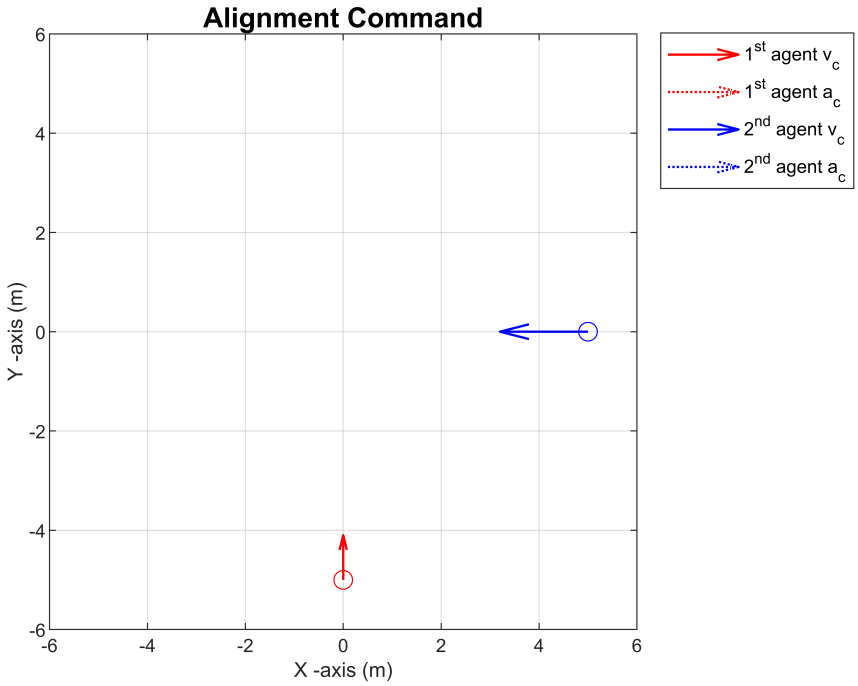

Figure 2.8: Position, velocity and acceleration at initial

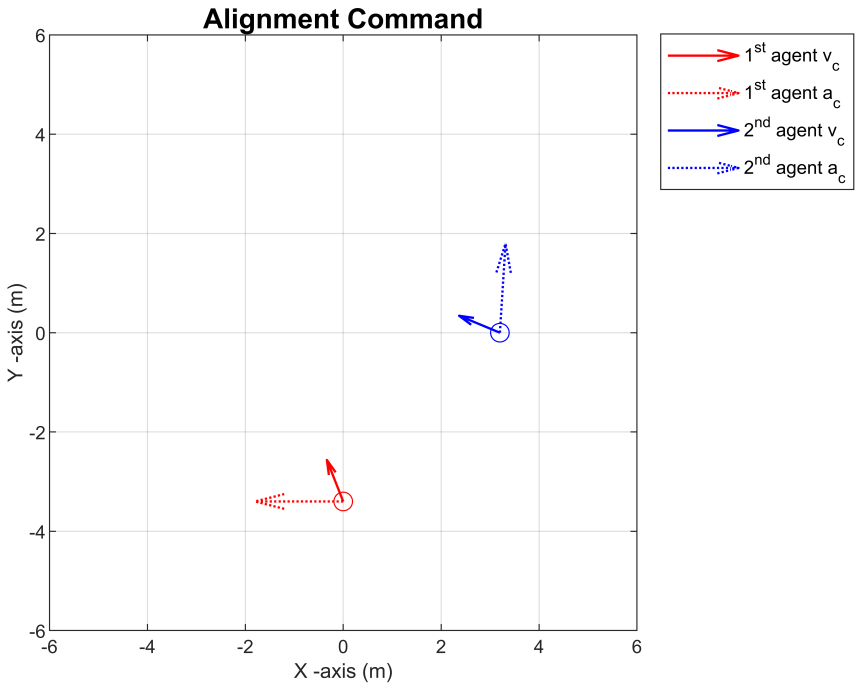

Figure 2.9: Position, velocity and acceleration at 1.6 seconds

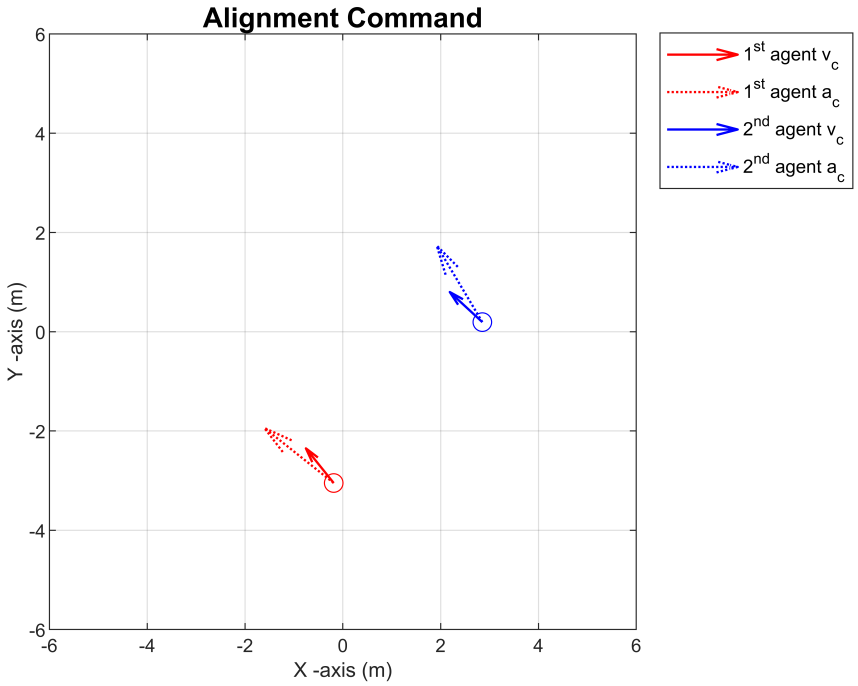

Figure 2.10: Position, velocity and acceleration at 2 seconds

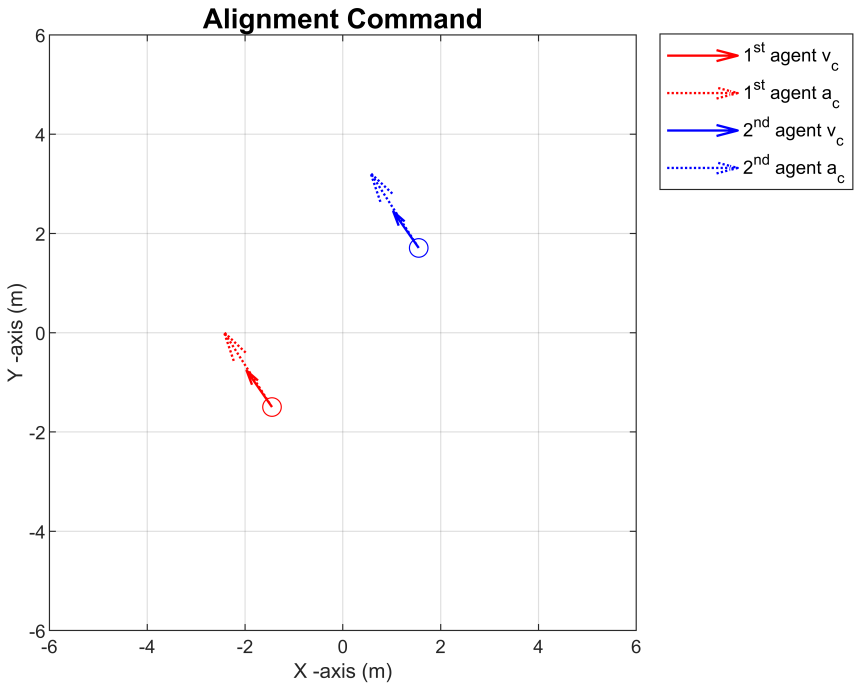

Figure 2.11: Position, velocity and acceleration at 4 seconds

Only when the alignment rule is active, velocity and acceleration of the agents are obtained as figures 2.8, 2.9 2.10 and 2.11.

2.2.1.4 Separation Rule

It is the desired that the swarm agent stays away from other agents that are in the neighborhood. Repulsive force is inversely correlated to the distance between two agents. Separation acceleration command is calculated using the following equation (2.4)[10].

$$a_4^i = \begin{cases} \sum_{j \in NB_i} (P_j - P_i) \cdot \left(1 - \dfrac{P_j - P_i}{d_s}\right), & \mid P_j - P_i \mid \leq d_s \\ 0, & \mid P_j - P_i \mid > d_s \end{cases} \tag{2.4}$$

Where d_s is the separation distance.

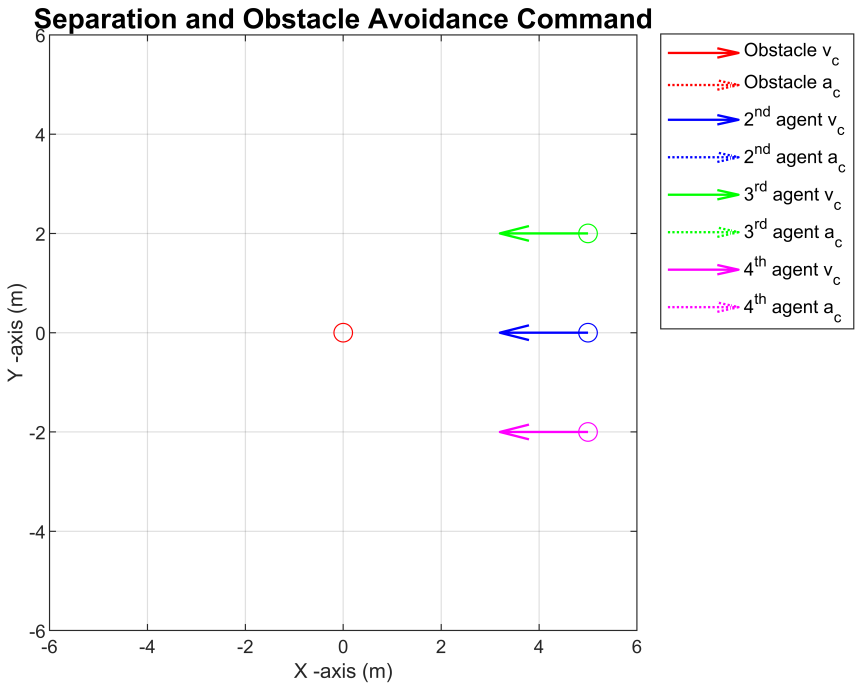

Figure 2.12: Initial position, velocity and acceleration of the agents

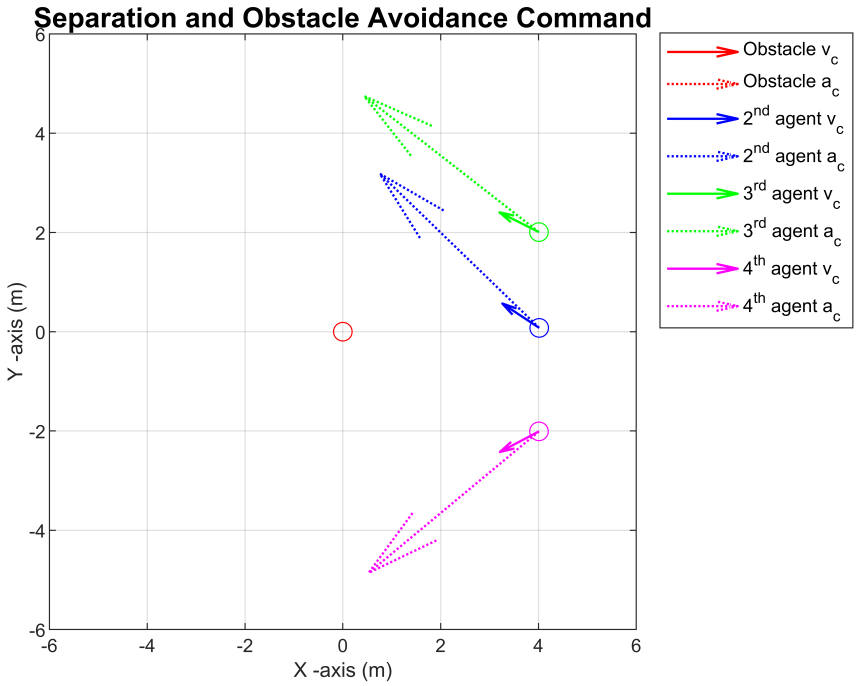

Figure 2.13: Position, velocity and acceleration at 0.8 seconds

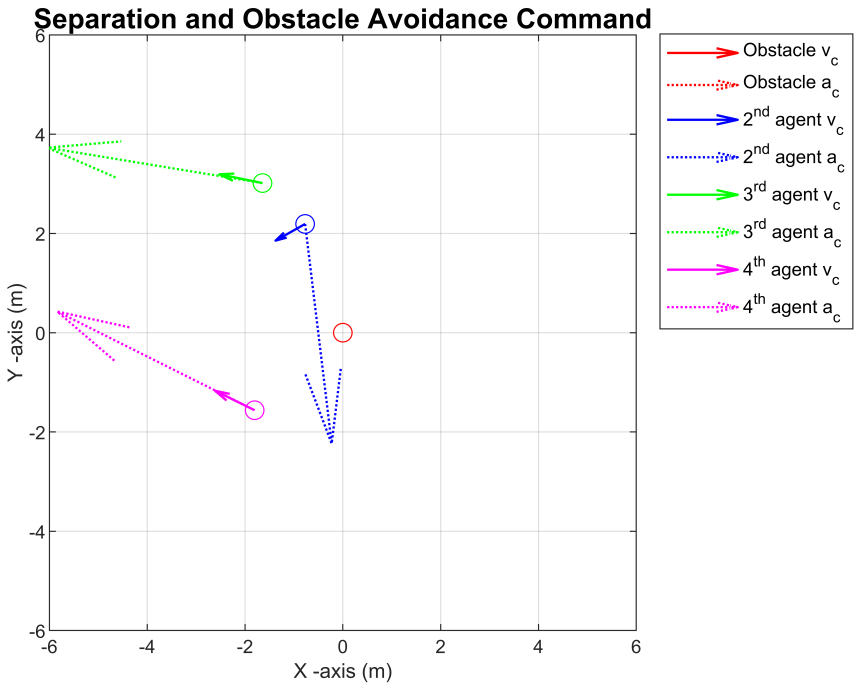

Figure 2.14: Position, velocity and acceleration at 6.8 seconds

A scenario is tested in which three agents move around an obstacle. The simulation results are presented in 2.12, 2.13 and 2.14. The figures show that the agents avoid from the obstacle as well as keep their separation.

2.2.1.5 Obstacle Avoidance Rule

The desire to avoid obstacles around the swarm agent. Obstacle avoidance acceleration command is calculated using following equation (2.5)[10].

$$a_5^i = \begin{cases} \sum_{j \in NB_o} (P_o - P_i) \cdot \left(1 - \dfrac{P_o - P_i}{d_o}\right), & \mid P_o - P_i \mid \leq d_o \\ 0, & \mid P_o - P_i \mid > d_o \end{cases} \tag{2.5}$$

Where d_o is obstacle avoidance distance and P_o is the obstacle position.

2.2.2 Lennard-Jones Potential Rule Based Model

LJP is generally preferred as the basic approach for interaction between atoms or molecules. This method was proposed in 1924 by John Lennard-Jones [16]. The main purpose of the potential function is to generate the attractive and repulsive force by following equation.

$$V(r) = 4\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right] \tag{2.6}$$

In the equation (2.6) ϵ is energy-scale parameter and σ is a basic length-scale parameter. Minimum potential is equal to -ϵ at $2^{1/6}$ σ [8].

$$r_{min} = 2^{1/6}\sigma \tag{2.7}$$

2.2.2.1 Cohesion and Separation Rule for Center of Swarm

This rule forces the agent to keep an optimum distance from center of agents which are in the neighborhood. If distance is greater than the optimum value, acceleration command is generated to close the distance. Conversely, if the distance is lower than it should be, the acceleration command is generated to move away. Equation

(2.9) governs the calculation of acceleration commands which is used for providing optimum distance to the center of swarm.

$$\vec{r}_{i,1} = \left(\frac{\sum_{j \in NB_i} \vec{P}_j}{N_i} - \vec{P}_i \right) \tag{2.8}$$

$$\vec{n_1^i} = \left(\frac{\vec{r}_{i,1}}{\|\vec{r}_{i,1}\|} \right) \tag{2.9}$$

$$a_1^i = \frac{d\left(V\left(\|\vec{r}_{i,1}\|\right)\right)}{dx} \vec{n_1^i} \tag{2.10}$$

If the distance where the attractive and repulsive force is zero is selected as 2 meters, the following equation (2.11). σ is obtained by using Equation (2.7). Where ϵ is chosen as 5.

$$V(r) = 20 \left[\left(\frac{1.7817}{r} \right)^{12} - \left(\frac{1.7817}{r} \right)^{6} \right] \tag{2.11}$$

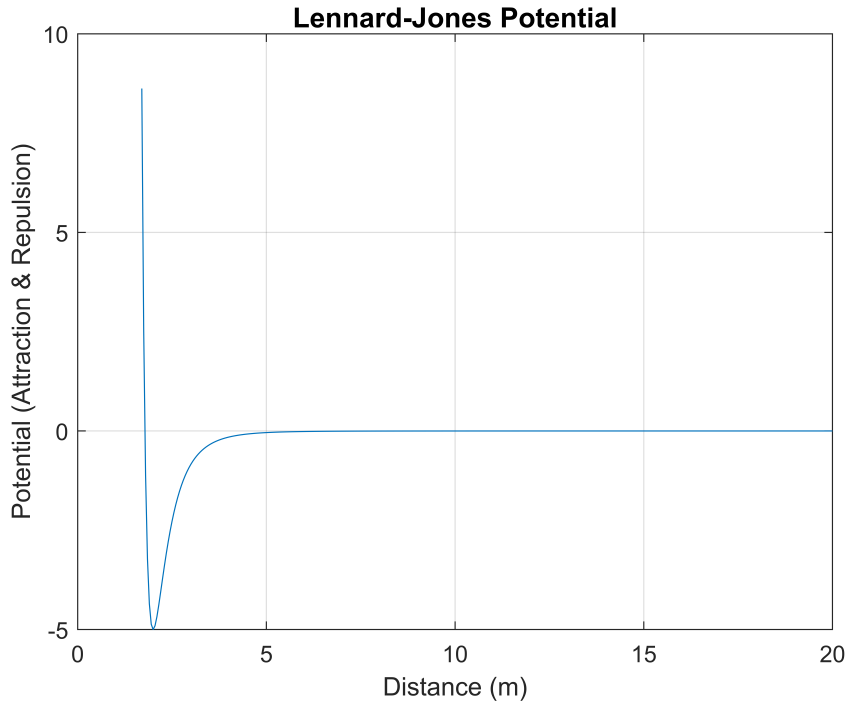

Figure 2.15: Lennard-Jones Potential

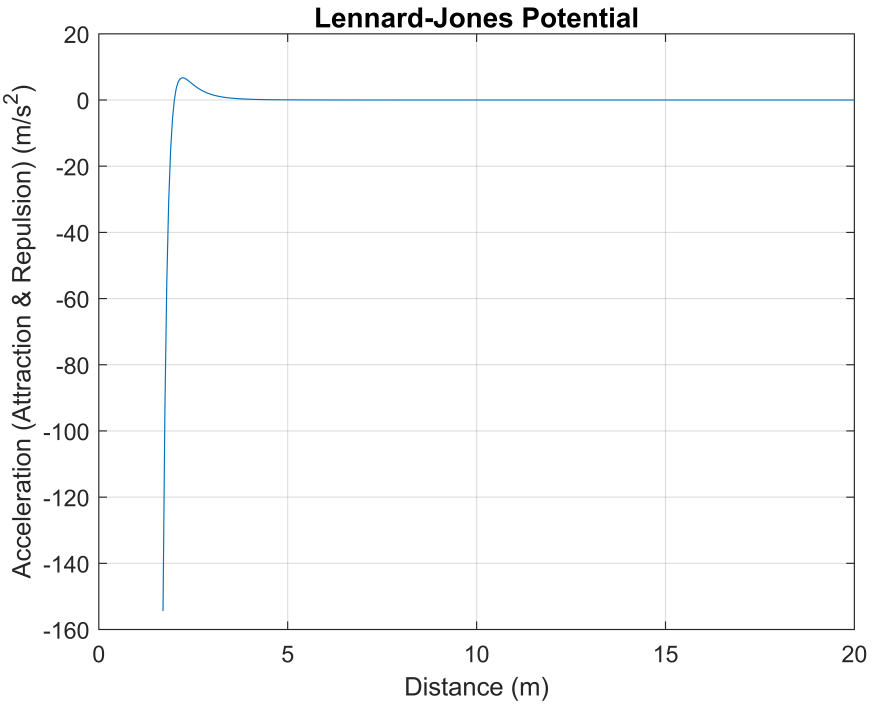

Figure 2.16: Derivative of Lennard-Jones Potential

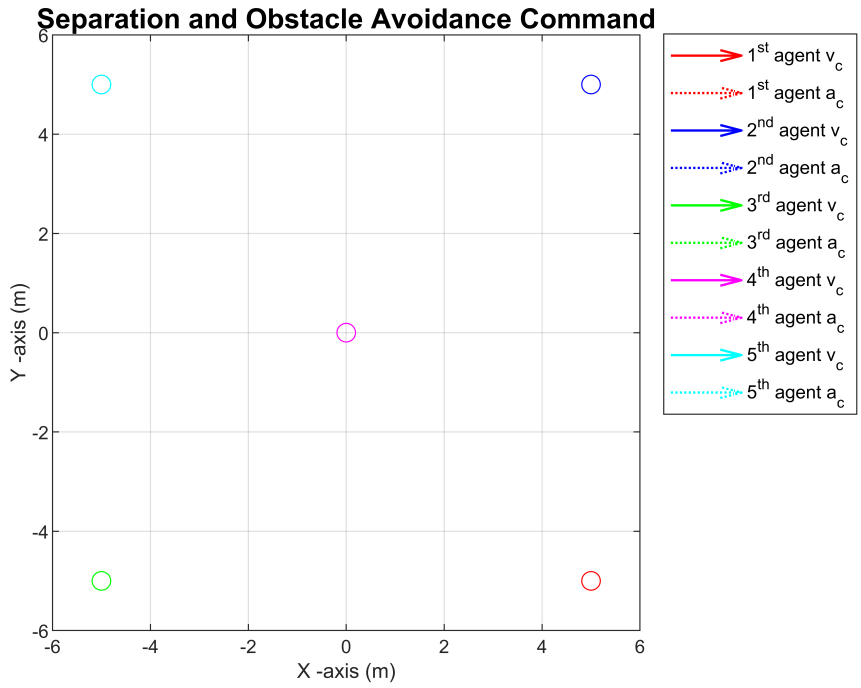

Figure 2.17: LJP: Cohesion and Separation Rule for Center of Swarm at Initial

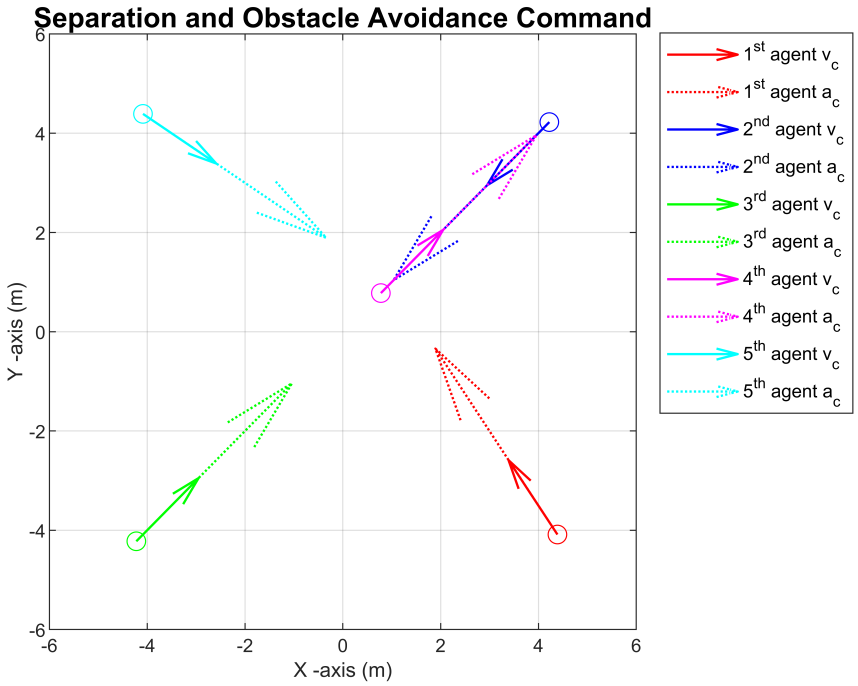

Figure 2.18: LJP: Cohesion and Separation Rule for Center of Swarm at 0.8 seconds

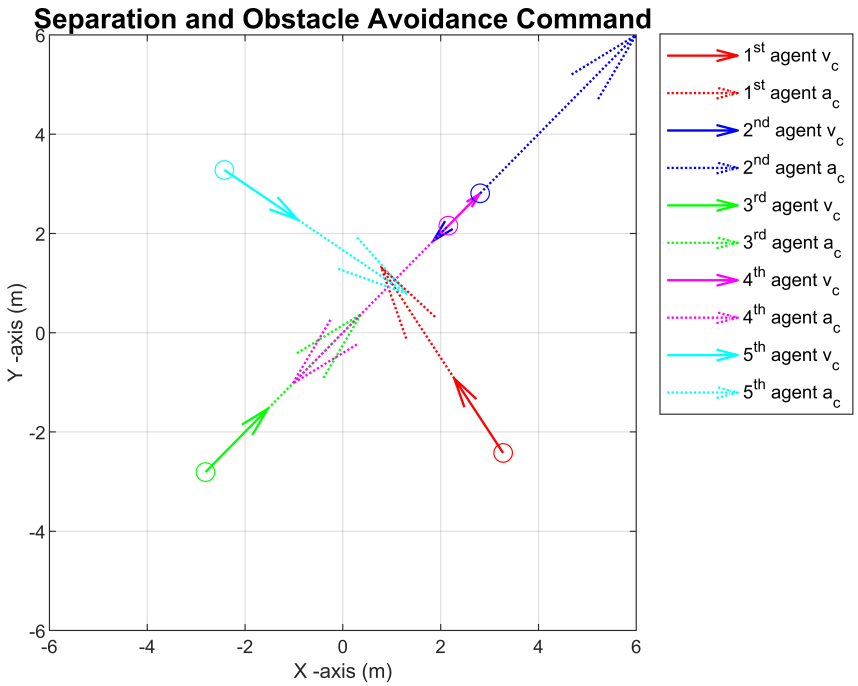

Figure 2.19: LJP: Cohesion and Separation Rule for Center of Swarm at 1.8 seconds

Figure 2.17, 2.18 and 2.19 presents the simulation of LJP. It is seen in figures where the particles approach each other and move away.

2.2.2.2 Cohesion and Separation Rule for Agents of Swarm

This rule keeps the distance of an agent from other agents. If distance is greater than the optimum value, acceleration command is generated to close the distance. Conversely, if the distance is lower than it should be, the acceleration command is generated to move away. Same scenario which is mentioned in section 2.2.2.1 is valid for this part. The difference of this rule is that it uses a different coefficient for the potential function.

$$\vec{r}_{i,2} = \sum_{j \in NB_i} \left(\vec{P}_j - \vec{P}_i \right) \tag{2.12}$$

$$\vec{n_2^i} = \left(\frac{\vec{r}_{i,2}}{\|\vec{r}_{i,2}\|} \right) \tag{2.13}$$

$$a_2^i = \frac{d\left(V\left(\|\vec{r}_{i,2}\|\right)\right)}{dx} \vec{n_2^i} \tag{2.14}$$

2.3 Decision Algorithm

The decision algorithm is the logic that changes the phases of the flight. It uses various algorithms to make the changes. Actions such as probability calculation, enemy sharing and assault order are made with the decision algorithm.

Algorithm 1 Overall Decision Algorithms

```
initialization;
while Munitions fly do
    if does not lock on the target then
        Instructions;
        Target Information Algorithm;
        Distance Algorithm;
        Probability Algorithm;
        Target Value Algorithm;
    else
        Instructions;
        Target Information Algorithm;
        Locking Information Algorithm;
        Target Value Algorithm;
    end
end
```

2.3.1 Target Information Algorithm

Target Information Algorithm (TIA) is the most basic algorithm among all algorithms. It uses the last condition about targets, such as their position, calculated target value and whether it is locked on by another agent of swarm. In other words, it assists other algorithms.

2.3.2 Distance Algorithm

Distance Algorithm (DA) determines the distance between itself and an enemy element with the help of information fed by TIA . Then, this calculation feeds probability algorithm and weighted matrices of proportional navigation acceleration commands and swarm algorithm acceleration commands.

2.3.3 Probability Algorithm

Probability Algorithm (PA) calculates the probability to hit the target. However, it does not mean hit prediction algorithm. It considers some information such as value of target, distance of target and energy of the munition to go to the enemy. Then, these values are used to obtain a probability number which is between zero to one. Probability number is calculated for every target according to an each swarm agent at equation (2.15). Obtained numbers are compared to a random value that is between 0.7 and 0.8. If the comparison result is that probability number is greater than random number, the agent will lock on the enemy. Finally, the result is informed to TIA and LIA [10].

$$P = \frac{\delta^2}{\delta^2 + \Theta^2} - \beta(1 - e^{-\frac{d}{\gamma}}) \tag{2.15}$$

In equation (2.15), δ means value of target, regulated by the target value algorithm (TVA). Θ is the threshold constant for value of target. It helps reduce order of complexity to lock on enemy. β is the constant weight value. Second term is related to importance of distance which is to go to the enemy. Therefore, γ is the constant threshold term for distance. d is the distance which is between agent and enemy. Distance is fed by DA. Any two agents can calculate the same probability value at the same time against a single target. Against this possibility, the precaution is taken by using equation (2.16) and randomness is created.

$$I = \begin{cases} 1, & P \geq rand[0.7, 0.8] \\ 0, & P < rand[0.7, 0.8] \end{cases} \tag{2.16}$$

Algorithm 2 Probability Algorithms

Result: Lock on enemy!

initialization;

while I=*0* **do**

  while T=T_n **do**

    instructions;

    Equation (2.15)

    Equation (2.16)

    $T = T + 1$

  end

end

2.3.4 Target Value Algorithm

Target Value Algorithm (TVA) calculates the value of target. Initial values are given by users. However, values change by with action of the swarm system with the following equation. Also, this value is reduced when a decision to destroy is taken by any agent[10].

$$\delta(t+1) = \frac{\delta(t) + (\Delta t)\zeta\delta(t)}{m_{action}} \tag{2.17}$$

In (2.17), Δt is approximate time coefficient for CPU time. ζ means percent value of increment and m_{action} is the number of munition that decided to attack the target.

2.3.5 Locking Information Algorithm

Locking Information Algorithm (LIA) is the algorithm used to prevent the swarm agent from changing its target after determining the target due to possible disruptive effects and changes in the valuation algorithm.

2.4 Guidance Algorithm

In this thesis, proportional navigational guidance(PNG) algorithm is preferred. PNG is the most common guidance law [17]. This tries to null the line-of-sight (LOS) direction change rate by acceleration commands. When LOS rate is equal to zero, collision becomes inevitable.

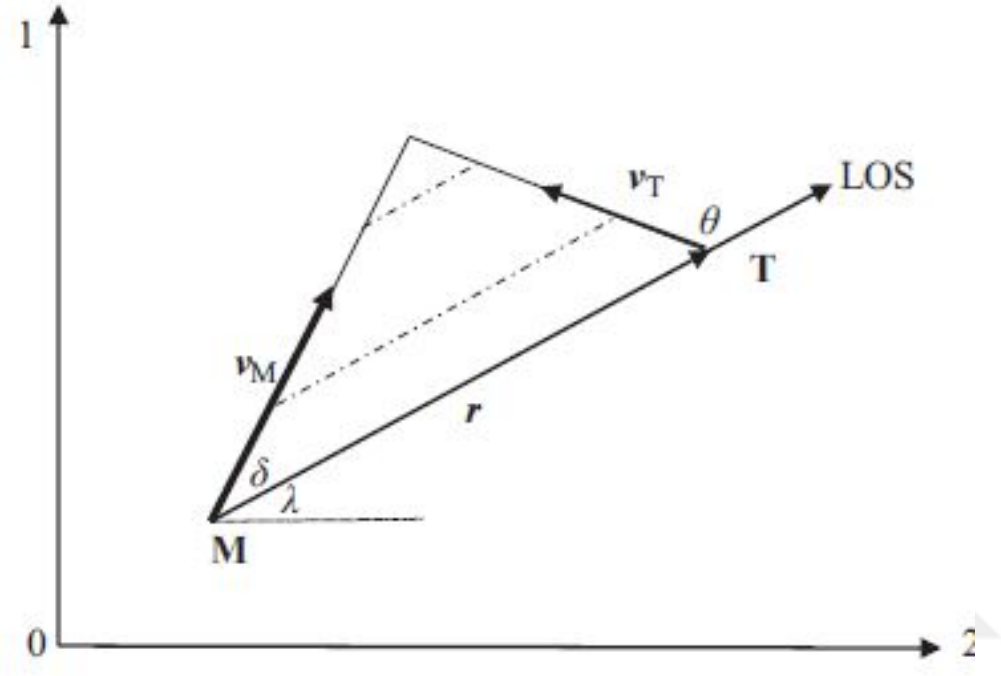

Figure 2.20: Geometry of planar engagement [17]

$$a_c(t) = N v_{cl} \dot{\lambda} \tag{2.18}$$

Where equation (2.18) a_c is acceleration guidance command and λ is LOS angle which is generally defined with respect to inertial axis. In addition, v_{cl} means the closing velocity.

CHAPTER 3

MODELLING AND SIMULATION

In this part, modelling of the munitions is given. Aerodynamic database and geometry of munition is defined. In this study, linear controller design is preferred. Therefore, state variable modelling is necessary condition to develop controller. Thus, linearization of equation of motion and and state variable model is obtained also in this chapter.

3.1 Basic Properties of Munition

It is assumed that guided munition is obtained by attaching a guidance kit to non-guided munition. A 40 mm grenade is used in this thesis. Another assumption is that the guidance kit has tail control actuator system, processor board, Global Position System (GPS), communication module and battery. The whole munition is shown at figure 3.1. The details are listed in Table 3.1. There is no thruster to gain speed or accelerate laterally. They will accelerate with the help of free fall. Maneuvers of the munition are realized by the tail surface. Tail controlled system has three actuators in order to roll, yaw and pitch maneuvers.

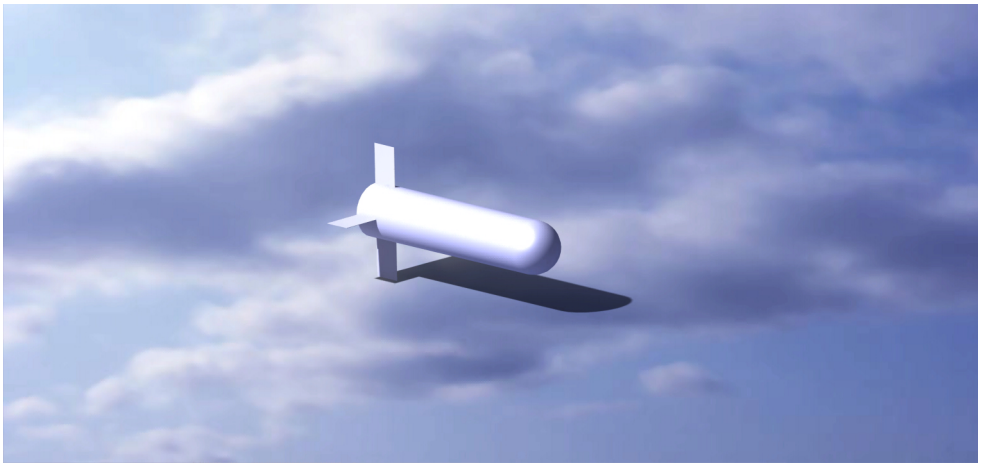

Figure 3.1: Rough view of the munition

Table 3.1: Results

Parameter	**Value**	**Unit**
Diameter	40	mm
Length	180	mm
Mass	0.9	kg
I_{xx}	$1.17e^{-4}$	kg.m^2
I_{yy}	0.001	kg.m^2
I_{zz}	0.001	kg.m^2

There are four fins at the aft of munition. Three actuators are driving them. Two of them provide yaw and pitch movement by moving the opposite fins in the same direction. Last one is used for roll movement by using opposite fins in opposite direction. In this design, any system that can provide propulsion is not considered. Munitions are released at an appropriate speed and altitude. Converting potential energy to kinematic energy is significant for this concept and gravity force is used for this. In the design, it is assumed that munition has a high resolution camera as a seeker. Global position systems and communication systems are described in following parts.

3.2 Aerodynamic Model

The most fundamental component is the aerodynamic model to simulate at the highest level of fidelity. In a high-level system design, computational fluid dynamics(CFD) programs are used for this model data. However, Missile DATCOM is preferred in order to obtain aerodynamic data because it is based on semi-empirical method within conceptual design and it is more agile than CFD. With the help of this tool, aerodynamic coefficients are obtained under certain flight conditions using the basic geometric information of the missile.

Table 3.2: Flight Conditions

Parameter	**Range**	**Unit**
Mach Vector	[0.1 0.2 0.3 0.4 0.5 0.6]	-
β Vector	[-5 0 +5]	Degree
α Vector	[-15 -10 -5 -1 0 +1 +5 +10 +15]	Degree
δ Vector	[-5. -1. 0 1. 5.]	Degree
Altitude Vector	[0 500 1000 1500 2000 2500 3000 3500 4000]	Meter

The flight conditions which are given at Table 3.2 are used to drive DATCOM. As a result of this, aerodynamic coefficients are created according to combinations of flight conditions specified in the table. Aerodynamic force and moment terms in Equation (3.1)-(3.6) are calculated with air density (ρ), reference area (S), chord length (c) and speed of missile relative to air($V_{B/W}$).

$$X = \frac{1}{2} C_x \rho V_{B/W}^2 S \tag{3.1}$$

$$Y = \frac{1}{2} C_y \rho V_{B/W}^2 S \tag{3.2}$$

$$Z = \frac{1}{2} C_z \rho V_{B/W}^2 S \tag{3.3}$$

$$L = \frac{1}{2} C_{M_x} \rho V_{B/W}^2 S c \tag{3.4}$$

$$M = \frac{1}{2} C_{M_y} \rho V_{B/W}^2 Sc \tag{3.5}$$

$$N = \frac{1}{2} C_{M_z} \rho V_{B/W}^2 Sc \tag{3.6}$$

Aerodynamic coefficients which are used in equation (3.1) - (3.6) are obtained from the following equations.

$$C_x = -C_A \tag{3.7}$$

$$C_y = C_Y + C_{Y_r} r \frac{c}{2V_{B/W}} \tag{3.8}$$

$$C_z = -C_N - C_{N_q} q \frac{c}{2V_{B/W}} \tag{3.9}$$

$$C_{M_x} = C_{LL} + C_{L_p} p \frac{c}{2V_{B/W}} \tag{3.10}$$

$$C_{M_y} = C_M + C_{M_q} q \frac{c}{2V_{B/W}} \tag{3.11}$$

$$C_{M_z} = C_{LN} + C_{N_r} r \frac{c}{2V_{B/W}} \tag{3.12}$$

The parameters and definition of the aerodynamic coefficients in the above equation are explained in the table 3.3. The coefficients in the table are extracted from a previously prepared database during the simulation. Instantaneous flight conditions not included in the database are found using linear interpolation.

Table 3.3: Function and Definition of The Aerodynamic Coefficients

Symbol	Definition	Function
C_A	Axial force coefficient (body axis)	$f(M, \alpha, \beta, \delta_e, \delta_r, \delta_a)$
C_Y	Side force coefficient (body axis)	$f(M, \alpha, \beta, \delta_e, \delta_r, \delta_a)$
C_{Y_r}	Side force coefficient derivative with yaw rate	$f(M)$
C_N	Normal force coefficient (body axis)	$f(M, \alpha, \beta, \delta_e, \delta_r, \delta_a)$
C_{N_q}	Normal force coefficient derivative with pitch rate	$f(M)$
C_{LL}	Rolling Moment coefficient (body axis)	$f(M, \alpha, \beta, \delta_e, \delta_r, \delta_a)$
C_{L_p}	Rolling moment coefficient derivative with roll rate	$f(M)$
C_M	Pitching Moment coefficient (body axis)	$f(M, \alpha, \beta, \delta_e, \delta_r, \delta_a)$
C_{M_q}	Pitching moment coefficient derivative with pitch rate	$f(M)$
C_{LN}	Yawing Moment coefficient (body axis)	$f(M, \alpha, \beta, \delta_e, \delta_r, \delta_a)$
C_{LN_r}	Yawing moment coefficient derivative with yaw rate	$f(M)$

3.2.1 Atmosphere Model

COESA Atmosphere Model [18] is preferred in order to compute temperature, speed of sound, atmospheric pressure and air density (3.13).

$$[T, a, P, \rho] = f(h) \tag{3.13}$$

WGS84 gravity model [19] is used in order to calculate gravitational acceleration with latitude, longitude and altitude (3.14).

$$g = f(\mu, l, h) \tag{3.14}$$

3.2.2 Wind Model

In this study, continuous and constant speed wind is used at certain altitude. At equation (3.15) and (3.16), $\vec{V}_w^I$ is wind velocity vector defined in inertial frame. C_{BI} is the

direction cosine matrix inertial frame to body frame.

$$\vec{V}_w^I = \begin{bmatrix} u_w \\ v_w \\ w_w \end{bmatrix}^I \tag{3.15}$$

$$\vec{V}_{B/W}^B = C_{BI}(\vec{V}_B^I - \vec{V}_W^I) \tag{3.16}$$

Also there is an approach to determine the wind speed at different altitude at equation (3.17) [20]. w_n means wind speed at height Z_n and P is exponential parameter which is taken 1/7 [21].

$$\frac{w_1}{w_2} = (\frac{Z_1}{Z_2})^P \tag{3.17}$$

3.3 Whole Simulation Topology

Simulation environment is significant for studies of swarm systems because there are a lot of agents that demand processing load. Therefore, historical trend is an object oriented simulation environment for this topic.This study is not prepared with an object oriented simulation environment, but the necessity of this is seen during the study. However, the number of agents in the simulation environment can be easily scaled and the simulation is able to adapt it. In this simulation, major parts are flight mechanics, swarm and decision algorithms, guidance and control algorithms , GPS model, CAS (Control Actuation System), atmosphere model, communication model and target model. Figure 3.2 and 3.4 are shown to create a visual awareness.

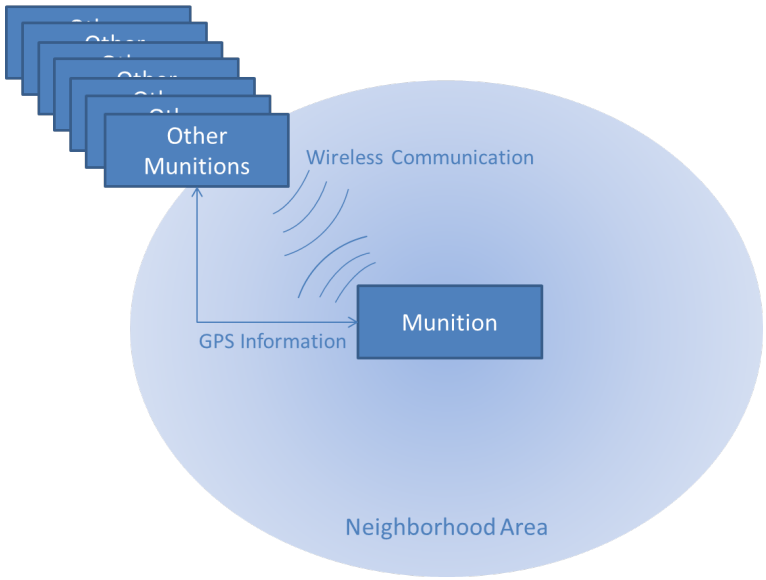

Figure 3.2: Relationship Between Munitions

As shown Figure 3.4, communication model is used jointly but each munition has own specific communication disturbance. During Monte Carlo analysis, they communicate with each other with different frequency. In this way, study is tested for robustness. In addition, GPS model has some disturbance.

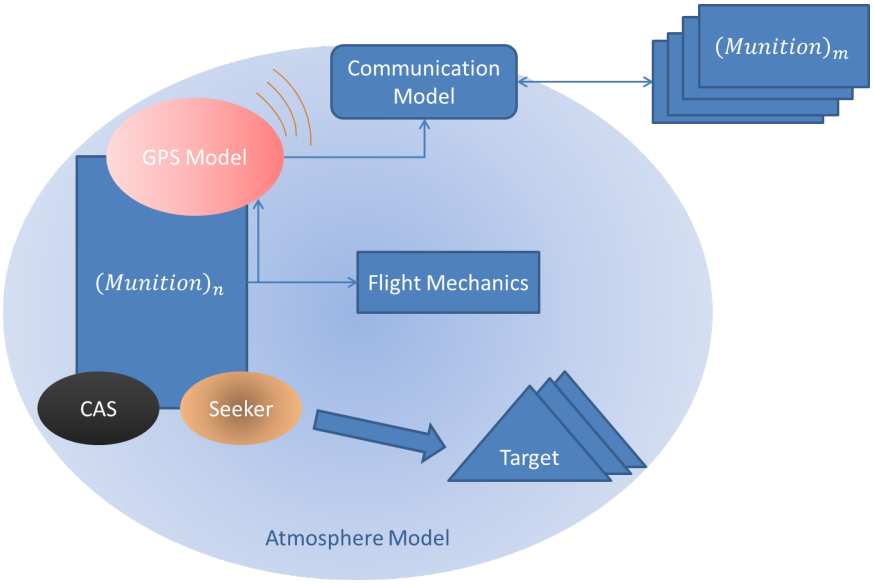

Figure 3.3: The Topology of Simulation

3.3.1 GPS Model

GPS is modelled as to get position according to inertial frame. The frequency of obtaining global position signal is taken as 5 Hz [12].

3.3.2 Communication Model

Communication module is modelled as sending and receiving the packages which contain munition identification, GPS data and engagement information. The communication range is limited to 50 meters. In addition, the communication frequency is taken 10 Hz.

3.3.3 Seeker Model

Seeker model is used to feed guidance model. In this thesis, high level fidelity seeker model is not preferred because of necessity of processing load. Instead, it proceeds with a simpler modeling.

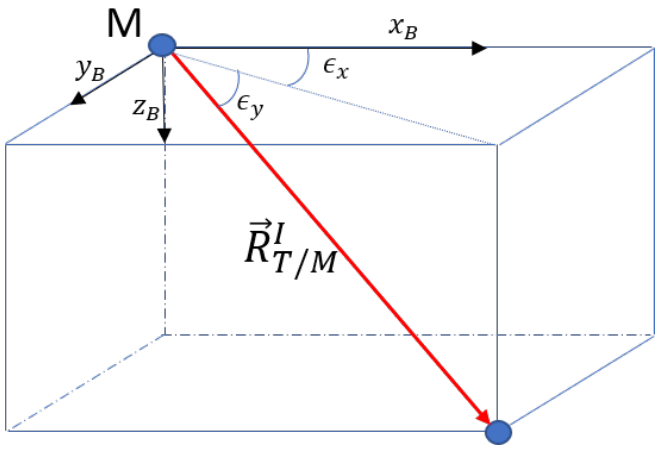

Figure 3.4: Look Angles

$$\vec{R}^I_{T/M} = \vec{X}^I_T - \vec{X}^I_M \tag{3.18}$$

$$\vec{R}^B_{T/M} = C_{BI}\vec{R}^I_{T/M} \tag{3.19}$$

$$\vec{R}^B_{T/M} = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \end{bmatrix} \tag{3.20}$$

$$\epsilon_x = atan2(\epsilon_2, \epsilon_1) \tag{3.21}$$

$$\epsilon_y = atan2(\epsilon_3, \sqrt{\epsilon_1^2 + \epsilon_2^2}) \tag{3.22}$$

ϵ_x and ϵ_y are the angle values at which the munition sees the target from a strapdown seeker.

3.3.4 Control Actuation Model

CAS is modeled as a second order system with 10 Hz natural frequency and 0.7 damping ratio.

$$TF_{CAS} = \frac{\omega_n}{s^2 + 2\zeta\omega_n s + \omega^2} \tag{3.23}$$

3.3.5 Flight Mechanics

6 degree of freedom(DOF) equations of motion are used with the flat and non-rotating earth assumption [22].

$$m(\dot{u} + qw - rv) = -mg\sin\Theta + (F_{AX} + F_{TX}) \tag{3.24}$$

$$m(\dot{v} + ur - pw) = mg\cos\Theta\sin\Phi + (F_{Ay} + F_{Ty}) \tag{3.25}$$

$$m(\dot{w} + pv - qu) = mg\cos\Theta\cos\Phi + (F_{Ay} + F_{Ty}) \tag{3.26}$$

$$\dot{p}I_{xx} + rq(I_{zz} - I_{yy}) = L_A + L_T \tag{3.27}$$

$$\dot{q}I_{yy} + pr(I_{xx} - I_{zz}) = M_A + M_T \tag{3.28}$$

$$\dot{r}I_{xx} + pq(I_{yy} - I_{xx}) = N_A + N_T \tag{3.29}$$

Euler angles are updated with the kinematic equations (3.30), (3.31), (3.32).

$$p = \dot{\Phi} - \dot{\psi}\sin\Theta \tag{3.30}$$

$$q = \dot{\Theta}\cos\Phi + \dot{\Psi}\cos\Theta\sin\Phi \tag{3.31}$$

$$r = -\dot{\Theta}\sin\Phi + \dot{\Psi}\cos\Theta\cos\Phi \tag{3.32}$$

3.3.6 Coordinate Systems

There are two main coordinate systems. These are body frame (B) and inertial frame (I). Body frame is a right-handed coordinate system which is fixed on missile's center of gravity. The x-axis points through the nose, the y-axis points through right side on control surface plane and the z-axis points down which is perpendicular to the xy plane 3.5. Inertial frame is earth-fixed frame.The x-axis points north, the y-axis points east and the z-axis points down.

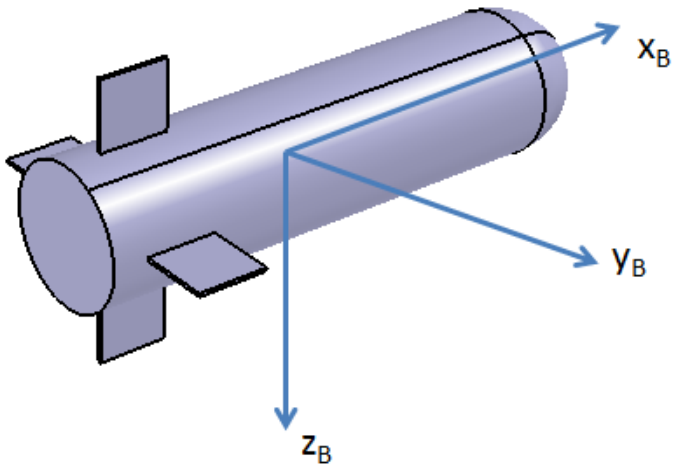

Figure 3.5: Body Coordinate System

3.3.7 Neighborhood Algorithm Modelling

Other agents within a certain distance perimeter of a swarm agent are called neighboring agents. This distance is called the neighborhood distance. In a real environment, if an agent can receive and transmit data including identification and location from other agents, they have a neighborhood relationship. In other words, if data can be transmitted from one agent to another, it is in the neighborhood of it. In the simulation environment, it is modeled without increasing the processing load. Because the fact that these operations will be done in every step will increase the computational cost. P_i is the position of agent i. n_{ij} indicates the neighborhood relationship of the swarm agent "i" with the swarm agent "j". x_{c_i} means the central point of the neighborhood that i-agent has network with other which in neighborhood of it. m is number of munition. In addition, the neighborhood distance is specified with d_n.

$$x = \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ \vdots & \ddots & \vdots \\ P_{i1} & P_{i2} & P_{i3} \end{bmatrix} \tag{3.33}$$

$$n_{ij} = \begin{cases} 1, & \mid P_i - P_j \mid \leq d_n \\ 0, & \mid P_i - P_j \mid > d_n \end{cases} \tag{3.34}$$

$$N_{mat} = \begin{bmatrix} n_{11} & n_{12} & n_{13} & \dots & n_{1j} \\ n_{21} & n_{22} & n_{23} & \dots & n_{2j} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ n_{i1} & n_{i2} & n_{i3} & \dots & n_{ij} \end{bmatrix} \tag{3.35}$$

$$x_{c_i} = \frac{n_{mat} x}{\sum_{j=1}^{m} N_{mat}(i,j)} \tag{3.36}$$

3.4 Inertial Navigation Model

It is assumed that acceleration, angular velocity and Euler angles are ideally measured and calculated. These values are taken from flight mechanics.

CHAPTER 4

CONTROLLER DESIGN

Each system makes calculations in order to be able to follow a certain trajectory, reach the target or take the correct position. These calculated values can be velocity, acceleration or position. In this study, outputs of swarm and guidance algorithm are represented. These outputs are based on reference acceleration commands. A controller is needed to follow the reference acceleration commands. In this section, controller design is explained. The controller design section is separated in three parts. In the first part, a state variable model is obtained. In the second part, a controller design suitable for the linear model is studied. In the third part, model reference adaptive controller is designed in order to use controller at every stage of the flight.

4.1 Linear Model: State Variable Modelling

State variable modeling is used to determine the future behavior of the system, using the current state of the system, with or without input. The dynamic state of a system is the smallest set of variables. Thus, the behavior of the system at any time is determined by knowing the state values at time $t = t_0$ and the input values at time $t \geq 0$.

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned} \tag{4.1}$$

Where equation (4.1), $[A]_{nxn}$ is the state matrix, $[B]_{nxm}$ is the control matrix, $[C]_{lxn}$ is the observation matrix, $[D]_{lxm}$ is the input observation matrix.

By using the small perturbation and steady-state conditions, 6 degree of freedom equations of motion is linearized [23] [24]. In this study, the longitudinal dynamics is considered because the munition is symmetric according to the xy-plane and the xz-plane. In other words, the lateral dynamics is similar to the longitudinal dynamics. According to steady state condition lateral velocity, angular velocity and acceleration are equal to zero. Also roll angle is taken zero. The small perturbation theorem is handled as in equation (4.2). There is state instead of the specified "x" value.

$$\begin{aligned} X &= X + x, \\ \dot{X} &= \dot{X} + \dot{x} = \dot{x} \end{aligned} \tag{4.2}$$

The approximation in the equation in equation (4.3) is considered.

$$\begin{aligned} \cos\theta = \cos\phi &= 1, \\ \sin\theta &= \theta, \\ \sin\phi &= \phi \end{aligned} \tag{4.3}$$

Small perturbation theorem is applied at equation (3.24), (3.26), (3.28) and (3.31).

$$m(\dot{u} + (Q+q)(W+w) - (R+r)(V+v)) = -mg\sin(\Theta+\theta) + (F_{Ax} + F_{Tx}) \tag{4.4}$$

$$\begin{aligned} &m(\dot{w} + (P+p)(V+v) - (Q+q)(U+u)) = \\ &-mg\cos(\Theta+\theta)\cos(\Phi+\phi) + (F_{Az} + F_{Tz}) \end{aligned} \tag{4.5}$$

$$\dot{q}I_{yy} + (P+p)(R+r)(I_{xx} - I_{zz}) = M_A + M_T \tag{4.6}$$

$$Q + q = \dot{\theta}\cos(\Phi+\phi) + \dot{\psi}\cos(\Theta+\theta)\sin(\Phi+\phi) \tag{4.7}$$

Compared with the linear terms, nonlinear terms are considered to be negligible. In addition, small terms are ignored since it is considered in terms of its effect.

$$m(\dot{u} + qW) = -mg\theta\cos\Theta + (F_{Ax} + F_{Tx}) \tag{4.8}$$

$$m(\dot{w} - qU) = -mg\cos(\Theta+\theta)\cos(\Phi+\phi) + (F_{Az} + F_{Tz}) \tag{4.9}$$

$$\dot{q}I_{yy} = M_A + M_T \tag{4.10}$$

$$q = \dot{\theta} \tag{4.11}$$

Force and moments are considered at equation (4.12), (4.13) and (4.14). In this thesis, there is no thrust force for the munition. Therefore, moment and force in terms of thrust are ignored. V_{P_1} is steady state velocity. C_{L_1}, C_{D_1} and C_{m_1} are the steady-state terms. α means the angle of attack. δ_e is the deflection of surface of elevator.

$$\begin{aligned} F_{Ax} = \bar{q}S\Big[&-(C_{D_u} + 2C_{L_1})(\frac{u}{V_{P_1}}) - (C_{D_\alpha} - C_{L_1})\alpha \\ &-C_{D_{\dot{\alpha}}}(\frac{\dot{\alpha}\bar{c}}{2V_{P_1}}) - C_{D_q}(\frac{\dot{\alpha}\bar{c}}{2V_{P_1}}) - C_{D_{\delta_e}}\delta_e\Big] \end{aligned} \tag{4.12}$$

$$\begin{aligned} F_{Az} = \bar{q}S\Big[&-(C_{L_u} + 2C_{L_1})(\frac{u}{V_{P_1}}) - (C_{L_\alpha} - C_{D_1})\alpha \\ &-C_{L_{\dot{\alpha}}}(\frac{\dot{\alpha}\bar{c}}{2V_{P_1}}) - C_{L_q}(\frac{\dot{\alpha}\bar{c}}{2V_{P_1}}) - C_{L_{\delta_e}}\delta_e\Big] \end{aligned} \tag{4.13}$$

$$\begin{aligned} m_{Ax} = \bar{q}S\bar{c}\Big[&(C_{m_u} + 2C_{m_1})(\frac{u}{V_{P_1}}) + C_{m_\alpha}\alpha \\ &+C_{m_{\dot{\alpha}}}(\frac{\dot{\alpha}\bar{c}}{2V_{P_1}}) + C_{m_q}(\frac{\dot{\alpha}\bar{c}}{2V_{P_1}}) + C_{m_{\delta_e}}\delta_e\Big] \end{aligned} \tag{4.14}$$

By using equation (4.12), (4.13) and (4.14), state matrix and control matrix is obtained at equation (4.1). The detailed information about coefficients of the state matrix and the control matrix can be found in Appendix B.

$$\begin{bmatrix} \dot{u} \\ \dot{\alpha} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} X'_u & X'_\alpha & X'_q & X'_\theta \\ Z'_u & Z'_\alpha & Z'_q & Z'_\theta \\ M'_u & M'_\alpha & M'_q & M'_\theta \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u \\ \alpha \\ q \\ \theta \end{bmatrix} + \begin{bmatrix} X'_{\delta_e} \\ Z'_{\delta_e} \\ M'_{\delta_e} \\ 0 \end{bmatrix} \delta_e \tag{4.15}$$

In addition, vertical acceleration is calculated with the equation (4.16) [23].

$$a_z = (V_{P_1} Z'_u)u + (V_{P_1} Z'_\alpha - g\sin\Theta)\alpha + \big(V_{P_1}(Z'_q - 1)\big)q \\ +(V_{P_1} Z'_\theta + g\sin\Theta)\theta + (V_{P_1} Z'_{\delta_e})\delta_e \tag{4.16}$$

$$\begin{bmatrix} a_z \\ u \\ \alpha \\ q \\ \theta \end{bmatrix} = \begin{bmatrix} Z''_u & Z''_\alpha & Z''_q & Z''_\theta \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ \alpha \\ q \\ \theta \end{bmatrix} + \begin{bmatrix} Z''_{\delta_e} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \delta_e \tag{4.17}$$

Equations (4.15) and (4.17) are calculated by using the coefficients in Appendix A. Then, the state space model and the nonlinear model are compared in order to assess whether the linear model is suitable. The comparison is shown at figure 4.1.

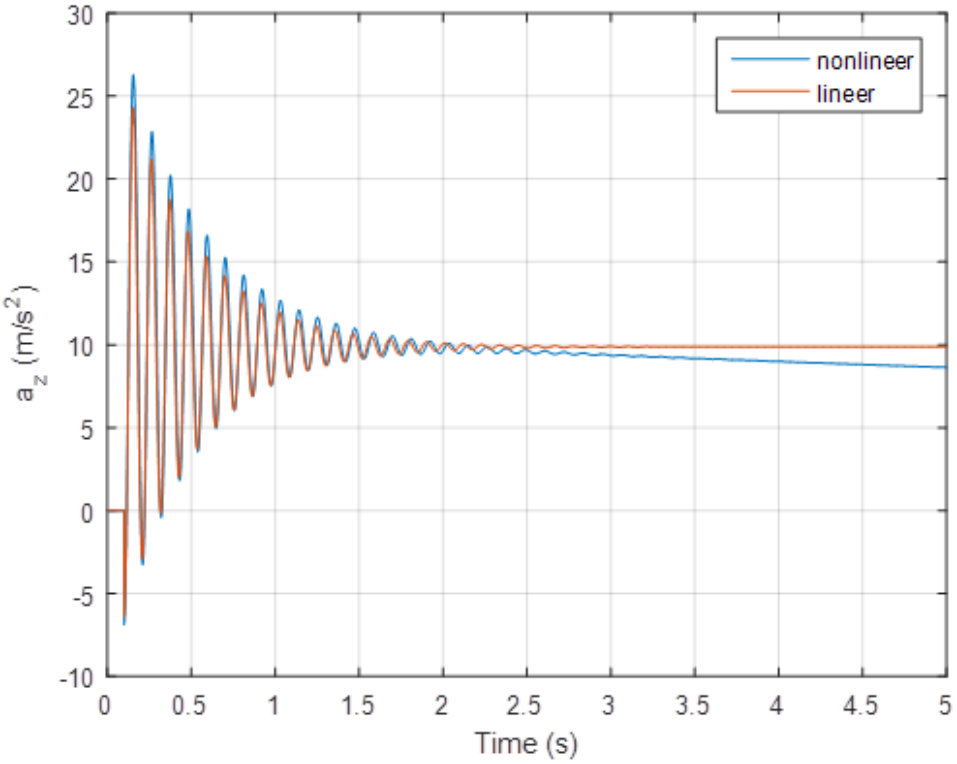

Figure 4.1: Open Loop Response of Linear and Non-Linear Plant

4.2 Linear Model: Controller Design

In this part, the linear controller design is explained. Pole placement method is preferred and linear parameter-varying control approach (LPV control) is applied. A linear model with reduced longitudinal state variable model is used for acceleration tracking control. The reduced linear model is shown at Equation (4.18).

$$\begin{bmatrix} \dot{\alpha} \\ \dot{q} \end{bmatrix} = \begin{bmatrix} Z'_{\alpha} & Z'_{q} \\ M'_{\alpha} & M'_{q} \end{bmatrix} \begin{bmatrix} \alpha \\ q \end{bmatrix} + \begin{bmatrix} Z'_{\delta_e} \\ M'_{\delta_e} \end{bmatrix} \delta_e \tag{4.18}$$

The normal acceleration is obtained from the state variables mentioned above.

$$a_z = (V_{P_1} Z'_{\alpha} - g \sin\Theta)\alpha + \big(V_{P_1}(Z'_q - 1)\big) q + (V_{P_1} Z'_{\delta_e})\delta_e \tag{4.19}$$

The second order linear control actuation system model is included in the reduced order longitudinal linear model. In this way, the effects of concepts such as actuator delay and actuator bandwidth are also included in the control design.

$$\begin{bmatrix} \dot{\delta} \\ \ddot{\delta} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega^2_{n,a} & -2\zeta_a \omega_{n,a} \end{bmatrix} \begin{bmatrix} \delta \\ \dot{\delta} \end{bmatrix} + \begin{bmatrix} 0 \\ \omega^2_{n,a} \end{bmatrix} \delta_c \tag{4.20}$$

For the actuator, $\omega^2_{n,a}$ is the natural frequency and ζ_a is the damping ratio. In addition, δ_c means the actuator command. A new state space is obtained by combining the actuator state space and the system state space.

$$\begin{bmatrix} \dot{\alpha} \\ \dot{q} \\ \dot{\delta} \\ \ddot{\delta} \end{bmatrix} = \begin{bmatrix} Z'_{\alpha} & Z'_{q} & Z_{\delta} & 0 \\ M'_{\alpha} & M'_{q} & M_{\delta} & 0 \\ 0 & 0 & 0 & 1 \\ -0 & 0 & \omega^2_{n,a} & -2\zeta_a \omega_{n,a} \end{bmatrix} \begin{bmatrix} \alpha \\ q \\ \delta \\ \dot{\delta} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \omega^2_{n,a} \end{bmatrix} \delta_c \tag{4.21}$$

$$\begin{bmatrix} a_z \\ q \end{bmatrix} = \begin{bmatrix} Z'_{\alpha} V_{P_1} - g \sin\Theta & V_{P_1}(Z'_q - 1) & Z_{\delta} V_{P_1} & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ q \\ \delta \\ \dot{\delta} \end{bmatrix} \tag{4.22}$$

By using the pole placement method, the eigenvalues of the linear models of the munition are re-shaped using the full-state feed according to the desired behavior of the munition. Controllability of munition is examined to assess whether the pole placement method is applicable. States,inputs and outputs of state space are defined as $x = [x_m, x_a]^T \in \mathbb{R}^n$, $u \in \mathbb{R}^m$ and $y \in \mathbb{R}^p$. Then, the state space representation can be expressed as follows.

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \end{aligned} \tag{4.23}$$

The controllability matrix (W_c) is obtained at Equation (4.24).

$$W_c = \begin{bmatrix} B & AB & A^2B & \dots & A^{n-1}B \end{bmatrix} \tag{4.24}$$

State matrix, control matrix and observation matrix are created according to the parameters in the table 4.1.

Table 4.1: Linearization Condition

Parameter	**Value**	**Unit**
V	140	m/s
Θ	0	Degree
ρ	1.125	kg/m^3

$$A = \begin{bmatrix} -1.2797 & 0.9991 & -0.5810 & 0 \\ -2152 & -2.2746 & -2537.4 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -3947.8 & -87.9646 \end{bmatrix} \tag{4.25}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ -3947.8 \end{bmatrix} \tag{4.26}$$

$$C = \begin{bmatrix} -179.1637 & -0.1326 & -81.3457 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{4.27}$$

The parameter equations for the defined configuration are $n = 4$, $m = 1$ and $p = 2$. If a system is controllable, the columns belonging to the controllability matrix of the system must be linearly independent. Therefore, W_c must be full rank. When the munition's controllability matrix is tested, it is calculated to be full rank.

$$Rank(W_c) = 4 \tag{4.28}$$

There are four eigenvalues for the defined state space. Two of these are the conjugate eigenvalues of the actuator. The other two eigenvalues belong to the munition and dominate the system dynamic according to others. Eigenvalues representing the desired behavior are written in the form of two pairs of conjugate eigenvalues at Equation (4.29).

$$\begin{aligned} s_{1,2} &= -\zeta_m \omega_{n,m} \pm i\omega_{n,m}\sqrt{1-\zeta_m^2} \\ s_{3,4} &= -\zeta_a \omega_{n,a} \pm i\omega_{n,a}\sqrt{1-\zeta_a^2} \end{aligned} \tag{4.29}$$

The desired damping and natural frequency parameters are obtained by using the expressions that characterize the system behavior in the literature. In a situation where the damping ratio is between 0.3 and 0.8, the bandwidth for the desired rise time is obtained [25].

$$\omega_{b,m} \approx \frac{(-1.119\zeta_m + 1.85)(2.16\zeta_m + 0.6)}{t_r} \tag{4.30}$$

t_r specified in the Equation (4.30) means the rise time which is the time required for the step input to be increased from 10 percent of its amplitude to 90 percent. The relationship between bandwidth ($\omega_{b,m}$) and natural frequency is given in equation (4.31).

$$\omega_{b,m} = \omega_{n,m}\sqrt{1 - 2\zeta_m^2 + \sqrt{(1-2\zeta_m^2)^2 + 1}} \tag{4.31}$$

The damping ratio is determined as 0.8 and the rise time is determined as 0.2 seconds. As a result of these design requirements, the desired eigenvalues($s_{1,2}$) are obtained at equation (4.32). In addition, the eigenvalues of the actuator are desired to have a faster dynamic. Therefore, three times the eigenvalues of the system are chosen as the desired eigenvalues of the actuator($s_{3,4}$).

$$\begin{aligned} s_{1,2} &= -8.3622 \pm 6.2716i \\ s_{3,4} &= -25.0865 \pm 18.8149i \end{aligned} \tag{4.32}$$

In this study, the servo system controller is preferred as the autopilot structure is shown at Figure 4.2.

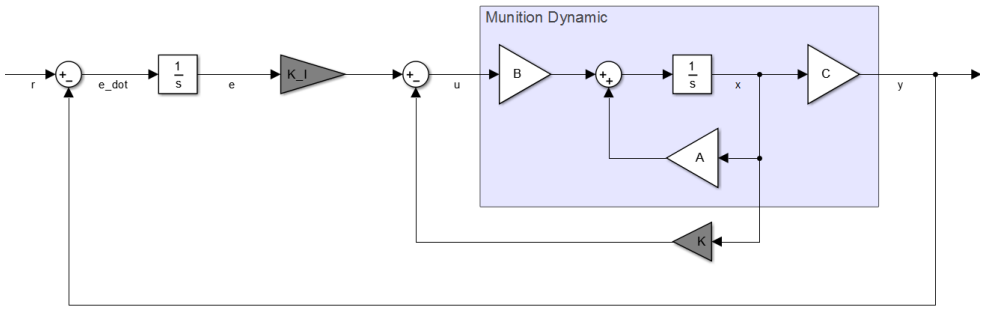

Figure 4.2: Controller Structure

$$u = -\boldsymbol{K}x + k_I\epsilon \tag{4.33}$$

$$\dot{\epsilon} = r - y = r - Cx \tag{4.34}$$

This structure is created for a reference step function. The reference step function is assumed to be applied at time $t = 0$.Thus, the expression specified in equation (4.35) for system dynamics for t> 0 is obtained.

$$\begin{bmatrix} \dot{x}(t) \\ \dot{\epsilon}(t) \end{bmatrix} = \begin{bmatrix} A & 0 \\ -C & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ \epsilon(t) \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} r(t) \tag{4.35}$$

For an asymptotically stable system, states, error and input converge to a constant value as time goes to infinity. In the steady-state state, $\dot{\epsilon}(\infty) = 0$ and $y(\infty) = r$ is obtained. Equation (4.35) is defined as equation (4.36) for the steady state.

$$\begin{bmatrix} \dot{x}(\infty) \\ \dot{\epsilon}(\infty) \end{bmatrix} = \begin{bmatrix} A & 0 \\ -C & 0 \end{bmatrix} \begin{bmatrix} x(\infty) \\ \epsilon(\infty) \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u(\infty) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} r(\infty) \tag{4.36}$$

Considering that $r(t)$ is step input, $r(\infty) = r(t) = r(c)$ equality is obtained for $t > 0$. If the above mentioned equations are taken into account when subtracting equation (4.35) from equation (4.36), equation (4.37) is obtained.

$$\begin{bmatrix} \dot{x_e}(t) \\ \dot{\epsilon_e}(t) \end{bmatrix} = \begin{bmatrix} A & 0 \\ -C & 0 \end{bmatrix} \begin{bmatrix} x_e(t) \\ \epsilon_e(t) \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u_e(t) \tag{4.37}$$

$$u_e(t) = -\boldsymbol{K}x_e(t) + k_I\epsilon_e(t) \tag{4.38}$$

The n + 1 order error vector ($e(t)$) is defined and expressed by equation (4.39).

$$e(t) = \begin{bmatrix} x_e(t) \\ \epsilon_e(t) \end{bmatrix} \tag{4.39}$$

In this case, the space state statement is as follows:

$$\dot{e} = \hat{A}e + \hat{B}u_e \tag{4.40}$$

$$\hat{A} = \begin{bmatrix} A & 0 \\ -C & 0 \end{bmatrix}, \quad \hat{B} = \begin{bmatrix} B \\ 0 \end{bmatrix} \tag{4.41}$$

If the equality is expressed as $\hat{\boldsymbol{K}} = [\boldsymbol{K} k_I]$ and u_e is equal to $-\hat{\boldsymbol{K}}e$, state error equation is obtained at equation (4.42).

$$\dot{e} = (\hat{A} - \hat{B}\hat{K})e \tag{4.42}$$

Provided that the system defined in equation (4.41) is fully controllable, the desired eigenvalues can be assigned to the system in equation (4.42) by pole placement method. When the values is considered,$\hat{A}$ and $\hat{B}$ are found as follows.

$$\hat{A} = \begin{bmatrix} -1.2797 & 0.9991 & -0.5810 & 0 & 0 \\ -2152.0 & -2.2746 & -2537.4 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -3947.8 & -87.9646 & 0 \\ 179.1637 & 0.1326 & 81.3457 & 0 & 0 \end{bmatrix}, \quad \hat{B} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 3947.8 \\ 0 \end{bmatrix} \tag{4.43}$$

$$\hat{W_c} = \begin{bmatrix} B & AB & A^2B & ... & A^{n-1}B \end{bmatrix} \tag{4.44}$$

$$Rank(\hat{W_c}) = 5 \tag{4.45}$$

Four eigenvalues defined at Equation (4.32). In addition to these, one more eigenvalue is defined according to integral part (4.46).

$$s_5 = -15.6791 \tag{4.46}$$

Bu using pole placement method, all gains are found as follow.

$$\hat{\boldsymbol{K}} = \begin{bmatrix} 0.0632 & 0.0119 & -0.8615 & -0.0023 & -0.0015 \end{bmatrix} \tag{4.47}$$

There is a difference between the states for which the controller coefficients are calculated and the parameters that we can measure in the system. For example, while the state matrix contains the angle of attack, the parameter we measure is acceleration. Besides, it is assumed that there is no actuator feedback in the system. Therefore, the projective control is preferred. In equation (4.48), T_j means the eigenvectors which belong to measured states which are acceleration, pitch rate and acceleration error. The gain values obtained for these state variables, [a_z q e], are found by the projective control at (4.49).

$$\bar{K} = \hat{K}T_j(CT_j)^{-1} \tag{4.48}$$

After all calculation, suitable controller coefficients are found for projective control at (4.49).

$$\bar{\boldsymbol{K}} = \begin{bmatrix} -0.0056 & -0.0019 & 0.0092 \end{bmatrix} \tag{4.49}$$

The final work for the linear model is coefficient scheduling. LPV control is preferred because the system requires different controller coefficients under different conditions. Important factor for this method is that the dynamics of the states that change the controller coefficients are slow [26]. In this study, the conditions in which the design of the controller coefficients are made can be found in the table 4.2.

Table 4.2: Gain Scheduling Conditions

Parameter	**Range**	**Unit**
V	[70 100 140 170 180 200]	m/s
Θ	[0 -45 -80]	Degree

The linear controller coefficient calculations are repeated for each value combination specified in the table 4.2. During the simulation, the previously obtained controller gain values are interpolated according to the flight condition. In this way, an approach is provided that controller gain values are updated according to flight conditions.

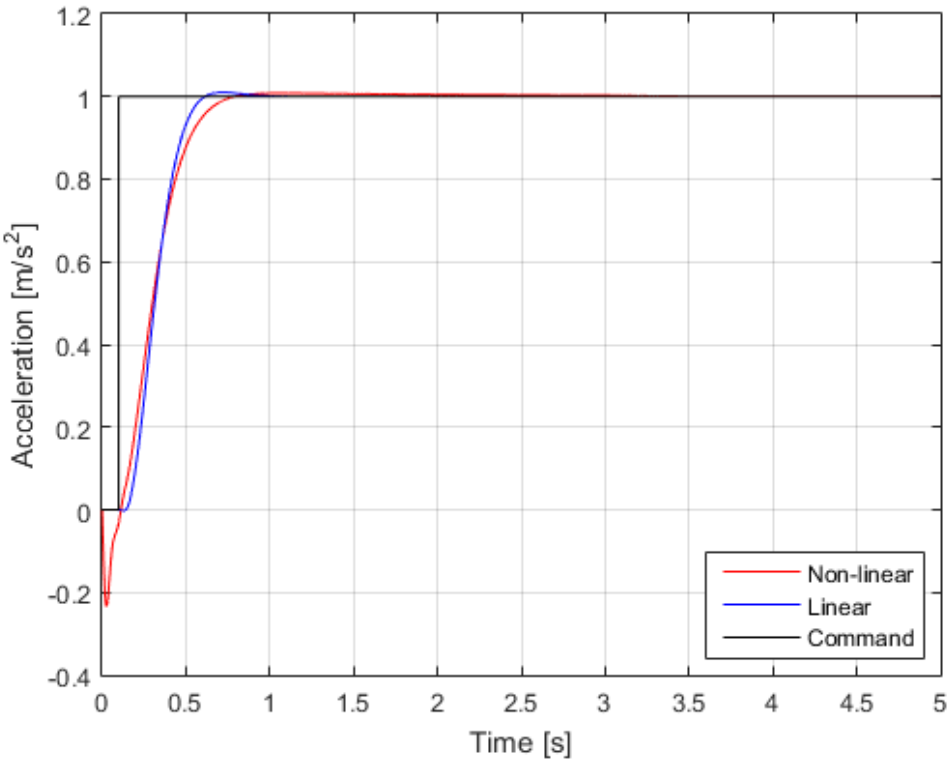

Figure 4.3: Closed Loop Response of Linear and Non-Linear Plant

Closed loop responses for linear and nonlinear systems which belong to linearization point given in table 4.1 are shown in figure 4.3. Two models are considered suitable for the study.

CHAPTER 5

OPTIMIZATION OF THE SWARM ALGORITHM WEIGHTS

In this chapter, optimization method and its application is explained. Particle Swarm Optimization (PSO) method is preferred. PSO was first announced by Kennedy and Eberhart as a stochastic population-based optimization algorithm [27]. It is a preferred algorithm because of its fast convergence to the global optimum point, easy implementation and adjustable with few parameters. The performances of PSO, Genetic Algorithm and Hill climbing methods, are examined for optimization test functions. A hybrid solution with these three algorithms is also added to the study. In the study, it is stated that PSO gives a faster solution in some test functions [28]. PSO is a method of computation that optimizes a problem by iteratively attempting to improve a candidate's solution with respect to a given quality metric. This method is inspired by the behavioral movements of a fish school and a bird flock [27]. The particle population, called candidate solutions, is moved in the search space with simple mathematical formulas. Each particle searches a new position taking into account its own best location, the best position the rest of the population has found, and its instantaneous velocity. The best position means where the cost function has the lowest value. In this way, the population moves towards the optimum result. As shown in equation (5.1), N particles are placed in the scanning area and each particle updates its position with its speed. $X_i(t)$ represents the position of particle of i at t^{th} iteration step. V_i means the velocity of particle of i at t^{th} iteration step.

$$X_i(t+1) = X_i(t) + V_i(t+1) \tag{5.1}$$

The velocity is updated at each iteration step, as specified in equation (5.2). $W_i(t)$ is the inertia weight which represents its inflexibility. r_1 and r_2 are random numbers are between zero and one. These coefficients support the stochastic property of the

optimization method. c_1 is a weight coefficient of acceleration that represents its cognitive ability of the particle. The relationship between the best position($P_{i_{best}}$) the particle has found so far and its own position is called cognitive behavior. c_2 is a weight coefficient of acceleration that represents its social ability for particle. The relationship between the best position (G_{best}) the population has found so far and particle's position is called social behavior.

$$V_i(t+1) = W_i(t)V_i(t) + c_1 r_1(P_{i_{best}} - X_i(t)) + c_2 r_2(G_{best} - X_i(t)) \tag{5.2}$$

ξ is the damping coefficient used in the inertia weight adjustment increases convergence speed.

$$W_i(t+1) = \xi W_i(t+1) \tag{5.3}$$

Algorithm 3 Overall Particle Swarm Algorithms

Problem Definition;

Cost Function, Number of Unknowns

Parameters PSO;

c_1, c_2, *Number of Population, Maximum Iteration*

Initialization;

while *Exists Unassigned Particle* **do**
  Initialize all particle position, velocity, cost and global best
end

Main Loop;

while *iteration* $\leq$ *maximum iteration* **do**
  while *n* $\leq$ *number of population* **do**
    Update velocity, position and cost;
    if *cost* $\leq$ *best cost* **then**
      if *cost* $\leq$ *global cost* **then**
        Update global cost;
      end
      Update best cost;
    end
  end
end

The metrics given in the literature are used while determining the cost function needed

for optimization. The cost function is based on velocity correlation (Ψ), collision avoidance (Ω) and social entropy. Velocity correlation function is related to angular cohesion shown in equation (5.4). The velocity correlation converges to one while swarm agents are aligned each other. Unlike, it converges to zero while swarm agents are not aligned [15].

$$\Psi(t) = \frac{1}{N}\left|\sum_{k=1}^{N} e^{i\psi_k}\right| \tag{5.4}$$

The collision avoidance function calculates the collision frequency with a simple operation which is given at equation (5.5). If this is minimized, collision risks are reduced. In equation (5.5), H is the Heaviside step function.

$$\Omega(t) = \frac{1}{N(N-1)}\sum_{i=1}^{N}\sum_{j\neq i} H\big(r_c - |x_i(t) - x_j(t)|\big) \tag{5.5}$$

Social entropy function is a function that measures how much swarm agents act together. Social entropy is related to the number of agents in clusters determined by distance with respect to a point. This point can be chosen as a fixed point, temporary position of a swarm agent or the instant center of the swarm. In this study, the center of the swarm is taken as the center of cluster during flocking for optimization. Entropy calculation is given at equation (5.6) and (5.7). Where, p_k is the proportion of individuals in the kth cluster.

$$s(h) = \sum_{k=1}^{M} p_k \log_2(p_k) \tag{5.6}$$

$$S = \int_0^{\infty} s(h)dh \tag{5.7}$$

The cost function (J) is chosen in the following equation. When choosing the cost function, coefficients are used to balance the magnitude of order and try to avoid collisions absolutely.

$$J = \int_{t_0}^{t_f} \big(6700(1-\Psi(t)) + 10^6\Omega(t)\big)dt + \max_{t_0\leq t\leq t_f} S(t) \tag{5.8}$$

The change of cost functions after optimization is shown at figure 5.1.

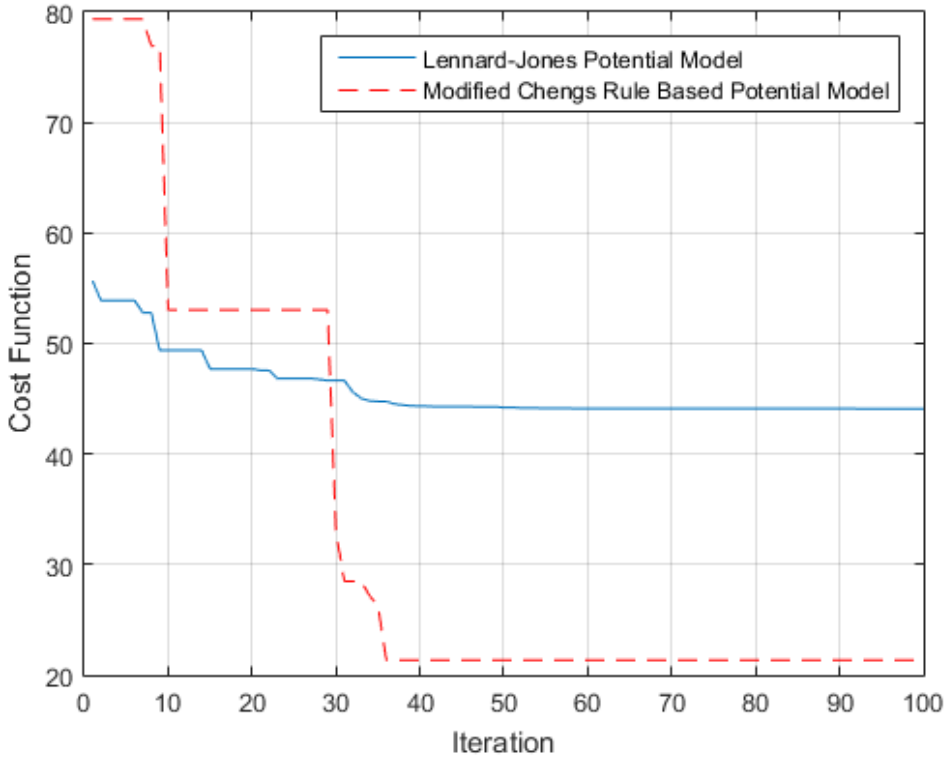

Figure 5.1: Cost Functions of Algorithms

Two different algorithms are studied in this thesis. Therefore different coefficients are optimized for the same cost function. For the first of these algorithms, the weight matrix of acceleration commands based on rules is optimized. Weight coefficients are shown at equation (5.9) and table 5.1.

$$\vec{a^i} = k_1\vec{a_1^i} + k_2\vec{a_2^i} + k_3\vec{a_3^i} + k_4\vec{a_4^i} + k_5\vec{a_5^i} \tag{5.9}$$

Table 5.1: Modified Cheng's Rule Based Potential Model's Optimized Weight Values

k_1	k_2	k_3	k_4	k_5
43.82	320.39	-16.47	-35.19	-28.47

The second algorithm is based on LJ potential functions. In this algorithm, using two potential functions, following the swarm center and push-pull actions among swarm agents are performed. Therefore, in the second algorithm, the LJ potential functions and the weight matrices of the acceleration commands obtained from these functions

are optimized. Weight coefficients are shown at following equations and table 5.2.

$$V_1(r) = 4w_1 \left[\left(\frac{w_2}{r} \right)^{12} - \left(\frac{w_2}{r} \right)^{6} \right] \tag{5.10}$$

$$V_2(r) = 4w_3 \left[\left(\frac{w_4}{r} \right)^{12} - \left(\frac{w_4}{r} \right)^{6} \right] \tag{5.11}$$

$$\vec{a^i} = w_5 \frac{dV_1(r)}{dr} + w_6 \frac{dV_2(r)}{dr} \tag{5.12}$$

Table 5.2: Lennard-Jones Potential Rule Based Model's Optimized Weight Values

w_1	w_2	w_3	w_4	w_5	w_6
6.75	1.75	4.46	1.72	1.02	0.72

Unlike Modified Cheng's Rule Based Potential Mode, Lennard-Jones Potential Rule Based Model has no rules for alignment purposes. Therefore, the optimization result is as expected.

CHAPTER 6

RESULTS AND ANALYSIS

In this section, the swarm algorithms obtained in the thesis are tested with Monte Carlo analysis. There are various disturbance effects in Monte Carlo analysis. Some of these distorting effects are discussed in Chapter 3. They are GPS errors, communication delays and wind effect. In addition to these effects, the release delay from the platform releases and the separation after the platform at different angles and angular velocities. An operational concept depicted are used for analysis. According to this operational concept, the swarm agents are carried by an aircraft flying at a cruise speed. The aircraft releases the agents at a certain point and they glides in a certain time. At the end of gliding motion, the agents reach the speed that they can get efficiency from their aerodynamic surfaces. From this point on, the guidance commands of swarm algorithm specified in Chapter 2 are generated. The swarm agents that follow the algorithm commands flock to the previously specified point. The swarm agents approaching the target point start the decision algorithm. Then, the swarm agents start broadcasting the target information they have been locked on. If each target lockout information can be easily received by the swarm agents, an equal number of agents are directed to each target. There are eight swarm agents and two targets in this study. A value assignment is applied to these targets. Various operational concept errors are also added to the above distorting effects according to the Monte Carlo analysis such as failure of all or several subsystems of any swarm agent. In other words, it will be evaluated whether the false movement of an agent in the swarm mislead whole swarm. In this way, various robustness comparisons are made between the two algorithms.

6.1 Case 1 - Test of The Benefits of The Swarm Algorithm: Collision Test

In this analysis, the benefits of the swarm algorithms are briefly tested and demonstrated. Single simulation is run for each test. In this scenario, two different simulations are made with and without a swarm algorithm under a lateral wind without using other disturbances. In this way, it is aimed to demonstrate the most basic benefit that the algorithm provides for a concept design. In this design, where the wingspan is 100 mm, the safe flight distance is taken as 250 mm from the center of one agent to the center of the other in order to fly the two systems away from each other and to reduce the aerodynamic effects. The information about the scenario and initial conditions in the simulation is given in the table 6.1 and 6.2 .

Table 6.1: Initial Conditions of Agents

ID	Initial Position (I) [m]	Initial Velocity (B) [m/s]	Initial Euler Rotation Angle [deg]	Initial Angular Rates (B) [deg/s]
m_1	[0 -4.5 3000]	[30 0 0]	[0 0 0]	[0 0 0]
m_2	[0 -3.5 3000]	[30 0 0]	[0 0 0]	[0 0 0]
m_3	[0 -2.5 3000]	[30 0 0]	[0 0 0]	[0 0 0]
m_4	[0 -2 3000]	[30 0 0]	[0 0 0]	[0 0 0]
m_5	[0 2 3000]	[30 0 0]	[0 0 0]	[0 0 0]
m_6	[0 2.5 3000]	[30 0 0]	[0 0 0]	[0 0 0]
m_7	[0 -3.5 3000]	[30 0 0]	[0 0 0]	[0 0 0]
m_8	[0 -4.5 3000]	[30 0 0]	[0 0 0]	[0 0 0]

(B): Body-Fixed Axes

(I): Reference Frame

Table 6.2: Wind Velocity and Target Position for Scenario

Parameter	Value
Target Position in NED Reference Frame	[900 0 0]
Wind Velocity in NED Reference Frame	[0 7 0] knot

An aircraft flying at approximately 60 knots cruising speed, and at an altitude of 10000 ft drops eight swarm agents. As a result of the analysis, it is seen that the minimum value of interagent distance shown at Figure 6.1 does not violate the collision threshold value of 0.25 m due to the collision avoidance algorithm provided by the swarm algorithm . However, in a situation where only the guidance algorithm works, it is shown in figure 6.2 that at least one collision occurred while the agent were flying to the target. The fact that the two mass flights are performed under the same conditions and the collision results are different supports why the swarm algorithm should be preferred in collective flights. The contribution of algorithm is not only to prevent collisions, but also to guide the group in harmony while avoiding collisions.

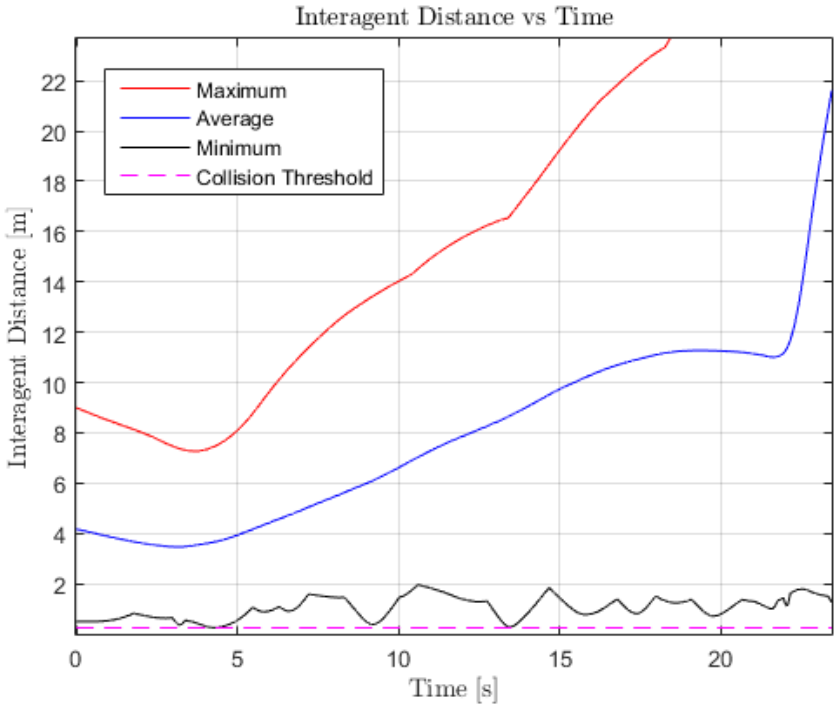

Figure 6.1: Collision Analysis for Swarm Flight

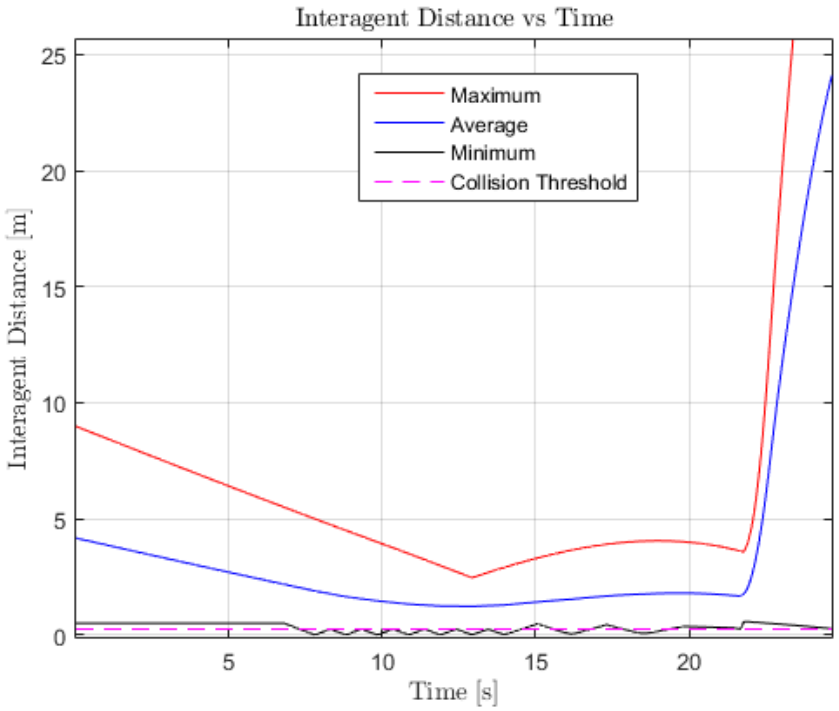

Figure 6.2: Collision Analysis for Mass Flight

6.2 Case 2 - Comparison of The Swarm Algorithms

In this section, the two algorithms mentioned in the study are tested under equal simulation environment conditions. The algorithms' shortcomings and strengths are determined. Monte Carlo method is used for analysis. The simulation of Monte Carlo is basically the generation of arbitrary events or processes using a machine. In this case, Monte Carlo analysis requires random data from certain probability distributions.The Monte Carlo method has a significant value as a method for investigating and interpreting the behavior of random processes and data [29]. Four basic steps are followed for the Monte Carlo analysis. The first step is to define the domain of possible inputs which means the conditions in the simulation environment in this study. The second step is to generate random inputs from the probability distribution on the domain. In the third stage, the calculation is performed with the inputs, which means to run the simulation for this study. Finally, the results are obtained. The results obtained are interpreted with sensitivity analysis.

Correlation coefficients are calculated for sensitivity analysis. Dependence or independence between two or more variables is examined by correlation.

$$\rho_{xy} = \frac{V_{xy}}{\sigma_x \sigma_y} \tag{6.1}$$

V_{xy} represents the covariance of x and y random variables, while σ_x and σ_y represent the standard deviation values of x and y variables.

Robustness and sensitivity to distortions are examined with Monte Carlo analysis results. In the table 6.3, the distorting effects that are subject to the simulation are given.

Table 6.3: The Distortion Parameters for Monte Carlo Analysis

Parameter	Mean	Standard Deviation	Unit
Air Density Multiplier	1	0.05	-
Wind Speed	5	5/3	kn
ϕ	0	5/3	deg
θ	0	5/3	deg
ψ	0	5/3	deg
A/C Velocity	30	10/3	m/s
AGL	3000	100/3	m
Communication Frequency*	[2 5 10 20 50]	-	hz
GPS Frequency*	[2 5 10 20 50]	-	hz
*: Uniform distribution is applied for the values in the specified vector instead of the normal distribution.			

Initial conditions in the simulation are given in table 6.1. In simulation, a fourth-order Runge-Kutta (RK4) is preferred with 0.5 ms step size. Each Monte Carlo's batch consists of 200 runs. The first Monte Carlo analysis is done for the modified Cheng's rule based potential model (MCRBPM). Two targets have equal target value. Equal number of sharing is expected if the swarm agent keep coherence with the help of algorithm. However, sometimes some agents arrive at their target areas earlier than the others and decide on the target. In this case, the target that is not attacked is more valuable due to the algorithm. Thus, agents may not be evenly distributed over the targets. There may be delays due to the nature of the flight and the algorithm. So a distribution of three to five, like four to four, is considered successful. While, other distributions are considered unsuccessful.

Table 6.4: Target Sharing Results for Agents of Swarm of MCRBPM

Target sharing	Percentage
4 to 4	40
5 to 3	39.5
6 to 2	15
7 to 1	4.5
8 to 0	1

According to Monte Carlo analysis, approximately eighty percent of the results given in the table 6.4 provided an appropriate sharing as desired. Agent concentration around the target at the end of flight is another comparison criteria. The circular error probable(CEP) is calculated for each target. It is defined as the radius of the circle centered on the target point, the boundary of which is supposed to include the landing points of fifty percent of the rounds. The landing point is taken as the center of the agents around the target. The Monte Carlo simulation result is presented in Table 6.5.

Table 6.5: The Circular Error Probable for MCRBPM

Target (#)	CEP (m)
1	1.15
2	2.25

The results show that the CEP value is below 5 meters for both targets. The CEP value is at the expected level because the algorithms to prevent collision between agents and provide distance flight from the swarm center effects the approach distance.

Another criteria is the distance between agents. In each simulation carried out for Monte Carlo analysis, maximum, minimum and average interagent distance values are recorded. To give an illustrative example, in a simulation the maximum value of distances between agents in each time step is calculated. The top and bottom values of the calculated data are taken. This is repeated in each simulation in the Monte Carlo analysis. When all data are obtained, the maximum and minimum values of

the maximum distance along Monte Carlo are evaluated. When comparing the two algorithms, unsuccessful simulation scenarios can have a misleading effect. Therefore, obtained average values of miss distance is sorted to from the best scenario to the worst scenario. Then, the best seventy-five percent of the simulation result are evaluated.

Table 6.6: Interdistance Information for MCRBPM

Parameter	Min	Max	Mean	Std	Unit
$\hat{I}_1$	1.66	38.62	6.21	4.88	m
$\hat{I}_2$	3.72	9	7.13	1.27	m

Suppose $f_I : \mathbb{N} \times \mathbb{N} \times \mathbb{N} \to \mathbb{R}$ which calculates interdistance between agents do that $(s, \tau, N) \mapsto f(s, \tau, N)$. Here, s is the number of Monte Carlo simulations, τ is the iteration number and N is the number of bonds between agents.

$$\begin{aligned} \hat{I}_1 &= \max_{\tau} \min_{N} f_I(s, \tau, N) \\ \hat{I}_2 &= \min_{\tau} \max_{N} f_I(s, \tau, N) \end{aligned} \tag{6.2}$$

Last criteria is the data of swarm performance which is used for optimization of the cost function. These are alignment ability, collision avoidance and social entropy. Three of them are compared separately and also cost function is calculate to compare too. In addition, sensitivity analysis is examined for them against to distortions. Alignment performance (Ψ), collision risk (Ω) and social entropy (S) are obtained using the formula below.

$$\Psi(t) = \frac{1}{N} \left| \sum_{k=1}^{N} e^{i\psi_k} \right| \tag{6.3}$$

$$\Omega(t) = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j \neq i} H\big(r_c - |x_i(t) - x_j(t)|\big) \tag{6.4}$$

$$s(h) = \sum_{k=1}^{M} p_k \log_2(p_k) \tag{6.5}$$

$$S = \int_0^{\infty} s(h)dh \tag{6.6}$$

The cost function (J) is chosen as follows:

$$J = \int_{t_0}^{t_f} \big(6700(1 - \Psi(t)) + 10^6\Omega(t)\big)dt + \max_{t_0 \le t \le t_f} S(t) \tag{6.7}$$

The weights of cost functions are used like as Equation (6.7) in order to balance the magnitude of order and try to point to collisions easily.

$$J = \int_{t_0}^{t_f} \big(J_1 + J_2\big)dt + J_3 \tag{6.8}$$

Table 6.7: Costs Obtained From MCRBPM Monte Carlo Runs

Parameter	Min	Max	Mean	Std
J1	12.45	331.40	76.77	52.02
J2	0	14568	262	1700
J3	24.10	179.30	94.31	33.51
Cost	51.86	14697	433	1700

The box chart is used when comparing and analyzing cost functions. This is an easy way to detect and show outliers. Those are values in the data that are much smaller or larger than other values. Furthermore there are other indicator for box plot. One of these is interquartile range which means a statistical spread measure that covers the middle half of an data series and shows the third quartile and first quartile range or difference. Others are maximum and minimum limits which are determined as 1.5 times above or below the quarter span. Box plot information is visualized in figure 6.3.

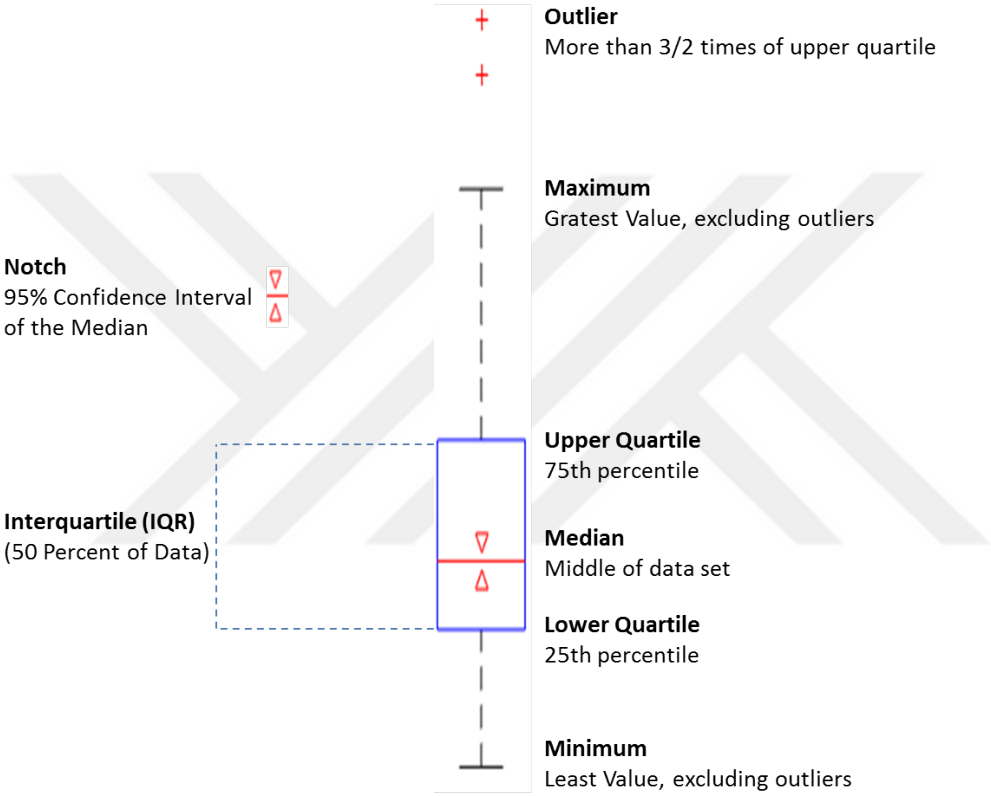

Figure 6.3: Guide for Box Plot

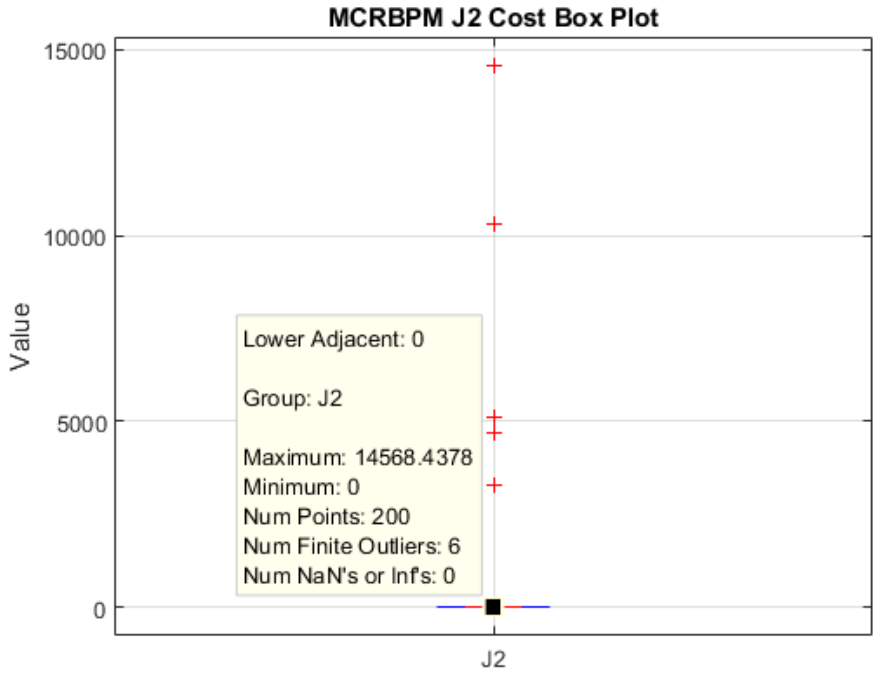

Figure 6.4: J_2 Box Plot for MCRBPM

In table 6.7, a high standard deviation value for J2 is obtained. This is because collisions occurred in six scenarios which are outliers shown in 6.4. These scenarios are eliminated and the values in table 6.8 are recalculated. When the outliers are removed, no change is observed in the maximum values of J1 and J3. This situation may show that the scenario has harsh conditions regardless of the outliers. In addition, by removing outliers, the comparison of the two algorithms is facilitated.When the mean and standard deviation in the table values are examined, also shows that the alignment is affected more than entropy. It may concluded that the algorithm is more dedicated for collective flight than align flight.

Table 6.8: Costs Obtained From MCRBPM Monte Carlo Runs Excluding Six Collision Scenarios

Parameter	Min	Max	Mean	Std
J1	12.45	331.40	77.06	52.30
J2	0	0	0	0
J3	24.10	179.30	93.61	33.00
Cost	51.86	420.62	170.67	68.58

Table 6.9: Sensitivity Analysis for MCRBPM Monte Carlo

Parameter	**J1**	**J2**	**J3**	**Cost**
Initial Altitude	-0,172	0,059	-0,079	0,052
ψ_w	0,004	0,018	0,058	0,020
V_w	-0,030	-0,028	-0,056	-0,030
ϕ	0,070	-0,006	0,009	-0,003
θ	0,032	-0,002	-0,063	-0,002
ψ	0,063	0,081	-0,183	0,080
GPS Frequency	-0,186	0,169	-0,006	0,163
Communication Frequency	0,205	0,208	-0,160	0,211

The sensitivity analysis is a method that identifies the sensitivity of system inputs on outputs. Correlation is used for this analysis. The correlation coefficients are calcu-

lated by using Equation (6.1). When the absolute value of the correlation coefficients indicated in the table is close to one, it shows that sensitivity level of the Monte Carlo parameter is high. On the contrary, when the absolute value of the correlation coefficients is close to zero, it shows that sensitivity level of the Monte Carlo parameter is low. As shown in equation (2.3), alignment command is related to position information. When GPS frequency is more than 5 Hz, this command is more efficient. Therefore, J1 and GPS Frequency are inversely proportional. A similar situation is also valid for J3 cost, which is related to entropy. As the communication frequency increases, agents move together more. This may provide to reduce J3 cost. When Monte Carlo outputs are examined directly, it is seen that J2 is not zero as communication frequency is high. In other words, the frequency of communication is high in simulation runs with collisions. The calculations are repeated and the Table 6.10 is obtained.

Table 6.10: Reorganized Sensitivity Analysis for MCRBPM Monte Carlo

Parameter	**Cost**
Initial Altitude	-0,186
ψ_w	0,107
V_w	-0,087
ϕ	0,096
θ	-0,100
ψ	0,000
GPS Frequency	-0,124
Communication Frequency	0,117

From the Table 6.10, it may be deduced that the altitude at which agents start flying is a very important parameter. This is expected because if swarms start to fly high, it will have more time to execute swarm algorithm. This is a situation that reduces cost values. The above evaluations for GPS and communication frequency are also valid for table 6.10. Using the conditions specified in table 6.1 and table 6.3, the Lennard-Jones Potential Rule Based Model (LJPRBM) is also tested. LJPRBM is an algorithm model with fewer and simpler rules.

Table 6.11: Target Sharing Results for Agents of Swarm of LJPRBM

Target sharing	Percentage
4 to 4	32.5
5 to 3	48.5
6 to 2	13
7 to 1	4
8 to 0	2

Decision algorithm is also used for the two swarm algorithms. Similar to MCRBPM algorithm, LJPRBM algorithm also shows a similar success in target sharing. This target indicates that the sharing algorithm is generic. Sharing results are given at table 6.11

Table 6.12: The Circular Error Probable for LJPRBM

Target (#)	CEP (m)
1	1.25
2	6.72

It is observed that target sharing seems successful while there is a poor guidance to second target.

Table 6.13: Interdistance Information for LJPRBM

Parameter	Min	Max	Mean	Std	Unit
$\hat{I}_1$	3.24	38.76	9.73	5.04	m
$\hat{I}_2$	4.48	9	8.60	1.05	m

According to table 6.6 and table 6.13, it is observed that the MCRBPM algorithm method performs more collective flight than the LJPRBM algorithm and also the interdistance between agents is tighter in the algorithm MCRBPM. This explanation may be related to LJ potential function. LJ potential graph has been shared visually

in previous sections. Due to LJ potential, the repulsion force dominates acceleration comments more quickly than the attractive force. In other words, there is a drastic change in the negative zone, while a rather low increase in the positive zone. As a result, the slightly distant flight of agents from each other is as expected because of the nature of the optimized LJPRBM.

Table 6.14: Costs Obtained From LJPRBM Monte Carlo Runs

Parameter	Min	Max	Mean	Std
J1	20.34	466.65	111.93	72.04
J2	0	5943.40	88.80	666.60
J3	42.79	96.80	115.42	31.09
Cost	87.56	6168.60	316.15	680.17

When all scenarios are included, it is seen that the value of the cost function of LJPRBM algorithm is higher than the algorithm of MCRBPM. LJPRBM provided more robustness to avoid collisions. This can also be seen in the LJPRBM's box plot for J2. Contrary to the previous algorithm, four collisions occurred as shown in figure 6.7. This is a consequence of the lightly dispersed flight of the LJPRBM.

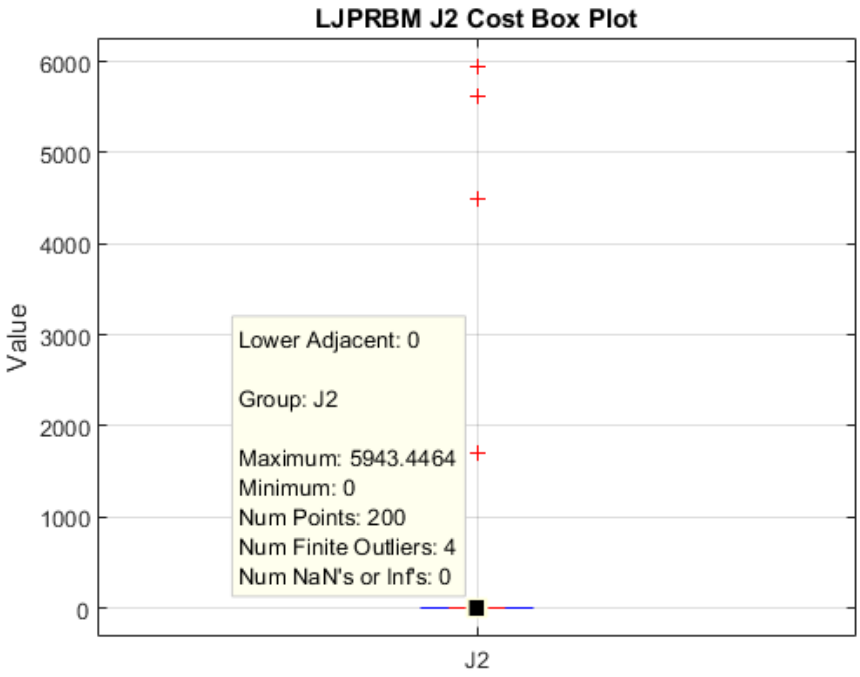

Figure 6.5: J_2 Box Plot for LJPRBM

As before, the following table is obtained again by eliminating the scenarios which have the collision cases.

Table 6.15: Cost Value for LJPRBM Monte Carlo Runs Excluding Four Collision Scenarios

Parameter	Min	Max	Mean	Std
J1	20.34	466.64	111.19	71.73
J2	0	0	0	0
J3	42.79	196.80	115.04	31.23
Cost	87.56	582.54	226.23	82.19

It is expected that the cost of J1 for LJPRBM algorithm is higher than MCRBPM algorithm, because MCRBPM algorithm has a special effort for alignment.As a result, since LJPRBM does not have a specific rule for alignment, the J1 cost function is measured higher. The sensitivity analysis is performed and the table below is shown.

Table 6.16: Sensitivity Analysis for LJPRBM Monte Carlo

Parameter	**J1**	**J2**	**J3**	**Cost**
Initial Altitude	0,102	0,069	0,278	0,091
ψ_w	-0,080	0,009	-0,036	-0,001
V_w	-0,015	-0,049	0,062	-0,047
ϕ	-0,028	-0,018	-0,040	-0,022
θ	0,047	-0,050	0,112	-0,039
ψ	0,061	0,124	0,022	0,129
GPS Frequency	-0,112	0,017	0,028	0,006
Communication Frequency	0,195	-0,085	-0,201	-0,071

According to the sensitivity analysis, J2 has a robust appearance. This is because there are no complex rules for this algorithm, and the agents are mostly focused on cohesion and collision avoidance. The aforementioned situation for the altitude effect is also valid for this algorithm. The scenarios which have collision are excluded from other Monte Carlo analysis results and sensitivity analysis is done again. Table 6.17 is obtained, but with respect to before, there is no substantial difference.

Table 6.17: Reorganized Sensitivity Analysis for LJPRBM Monte Carlo

Parameter	**Cost**
Initial Altitude	0,198
ψ_w	-0,088
V_w	0,013
ϕ	-0,023
θ	0,094
ψ	0,045
GPS Frequency	-0,091
Communication Frequency	0,104

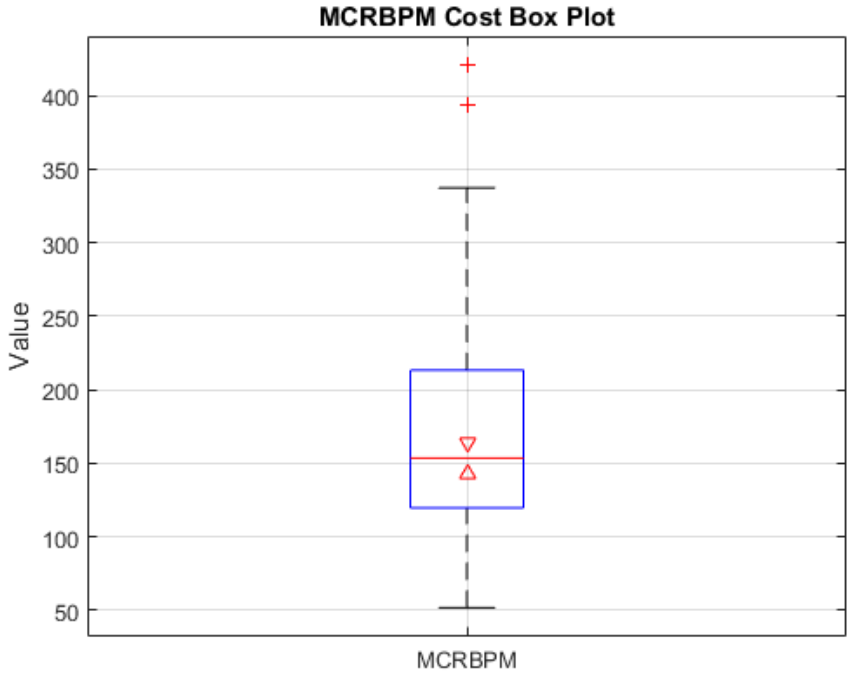

Figure 6.6: Cost Box Plot for MCRBPM

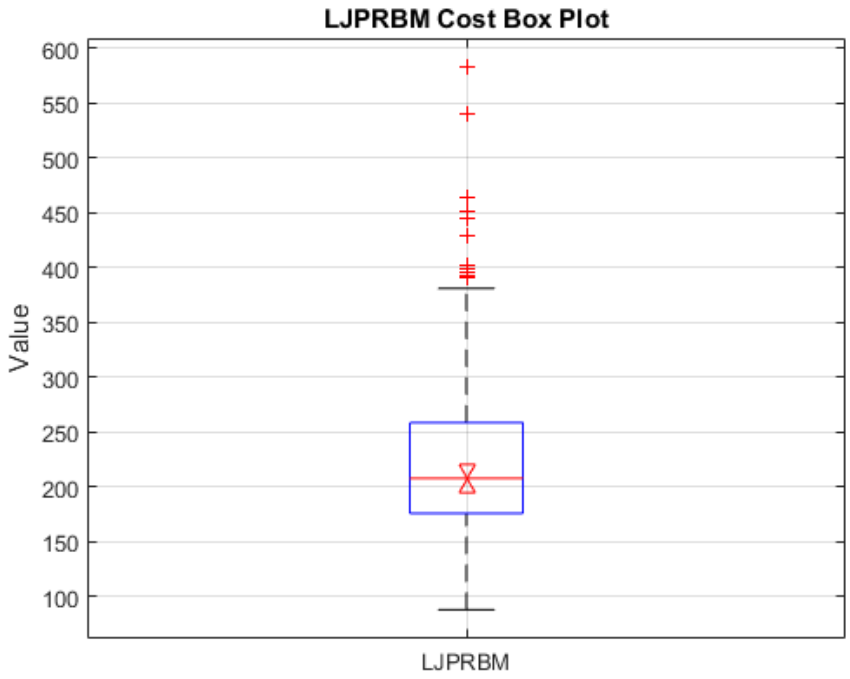

Figure 6.7: Cost Box Plot for LJPRBM

Figure 6.6 and 6.7 are related to overall cost value. When the algorithms are compared to each other, MCRBPM's mean value is about 150 while LJPRBM's mean value is about 200. This shows that MCRBPM's performance is better than other. Also LJPRBM has more overall outliers result than MCRBPM. This confirms the supremacy of MCRBPM in results for distortion Monte Carlo analysis.

6.3 Case 3 - Comparison of The Swarm Algorithms Under Malfunction of CAS

In this section, the malfunction impact of the control actuator system in swarm agents on others is examined. Three scenarios will be created for each algorithm: no malfunction occurs, one agent has a malfunction, and two agents have malfunction about CAS. The robustness of the algorithms is tested in this case study. Changes in the miss distance distance, the change in the lateral acceleration of the non-malfunctioning agents during flight and cost functions are chosen as indicators in the evaluation. Four actuator malfunction modes are modelled: loss of effectiveness, hard-over, and oscillatory failure [30]. The model and the related parameters are shown below.

$$u_{CAS} = \sigma_f \, k_f \, u_{CAS} + (1 - \sigma_f) u_f \tag{6.9}$$

Table 6.18: Model parameters for actuator faults

	Loss of Effectiveness	Locking-in-place	Hard-over	Oscillatory	No fault
σ_f	1	0	0	0	1
k_f	$0 < k_f < 1$	N/A	N/A	N/A	1
u_f	0	$u_{CAS}(t_f)$	u_S	Periodic	0

Here t_f is time when the fault occurred and k_f means the efficiency coefficient. u_{CAS} is the input, u_S is the saturated input limit. First fault scenario is oscillatory case. The model equation is given in the Equation (6.10).

$$u_{CAS} = \sin 5t \tag{6.10}$$

A single target is preferred so that the efficiency of the decision making algorithm does not affect this study. In addition, wind is also eliminated and case study is simulated. 50 Monte Carlo runs is done with RK4 solver and 1 ms step size, and the Table 6.19 is obtained.

Table 6.19: Oscillatory Failure Scenario

	No Fault (Mean, Std)	One Agent (Mean, Std)	Two Agents (Mean, Std)
MCRBPM: J	230.3 , 503.6	228.1, 372.7	310.2, 740.0
LJPRBM: J	268.2, 428.2	355.3, 594.7	505.4, 928.3
	No Fault (Type1, Type2)	One Agent (Type1, Type2)	Two Agents (Type1, Type2)
MCRBPM: Lateral Acc.	x , y	1.02x, 1.29y	1.01x, 1.30y
LJPRBM: Lateral Acc.	a, b	a, 1.30b	a, 1.26b
	No Fault (Mean, Std)	One Agent (Mean, Std)	Two Agents (Mean, Std)
MCRBPM: Miss Distance	0.25, 0.32	0.27, 0.33	0.40, 0.35
LJPRBM: Miss Distance	0.10, 0.02	0.22, 0.26	1.27, 7.23

According to result, the oscillatory control actuator system fault is a problem for MCRBPM because cost function which is given in the equation (6.7) is appeared that it is increased with negative scenarios. However, the change in cost function is higher for the LJPRBM. In this thesis, it is stated before that dispersed flight took place for LJPRBM algorithm. For this reason, distances between agents increase even more with distortions. In summary, this directly increases the cost function of the algorithm. Lateral accelerations are examined. There is no significant change in the maximum lateral acceleration value between the agents (Type 1) for either algorithm. Nevertheless there is an increase of 20 to 30 percent in the total accelerations demanded of whole swarm (Type2). The two algorithms do not differ significantly from each other in this evaluation. When the comparison is made based on the miss distance, the superiority of the MCRBPM algorithm is seen. Based on this case study, the MCRBPM can be evaluated more robust.

The second fault scenario is saturation case. The model equation is given in the Equation (6.11). Saturation limit is taken by one degree. After simulation, the Table

6.20 is obtained.

$$u_{CAS} = \begin{cases} u_{CAS} & \text{if } u_{CAS} \leq u_S \\ u_S, & \text{if } u_{CAS} > u_S \end{cases} \tag{6.11}$$

Table 6.20: Saturation Failure Scenario

	No Fault (Mean, Std)	One Agent (Mean, Std)	Two Agents (Mean, Std)
MCRBPM: J	230.3 , 503.6	137, 135.3	304.9, 632.9
LJPRBM: J	268.2, 428.2	238.0, 285.9	256.6, 514.2
	No Fault (Type1, Type2)	One Agent (Type1, Type2)	Two Agents (Type1, Type2)
MCRBPM: Lateral Acc.	x, y	1.03x, 0.9y	x, 0.80y
LJPRBM: Lateral Acc.	a, b	a, 0.91b	0.99a, 0.73b
	No Fault (Mean, Std)	One Agent (Mean, Std)	Two Agents (Mean, Std)
MCRBPM: Miss Distance	0.25, 0.32	0.28, 0.20	1.61, 2.31
LJPRBM: Miss Distance	0.10, 0.02	1.56, 8.95	1.02, 4.54

In contrast to the oscillation fault, a different trend is obtained in saturation fault in cost function. In this type of error, the CAS commands are limited. This does not make the agent show an action opposite to the desired acceleration command. It only provides a lower magnitude of action towards the requested acceleration. Therefore, the obtained results are different from the oscillation scenario and the value of cost functions is calculated lower. Limiting the requested control actuator system commands may have provided an advantage to the LJPRBM. This limitation is also reflected in the requested accelerations and the observed maximum acceleration. However, superiority in miss distance still belongs to MCRBPM. When all the comparisons are made, it can be evaluated that LJPRBM is more successful in this case study.

The last fault tested in case study is loss of effectiveness of control actuator system. According to this failure case study, the CAS lost thirty percent of its power. The related model equation is given in Equation 6.12. After doing Monte Carlo analysis, the Table 6.21 is obtained.

$$u_{CAS} = 0.7u_{CAS} \tag{6.12}$$

Table 6.21: Loss of Effectiveness Failure Scenario

	No Fault (Mean, Std)	One Agent (Mean, Std)	Two Agents (Mean, Std)
MCRBPM: J	230.3 , 503.6	297.8, 533.9	281.9, 836.0
LJPRBM: J	268.2, 428.2	328.8, 758.6	252.4, 381.6
	No Fault (Type1, Type2)	One Agent (Type1, Type2)	Two Agents (Type1, Type2)
MCRBPM: Lateral Acc.	x, y	1.01x, 0.96y	1.01x, 0.95y
LJPRBM: Lateral Acc.	a, b	1.01a, b	a, 0.96b
	No Fault (Mean, Std)	One Agent (Mean, Std)	Two Agents (Mean, Std)
MCRBPM: Miss Distance	0.25, 0.32	0.25, 0.34	0.25, 0.34
LJPRBM: Miss Distance	0.10, 0.02	0.13, 0.11	0.13, 0.12

The decrease in CAS efficiency has been a case where both swarm algorithms did not face any problems. The comments in the saturation scenario are also valid in this case study. The necessity of accelerations of the munitions has decreased. Also, this may indicate that control actuator commands are adequate to less than expected. In addition, it is appeared that the minimizing the acceleration commands can be added to the optimization cost function in the future studies.

At the end of the cases, the study for CAS faults analysis shows that the swarm could remain successful despite the problems experienced during the mission. This success can be associated with the three main advantages which are mentioned at the beginning of this thesis. Despite the losses and malfunctions during the mission,

the swarm agents preserved their integrity thanks to the algorithm. This points out that the robustness is gained for munition mass flight. In addition, the combinations of initial conditions, faults, communication frequency and missions are examined, and algorithm keeps its validity for almost all scenario. This shows that flexibility is provided for the munitions swarm. The last one is scalability which is related to number of agents which are used in same swarm algorithm. The number of swarm agents are not changed in this study which can be placed in the realistic simulations class. This reason is caused that the number of agents can not be changed easily. Also, scalibility is demonstrated with kinematic simulation in literature. However, the algorithms used in this thesis are derived for kinematic equation at first and then they are used for 6 DoF simulations. It is stated in the literature that algorithms based on models used in kinematic studies are scalable. As a result of this, the algorithms preferred in this study are scalable.

CHAPTER 7

CONCLUSIONS

In this study, two different swarm algorithm alternatives, MCRBPM and LJPRBM, are examined using a simulation for a swarm of munitions. The munition of the swarm has 40 mm diameter, 180 mm length and 0.9 kg mass. The aerodynamic database obtained from Missile DATCOM software is used in the simulation environment in order to predict the munition dynamics. The flight mechanics model, subsystem such as seeker and control actuator system and atmosphere were also modeled. A state variable model was derived in order to design controller. The controller was designed based on linearized models, and gains of the desired controller were obtained with the help of pole placement and projection control method. A target selection algorithm is also implemented. The weight matrices of the swarm algorithms were optimized by using particle swarm optimization method for nominal case. Optimized algorithms were examined by means of Monte Carlo analysis. In this examination, sensitivity analysis, performance cooperation and contribution of algorithms were evaluated. As a result, when the two algorithms are compared, it is observed that the MCRBPM algorithm performed better. The LJPRBM algorithm is not considered as a failure, but had a lower performance for the simulations presented in this manuscript. However, the LJPRBM algorithm is promising, as there are fewer rules and only depends on a basic potential structure based on location. Also, the required processor load for the LJPRBM algorithm is less than the MCRBPM algorithm. For this reason, for a cost-free approach, MCRBPM has a better performance.

REFERENCES

[1] G. Beni, "From swarm intelligence to swarm robotics," vol. 3342, pp. 1–9, 2004.

[2] E. Sahin, "Swarm robotics: From sources of inspiration to domains of application.," pp. 10–20, 2004.

[3] H. Cheng, J. Page, and J. Olsen, "Cooperative control of uav swarm via information measures," *Journal of Intelligent and Robotic Systems*, vol. 1, pp. 256–275, 2013.

[4] C. Reynolds, "Flocks, herds, and schools: A distributed behavioral model," *Computer Graphics*, vol. 21, pp. 25–34, 1987.

[5] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical Review E*, vol. 51, pp. 4282–4286, 1995.

[6] X. Tan, "Self-organization of autonomous swarms via langevin equation," *2007 46th IEEE Conference on Decision and Control*, pp. 1435 – 1440, 2007.

[7] J.-H. Son, H.-S. Ahn, and J. Cha, "Lennard-jones potential field-based swarm systems for aggregation and obstacle avoidance," in *2017 17th International Conference on Control, Automation and Systems (ICCAS)*, pp. 1068–1072, 2017.

[8] S. Yip, *Handbook of materials modeling*. Springer Science & Business Media, 2007.

[9] G. Beni and J. Wang, "Swarm intelligence in cellular robotic systems," in *Robots and biological systems: towards a new bionics?*, pp. 703–712, Springer, 1993.

[10] H. Cheng, J. Page, and J. Olsen, "Dynamic mission control for uav swarm via task stimulus approach," *American Journal of Intelligent Systems*, vol. 2, no. 7, pp. 177–183, 2012.

[11] K. Giles, "A framework for integrating the development of swarm unmanned aerial system doctrine and design," 2016.

[12] G. Vásárhelyi, C. Virágh, G. Somorjai, N. Tarcai, T. Szörenyi, T. Nepusz, and T. Vicsek, "Outdoor flocking and formation flight with autonomous aerial robots," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3866–3873, 2014.

[13] C. Virágh, G. Vásárhelyi, N. Tarcai, T. Szörényi, G. Somorjai, T. Nepusz, and T. Vicsek, "Flocking algorithm for autonomous flying robots," *Bioinspiration & Biomimetics*, vol. 9, no. 2, 2014.

[14] C. Virágh, M. Nagy, C. Gershenson, and G. Vásárhelyi, "Self-organized uav traffic in realistic environment," 2016.

[15] A. Turgut, H. Çelikkanat, F. Gökçe, and E. Sahin, "Self-organized flocking in mobile robot swarms," *Swarm Intelligence*, vol. 2, pp. 97–120, 2008.

[16] J. E. Jones, "On the determination of molecular fields.—i. from the variation of the viscosity of a gas with temperature," *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, vol. 106, no. 738, pp. 441–462, 1924.

[17] R. Yanushevsky, *Modern missile guidance*. CRC Press, 2018.

[18] "U.s. standard atmosphere, 1976," *U.S. Government Printing Office, Washington, DC*, vol. 40, p. 56, 1976.

[19] B. L. Decker, "World geodetic system 1984," 1986.

[20] S. G. Mathisen, F. S. Leira, H. H. Helgesen, K. Gryte, and T. A. Johansen, "Autonomous ballistic airdrop of objects from a small fixed-wing unmanned aerial vehicle," *Autonomous Robots*, pp. 1–17, 2020.

[21] J. S. Touma, "Dependence of the wind profile power law on stability for various locations," *Journal of the Air Pollution Control Association*, vol. 27, no. 9, pp. 863–866, 1977.

[22] P. H. Zipfel, *Modeling and simulation of aerospace vehicle dynamics*. American Institute of Aeronautics and Astronautics, 2007.

[23] M. Napolitano, *Aircraft Dynamics: From Modeling to Simulation*. Wiley Global Education, 2011.

[24] J. Roskam, *Airplane flight dynamics and automatic flight controls*. DARcorporation, 1998.

[25] R. C. Dorf and R. H. Bishop, *Modern control systems*. Pearson, 1998.

[26] J. S. Shamma and M. Athans, "Gain scheduling: potential hazards and possible remedies," *IEEE Control Systems Magazine*, vol. 12, no. 3, pp. 101–107, 1992.

[27] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4, pp. 1942–1948 vol.4, 1995.

[28] T. Krink and M. Løvbjerg, "The lifecycle model: Combining particle swarm optimisation, genetic algorithms and hillclimbers," pp. 621–630, 2002.

[29] D. P. Kroese, T. Brereton, T. Taimre, and Z. I. Botev, "Why the monte carlo method is so important today," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 6, no. 6, pp. 386–392, 2014.

[30] J. Marzat, H. Piet-Lahanier, F. Damongeot, and E. Walter, "Model-based fault diagnosis for aerospace systems: A survey," *Proceedings of the Institution of Mechanical Engineers Part G Journal of Aerospace Engineering*, vol. 226, pp. 1329–1360, 2012.

Appendix A

AERODYNAMIC COEFFICIENTS

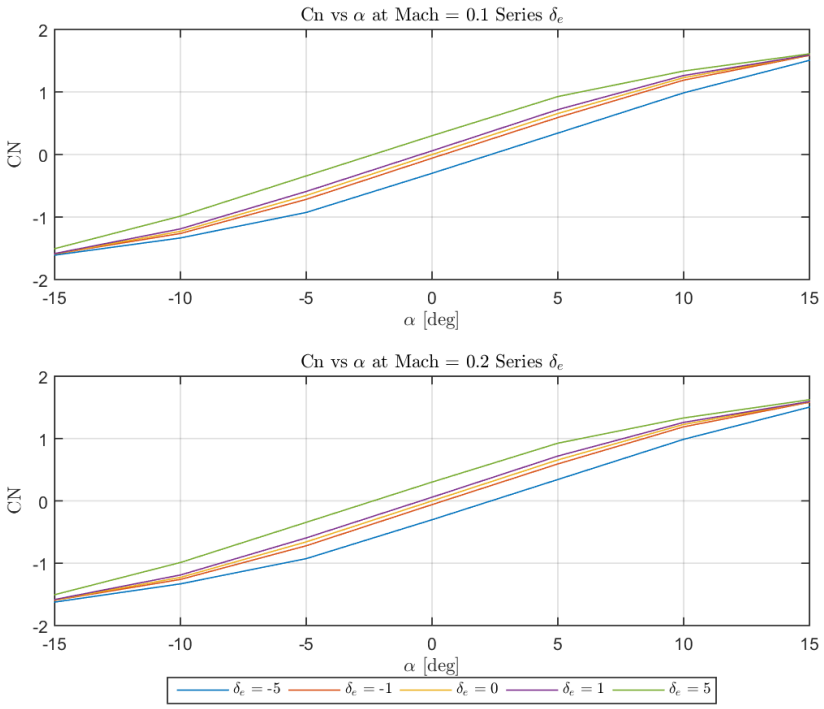

Figure A.1: C_N vs α at 0.1 and 0.2 Mach

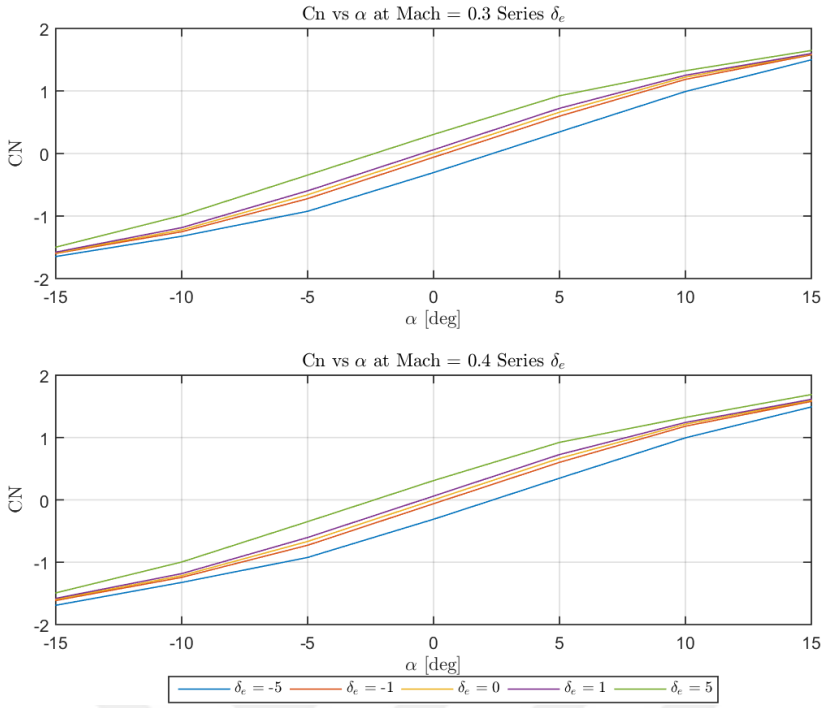

Figure A.2: C_N vs α at 0.3 and 0.4 Mach

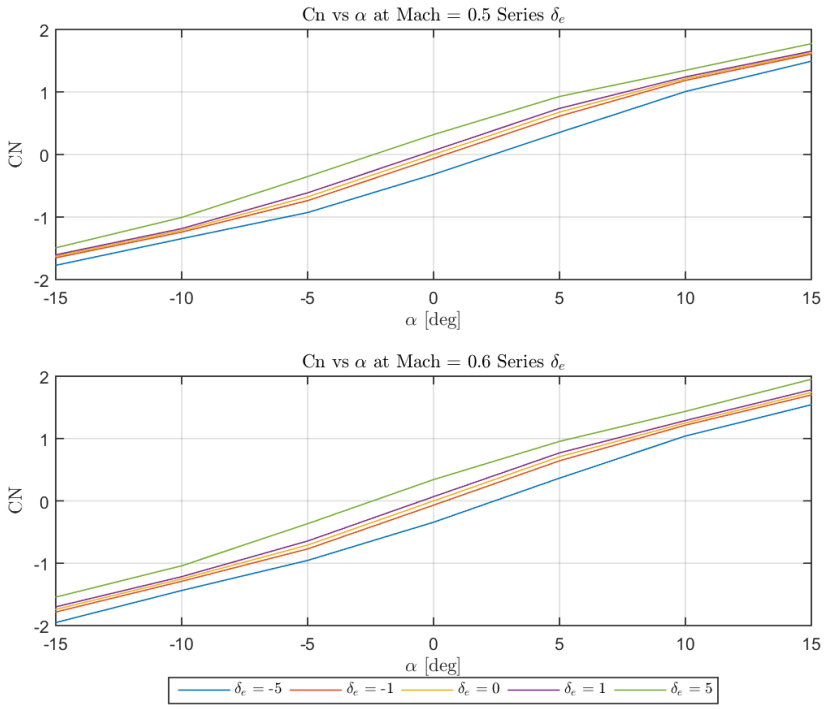

Figure A.3: C_N vs α at 0.5 and 0.6 Mach

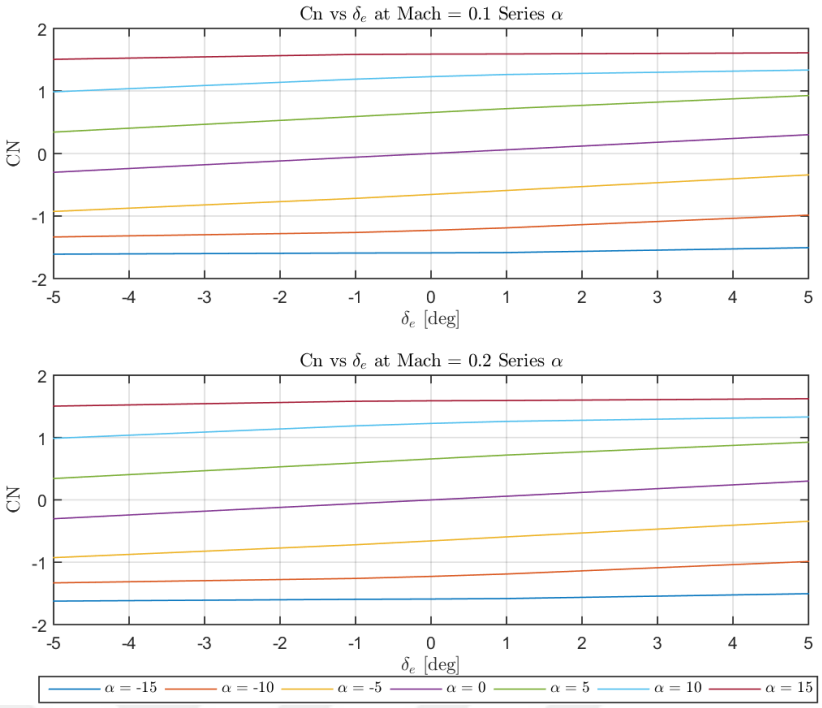

Figure A.4: C_N vs δ_e at 0.1 and 0.2 Mach

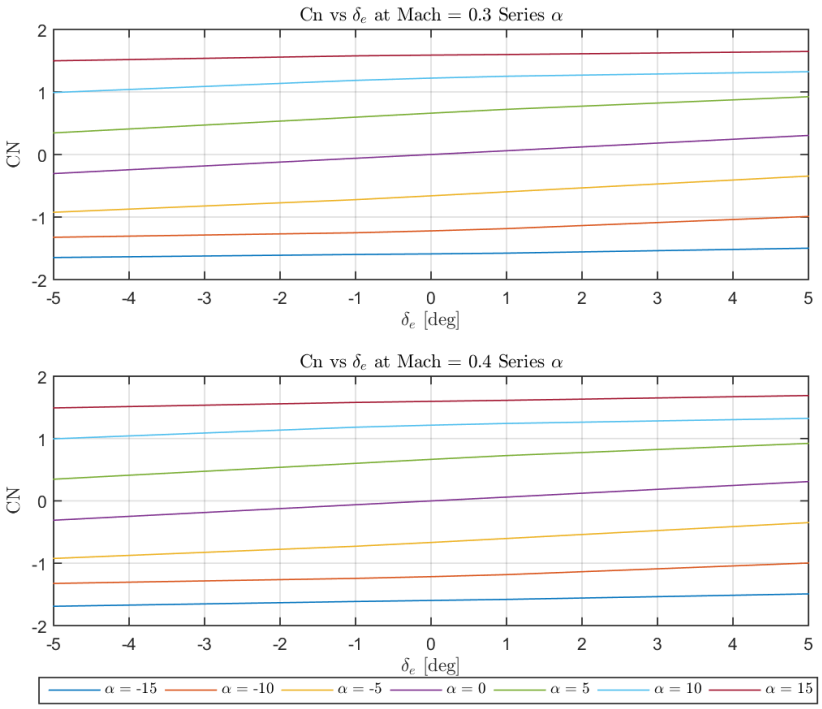

Figure A.5: C_N vs δ_e at 0.3 and 0.4 Mach

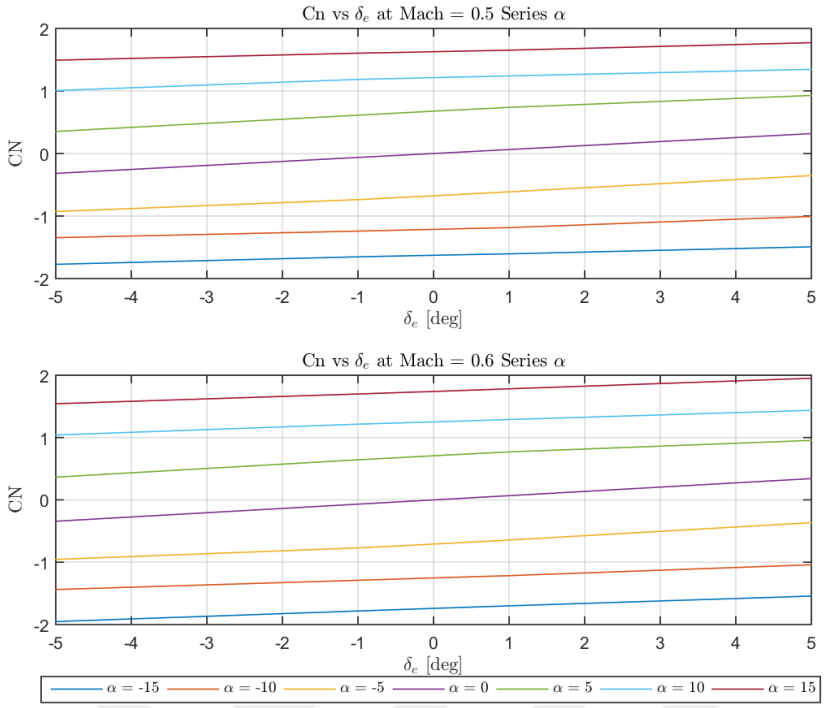

Figure A.6: C_N vs δ_e at 0.5 and 0.6 Mach

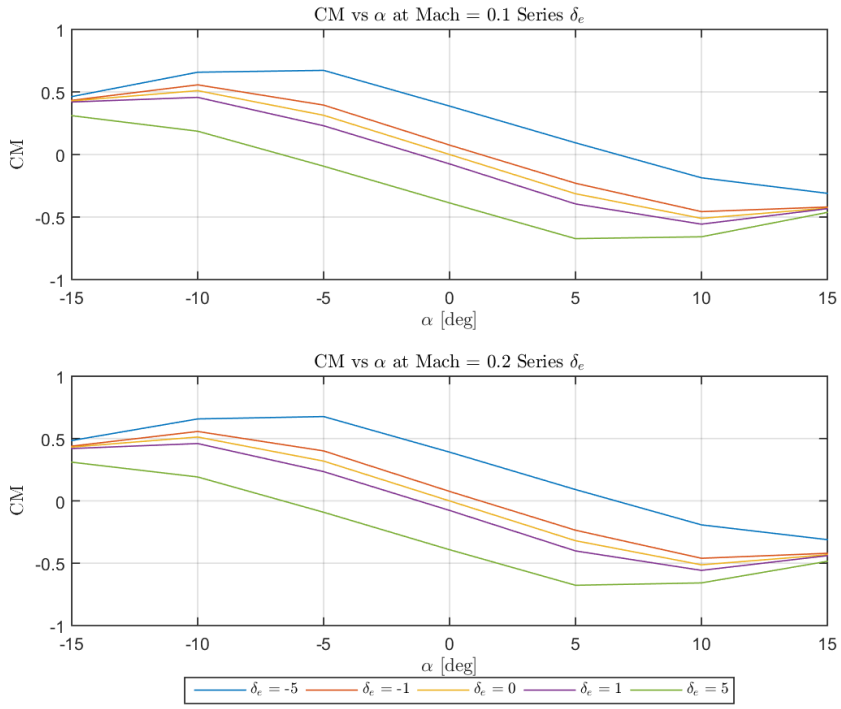

Figure A.7: C_M vs α at 0.1 and 0.2 Mach

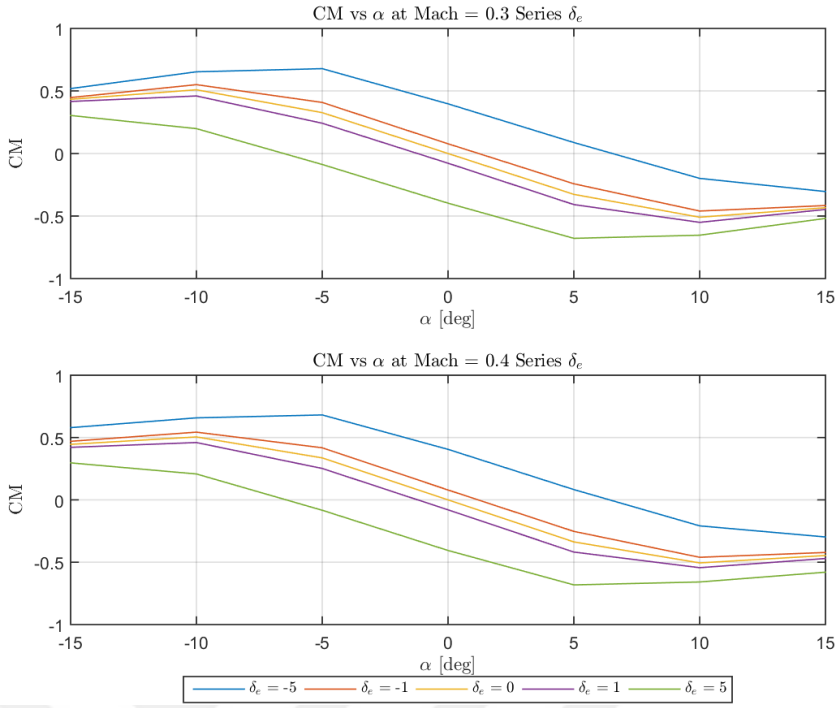

Figure A.8: C_M vs α at 0.3 and 0.4 Mach

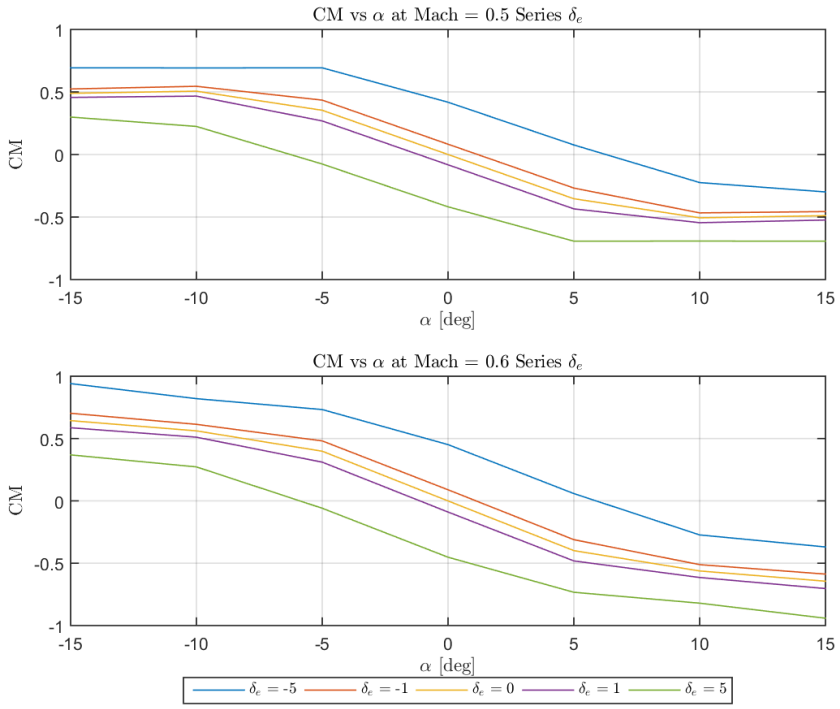

Figure A.9: C_M vs α at 0.5 and 0.6 Mach

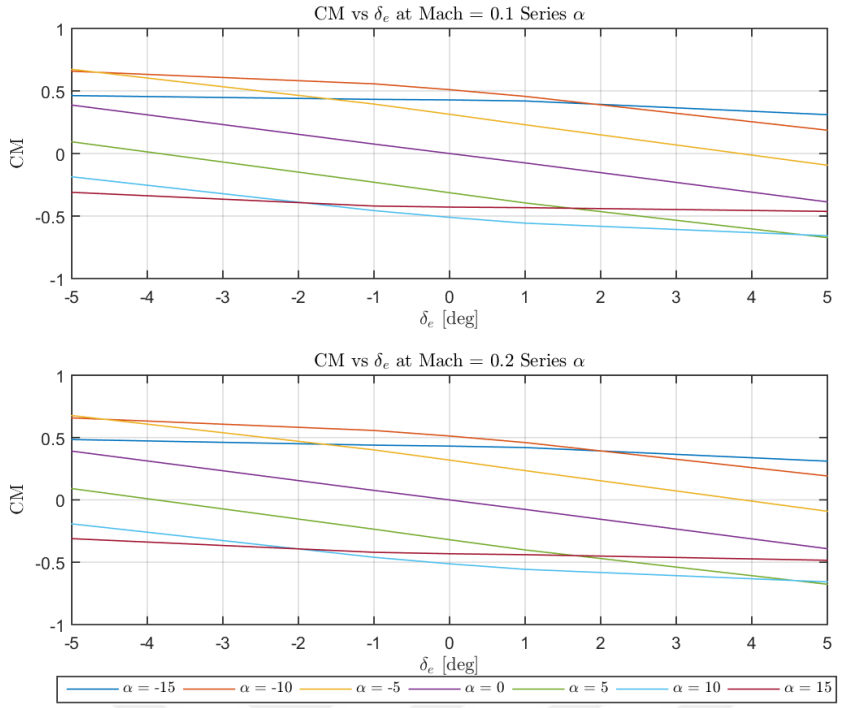

Figure A.10: C_M vs δ_e at 0.1 and 0.2 Mach

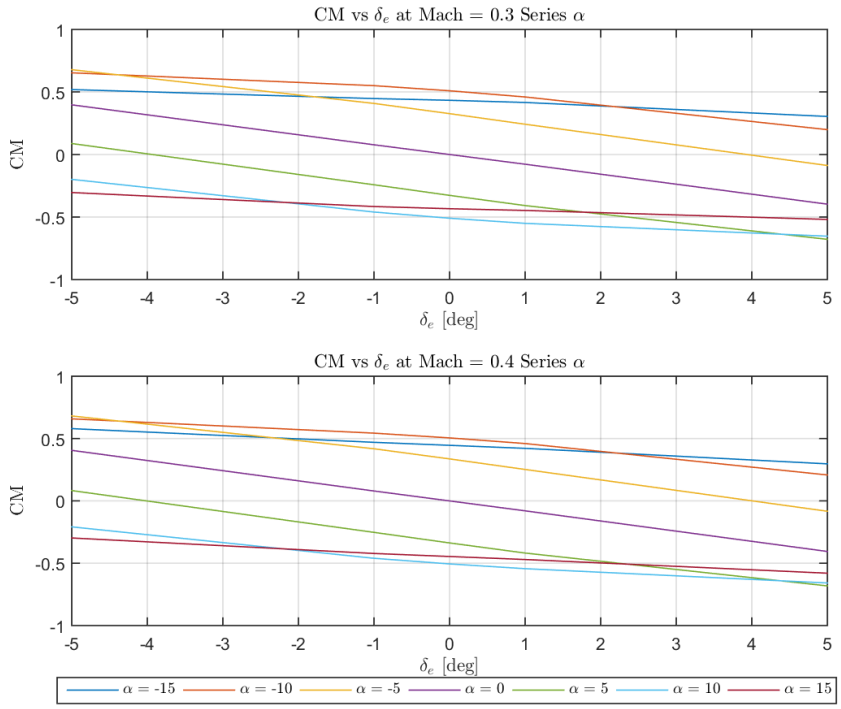

Figure A.11: C_M vs δ_e at 0.3 and 0.4 Mach

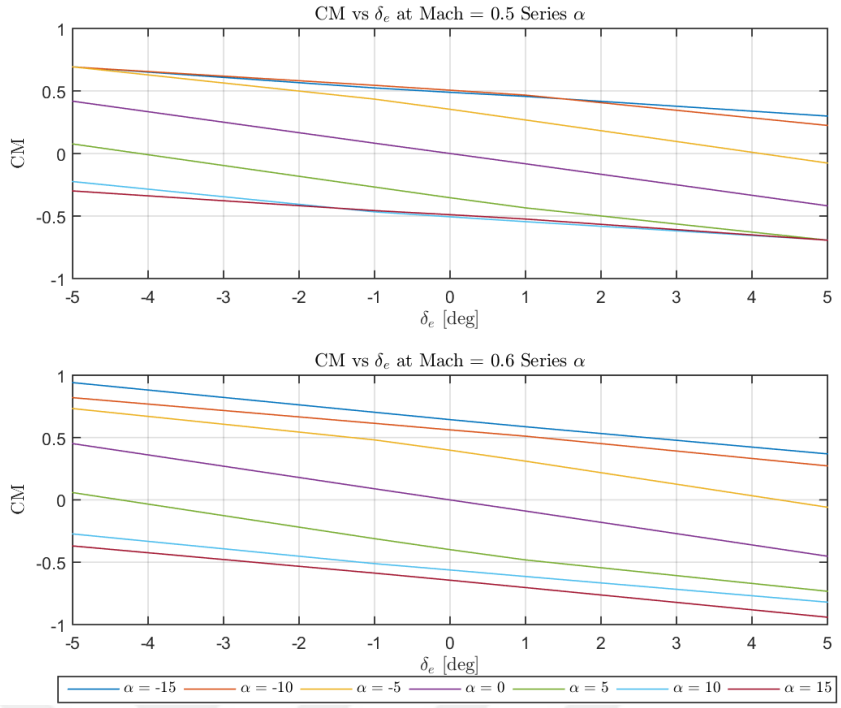

Figure A.12: C_M vs δ_e at 0.5 and 0.6 Mach

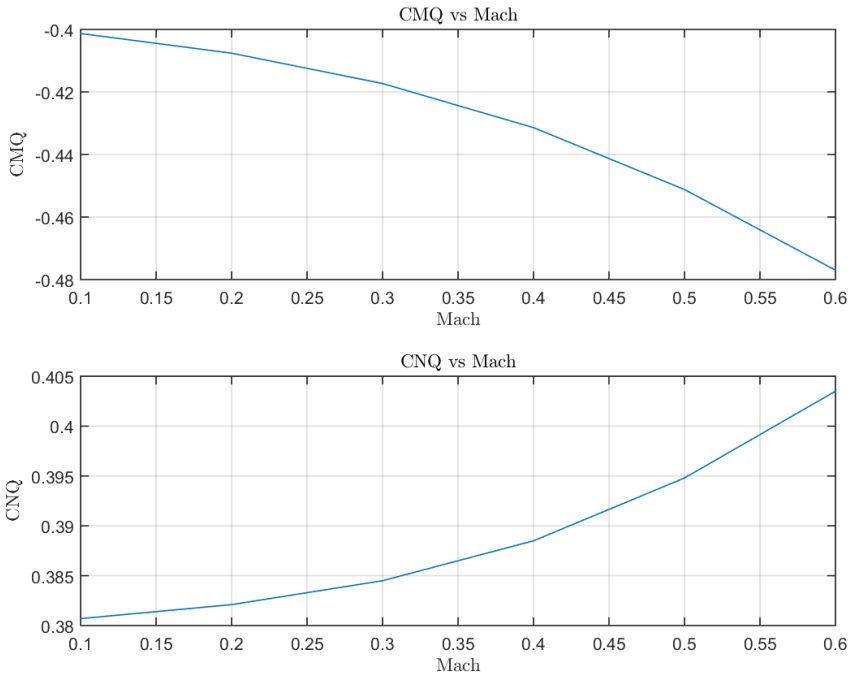

Figure A.13: C_{Mq} vs Mach C_{Nq} vs Mach

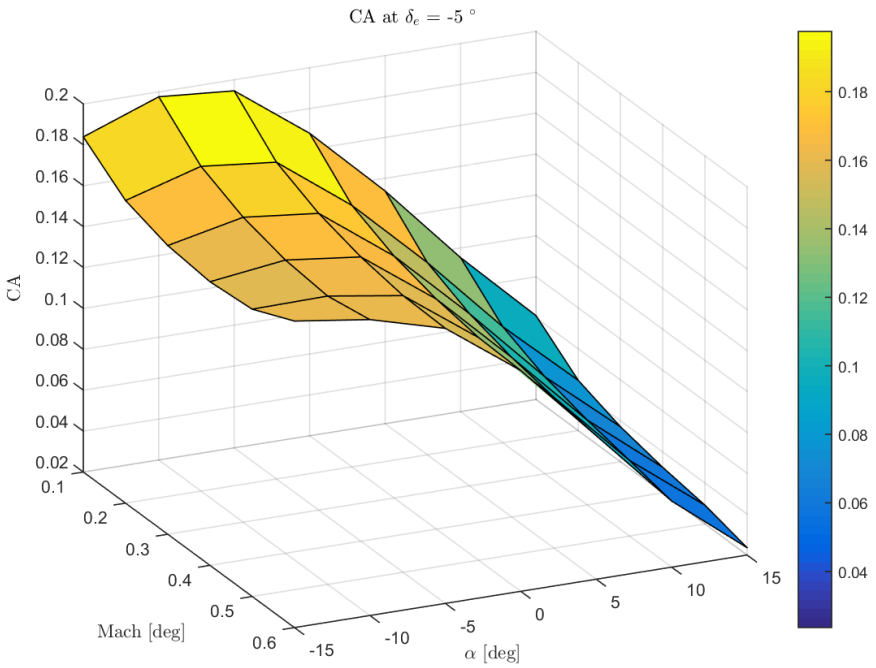

Figure A.14: C_A at δ_e = -5

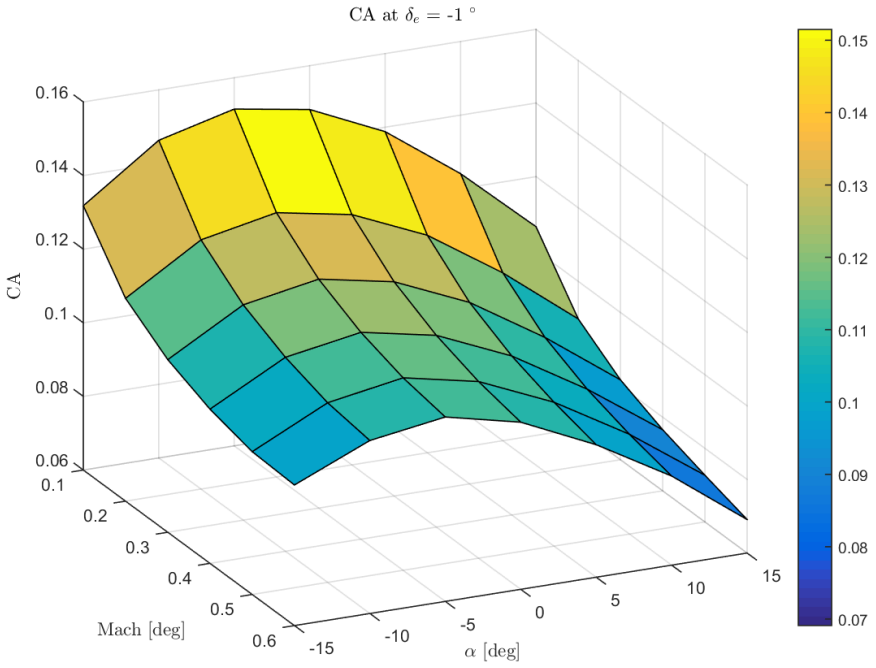

Figure A.15: C_A at δ_e = -1

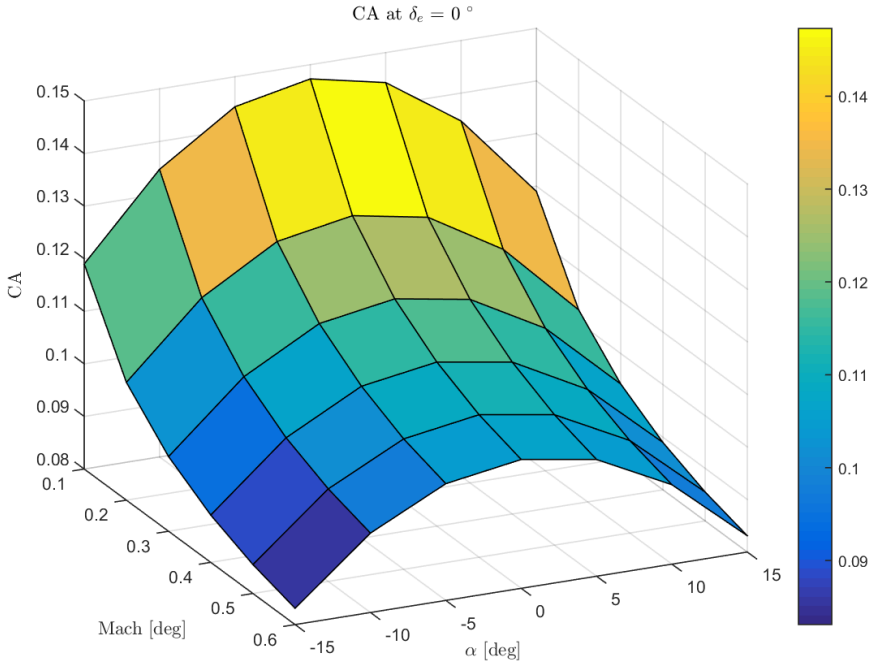

Figure A.16: C_A at $\delta_e = 0$

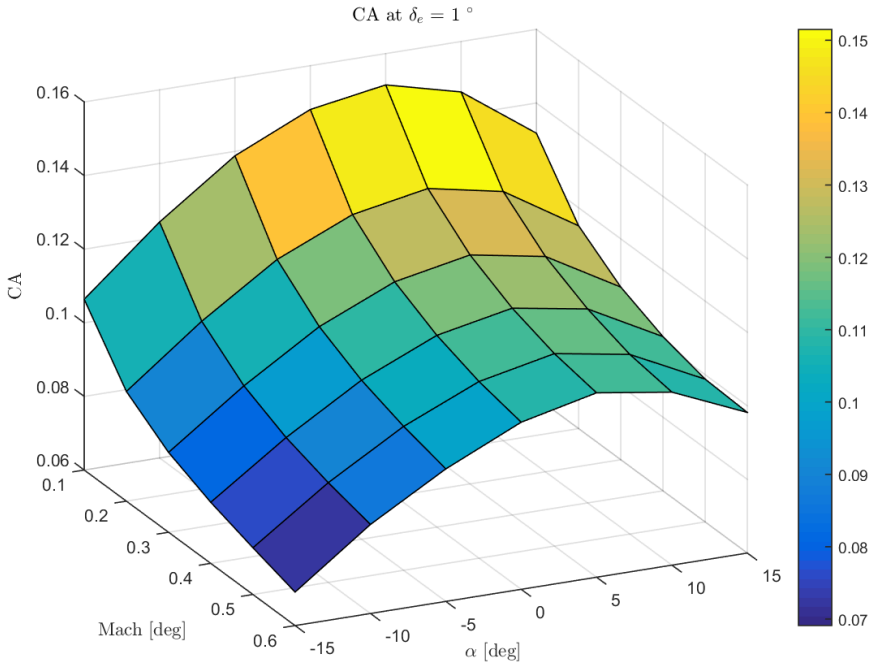

Figure A.17: C_A at $delta_e = 1$

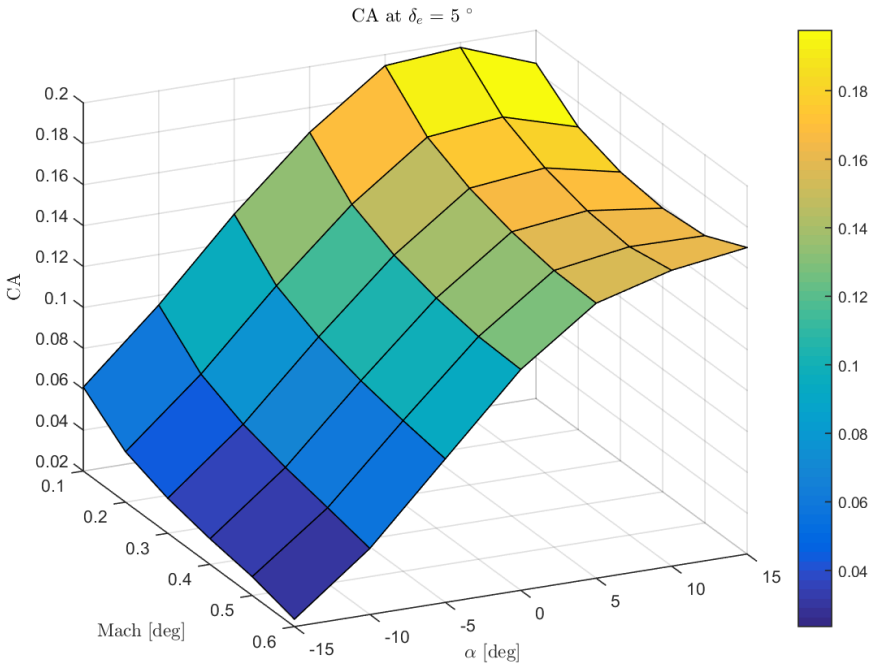

Figure A.18: C_A at $\delta_e = 5$

Appendix B

STATE VARIABLE MODELLING

$$\begin{bmatrix} X_u \\ X_\alpha \\ X_{\delta_e} \\ Z_u \\ Z_\alpha \\ Z_q \\ Z_{\delta_e} \\ M_u \\ M_\alpha \\ M_q \\ M_{\delta_e} \end{bmatrix} = \begin{bmatrix} -\frac{\bar{q}S(C_{D_u}+2C_D)}{mV_{P_1}} \\ -\frac{\bar{q}S(C_{D_\alpha}-C_L)}{m} \\ -\frac{\bar{q}SC_{D_{\delta_e}}}{m} \\ -\frac{\bar{q}S(C_{L_u}+2C_L)}{mV_{P_1}} \\ -\frac{\bar{q}S(C_{L_\alpha}+C_D)}{m} \\ -\frac{\bar{q}\bar{c}SC_{L_q}}{m} \\ -\frac{\bar{q}SC_{L_{\delta_e}}}{m} \\ \frac{\bar{q}S\bar{c}(C_{m_u}+2C_m)}{IyyV_{P_1}} \\ \frac{\bar{q}S\bar{c}C_{m_\alpha}}{Iyy} \\ \frac{\bar{q}S\bar{c}C_{m_q}}{Iyy}\frac{\bar{c}}{2V_{P_1}} \\ \frac{\bar{q}S\bar{c}C_{m_{\delta_e}}}{Iyy} \end{bmatrix} \tag{B.1}$$

$$\begin{bmatrix} X'_u \\ X'_\alpha \\ X'_q \\ X'_\theta \\ X'_{\delta_e} \\ Z'_u \\ Z'_\alpha \\ Z'_q \\ Z'_\theta \\ M'_u \\ M'_\alpha \\ M'_q \\ M'_\theta \end{bmatrix} = \begin{bmatrix} X_u \\ X_\alpha \\ 0 \\ -g\cos\Theta \\ X_{\delta_e} \\ \frac{Z_u}{V_{P_1}-Z_{\dot{\alpha}}} \\ \frac{Z_\alpha}{V_{P_1}-Z_{\dot{\alpha}}} \\ \frac{Z_q+V_{P_1}}{V_{P_1}-Z_{\dot{\alpha}}} \\ -\frac{g\sin\Theta}{V_{P_1}-Z_{\dot{\alpha}}} \\ M_{\dot{\alpha}}Z'_u + M_u \\ M_{\dot{\alpha}}Z'_\alpha + M_\alpha \\ M_{\dot{\alpha}}Z'_q + M_q \\ M_{\dot{\alpha}}Z'_\theta \end{bmatrix} \tag{B.2}$$