ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

FIGHTER PILOT BEHAVIOR CLONING AND TRANSFERRING TO ANOTHER AIRCRAFT

M.Sc. THESIS

Gülay SEVER

Department of Defense Technologies

Autonomous Systems Programme

DECEMBER 2022

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

FIGHTER PILOT BEHAVIOR CLONING AND TRANSFERRING TO ANOTHER AIRCRAFT

M.Sc. THESIS

Gülay SEVER
(514191012)

Department of Defense Technologies

Autonomous Systems Programme

Thesis Advisor: Assoc. Prof. Nazım Kemal Üre

DECEMBER 2022

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

SAVAŞ PİLOTU DAVRANIŞINI KLONLAMA VE FARKLI BİR HAVA ARACINA TRANSFERİ

YÜKSEK LİSANS TEZİ

Gülay SEVER
(514191012)

Savunma Teknolojileri Anabilim Dalı

Otonom Sistemler Programı

Tez Danışmanı: Assoc. Prof. Nazım Kemal Üre

ARALIK 2022

Gülay SEVER, a M.Sc. student of ITU Graduate School student ID 514191012 successfully defended the thesis entitled "FIGHTER PILOT BEHAVIOR CLONING AND TRANSFERRING TO ANOTHER AIRCRAFT", which she prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Assoc. Prof. Nazım Kemal Üre** Istanbul Technical University	.....
Jury Members :	**Prof. Dr. Mehmet Turan SÖYLEMEZ** Istanbul Technical University	.....
	Dr. Şükrü Akif ERTÜRK Turkish Aerospace Industries	.....
		.....

Date of Submission : **21 DECEMBER 2022**
Date of Defense : **30 DECEMBER 2022**

To Mert, my family and friends,

FOREWORD

I want to express my heartfelt gratitude to my advisor, Assoc. Prof. Nazım Kemal Üre, for his mentorship, honesty, and friendship. Not only have I had valuable research experiences under his supervision, but he has also played a significant role in shaping my academic journey. He is more than just a research advisor to me.

I also want to extend my thanks to my dear lab mates. It has been a pleasure to be a part of such a distinguished research group.I would like to send my special thanks to Abdullah Sadık Satır and Umut Demir, their friendship eased the workload of delivering such challenging projects.

I would like to express my sincere gratitude to my husband, Mert Sever, for his unwavering support throughout this entire process. It is hard to put into words how much credit he deserves for his support. Finally, I would like to thank my dear family and friends for their continuous support, love, and all the other things that they have done for me so far.

December 2022 Gülay SEVER

TABLE OF CONTENTS

ABBREVIATIONS

ANN	**:** Artificial Neural Network
BC	**:** Behavior Cloning
DAgger	**:** Data Aggregation
DNN	**:** Deep Neural Network
DOF	**:** Degree of Freedom
GNC	**:** Guidance Navigation and Control
IAS	**:** Indicated Airspeed
IL	**:** Imitation Learning
MSE	**:** Mean Square Error
NDI	**:** Nonlinear Dynamic Inversion
RL	**:** Reinforcement Learning
RRT	**:** Rapidly Exploring Random Tree
TL	**:** Transfer Learning
UAV	**:** Unmanned Aerial Vehicle

LIST OF TABLES

Page

LIST OF FIGURES

Page

FIGHTER PILOT BEHAVIOR CLONING
AND
TRANSFERRING TO ANOTHER AIRCRAFT

SUMMARY

The "Pilot-in-the-loop" flight simulators are important tools in the development of fighter aircraft because they allow engineers and designers to test different scenarios and algorithms in a controlled environment without the risks associated with actual flight testing. However, these simulations often require high pilot skill levels and can be time-consuming and costly to arrange. It is important to create realistic models of human fighter pilots in order to reduce the reliance on skilled pilots to demonstrate agile/aerobatic maneuvers in flight simulators.

Traditional controllers for aircraft require detailed knowledge about the aerodynamic model and physics of the aircraft in order to perform aerobatic maneuvers. Also, these control algorithms may not be able to match the performance of skilled human pilots, who are limited by a lack of bandwidth. This suggests that there may be potential for improving the performance of aircraft through the use of techniques that can take advantage of the superior speed and maneuverability of skilled pilots.

In that case, imitation learning is a potential solution to eliminate the dependency need of skillful pilots in the flight simulator. Imitation learning also known as learning from demonstrations has benefited from computational progresses brought on by deep learning and increased availability of demonstration data. It is aimed to emulate desired behavior in a given task. An agent is trained to learn mapping between observations and actions by utilizing demonstrations.

In this thesis aims the development of a pilot behavior model which is capable of autonomously performing agile maneuvers and is able to replace expert pilots' demonstrations over its full flight envelope in the flight simulator. Moreover, this model is transferable to other aircraft with limited data using transfer learning techniques. Besides all these features, the pilot behavior model can be able to run in real time in the flight simulator.

SAVAŞ PİLOTU DAVRANIŞINI KLONLAMA VE FARKLI BİR HAVA ARACINA TRANSFERİ

ÖZET

Pilotun döngüye dahil olduğu ("Pilot-in-the-loop") uçuş simülatörleri, savaş uçağı geliştirme sürecinin hayati bir parçasıdır. Yüksek güçlü bir savaş uçağının performans değerlendirmesinde üstün manevra kabiliyeti ve çeviklik giderek daha önemli hale gelmektedir. Bu performans ve emniyet hedeflerine ulaşmak için, uçuş simülatörleri gelecekteki uçak tasarımlarına ilişkin analizin ilk aşamalarından itibaren gerekli olacaktır. Yüksek doğruluklu uçuş simülatörleri, senaryoların ve algoritmaların, tam ölçekli insanlı hava aracı uçuş testiyle ilişkili riskler olmaksızın test edilebildiği kontrollü bir ortam sunar. Bununla birlikte, bu yeteneklerin analizi, uçağı kontrol etme yeteneklerini sınırlayan aşırı uçuş koşulları nedeniyle yüksek düzeyde pilot becerisi gerektirir. Ayrıca,modeli kalifiye pilotlar ile çoklu trim koşullarında farklı senaryoları test etmek için uçuş simülatöründe ayarlanaması gereken zaman ve bunun yüksek maliyeti nedeniyle mümkün olmayabilir. Yetenekli pilotların demonstrasyonlarına, yani uçuş simülatörlerindeki çevik/akrobatik manevralarına bağımlılığı ortadan kaldırmak için, kalifiye savaş pilotlarının gerçekçi modellemesi çok büyük önem taşımaktadır.

Geliştirilecek olan modelden beklenen temel sonuç, aynı uçak modeli için uzman bir pilotun gösterimleri ile uyumlu ve istenilen rotadan hafif sapmalara karşı dayanıklı bir yörünge üretmesidir. Problemin özellikleri düşünüldüğünde taklit öğrenme, yetenekli pilotların uçuş simülatörüne olan bağımlılık ihtiyacını ortadan kaldırmak için potansiyel bir çözümdür. Gösterilerden öğrenme olarak da bilinen taklit öğrenme, derin öğrenmenin getirdiği hesaplamalı ilerlemelerden ve gösteri verilerinin artan kullanılabilirliğinden yararlanmakatdır. Bu yöntem ile en temelde belirli bir görevde istenen davranışın taklit edilmesi amaçlanır. Tasarlanan model, gösterileri kullanarak gözlemler ve eylemler arasında eşleştirme yapmayı öğrenmek üzere eğitilir.
Optimum performans elde etmek için, eğitim verilerinin çok çeşitli durum ve aksiyonları içermesi gerekmektedir. Bu ister uzman pilotlar tarafından gerçekleştirilen az sayıda manevra gösterimi ile sağlanamaz. Ancak yörünge, çeşitli trim koşullarında uzman manevralarının açısal hızları (P, Q, R) kullanılarak yeniden üretilebilir. Pilotun açısal hızlarının yüksek doğrulukta doğrusal olmayan 6-DOF F-16 model simülasyonlarında takip edilmesi için uzman pilot yerine NDI kontrolörcüsü kullanılmış ve istenilen uçuş zarfındaki trim noktalarında simülasyonlar tekrarlanmıştır.
Bu veriler kullanılarak bir yapay bir sinir ağını, sensör durumları ve kontrol girdi değerleri arasındaki eşlemeyi öğrenerek uzman bir pilot gibi bir F-16'nın yörüngesini kontrol etmesi için eğiten bir denetimli öğrenme algoritması kullanılmıştır.

Split-S veri seti incelendiğinde manevranın uygulanması sırasında kullanılan kontrol girişlerinde yön ve büyüklük açısından hızlı ve ani değişiklikler gözlenmiştir. Ayrıca zaman serilerinin sayısı, uzunluğu ve aralarındaki korelasyonun da yüksek olduğu görülmektedir. Kullanılacak yapay sinir ağı modeli, verilerin tüm bu özelliklerini ele alabilmelidir. Tüm bunlar göze alınarak literatür incelenmiş ve yapay sinir ağı modeli olarak LSTM otomatik kodlayıcı (autoencoder) yapısı kullanılmıştır. Davranış klonlama yöntemleri, uzman veya insan gösterici tarafından gösterilen davranış ile model tarafından öğrenilen ve tipik olarak bir kayıp fonksiyonu olarak ifade edilen politika arasındaki farkın bir ölçüsünü gerektirir. Eğitimler sırasında, gösterilen davranışı olabildiğince yakın benzerlikte çoğaltmak için en yaygın seçim olan MSE kayıp fonksiyonu kullanılmıştır.
Simülasyon sonuçlarında eğitim sırasında model tarafından yapılan hataların kademeli olarak artarak kötü performansa yol açması nedeniyle tek başına davranış klonlamanın sorunlu olabileceği gösterilmektedir. Bunun nedeni, eğitim ve test veri setlerinin mutlaka aynı dağılımdan alınmaması ve bununda denetimli öğrenmede tipik olarak kullanılan bağımsız ve aynı şekilde dağıtılmış veri varsayımını ihlal etmesidir.
Etkileşimli gösterilerle davranış klonlamanın avantajı, yalnızca uzman tarafından sağlanan gösteri örneklerine güvenmek yerine modelin kendi hatalarından ders almasına izin vermesidir. Bu, modeli daha esnek ve uyarlanabilir hale getirir ve yeni durumlara veya ortamlara daha iyi genelleme yapabilir.
Literatürde bu bileşik hata sorununu çözmek için taklit öğrenmeyi etkileşimli denetimli öğrenmeye indirgeyen DAgger (Veri Toplama) adlı bir algoritma önerilmiştir. DAgger algoritması, bir uzman politikasından gelen girdiler ve çıktılarla etiketlenmiş yeni, değiştirilmiş eğitim verilerinin eklenmesini içeren bir veri toplama yöntemidir. Bu yaklaşımın amacı, ek uzman verileri dahil ederek bir NN modelinin performansını iyileştirmektir.
Ancak bizim problemimizde eğitim için gerçekte ihtiyaç duyulan şey daha fazla veri değil, istenen yörüngedeki sapmalardan kurtulmayı öğrenebileceği kritik verilerdir. Ayrıca eğitim sırasında yanlış eylemlerin kullanılması, eğitim performansı üzerinde olumsuz sonuçlar doğurabilir.
Bu tez kapsamında bu problemleri çözmek adına Limit-DAgger algoritması geliştirilmiştir. Bu algoritma ile ilkönce uzman bir pilotun dahi manevrayı düzeltemediği kritik *pqr* hata değeri bulunur. Bu kritik hata değerine ulaşana kadar NN model çıktıları sisteme uygulanır. Bu kritik değer aşıldığında uzman tarafından düzeltici aksiyonlar sisteme uygulanarak manevra gerçekleştirilir. Her itarsyonda pqr hata değeri yeniden hesaplanarak veri toplanır.
Simülasyon sonuçları ile doğrulandığı üzere bu metod ile 3 iterasyonda gerçek pilot performansına yakın bir NN modeli eğitilmiştir.

Bir sonraki aşamada eğitilen modelin başka bir hava aracı modeline limitli veri ile transfer edilmesi üzerine çalışılmıştır. Transfer öğrenimi, sınırlı miktarda eğitim verimiz olduğunda yararlı olabilecek test verileriyle bağımsız ve aynı şekilde dağıtılamayan eğitim verilerini kullanmamıza olanak tanır. Bu, transfer öğrenmeyi yetersiz eğitim verileri sorununu ele almak için uygun bir yaklaşım haline getirir. Bu çalışmada yalnızca son LSTM katmanı ve çıkışındaki yoğun katman eğitime açılmış ve düşük öğrenme oranı ile eğitimler başarı ile gerçekleştirilmiştir.

Özetle tez çalışmasında taklit öğrenme metodu kullanılarak, uçuş simülatöründeki tüm uçuş zarfı üzerinde uzman pilotların gösterilerinin yerini alabilen ve çevik manevraları otonom olarak gerçekleştirebilen bir pilot davranış modeli geliştirilmiştir. Ayrıca transfer öğrenme kullanılarak, belirli bir uçak tasarımı için bir model geliştirdikten sonra önceki modelleme sürecine göre daha az eforla farklı bir hava aracı modeline aktarılabilirlik sağlanmıştır. Tüm bu özelliklerin yanı sıra pilot davranış modeli, uçuş simülatöründe gerçek zamanlı olarak çalışabilecektir.

1. INTRODUCTION

The "Pilot-in-the-loop" flight simulators are a vital part of the development process of fighter aircraft [5]. In the performance evaluation of a high-thrust fighter aircraft, supreme maneuverability and agility are becoming increasingly important. In order to achieve these performance and safety goals, flight simulators will be required in earlier stages of analysis on future aircraft designs [6]. The high-fidelity pilot-in-the-loop simulators offer a controlled environment in which scenarios and algorithms can be tested without the risks associated with full-scale manned aircraft flight testing.

However, analyzing these capabilities requires high pilot skill levels due to the extreme flight conditions that limit their ability to control the aircraft. Moreover, arranging a time for qualified pilots in a flight simulator may not be practicable because of the time needed to test a wide range of scenarios under various trim conditions and high-cost [5,7]. In order to eliminate the dependency on skillful pilots' demonstrations i.e., aerobatic maneuvers in the flight simulators, realistic modeling of the human fighter pilots is crucial.

The expected outcome of this model is that produce a trajectory that will be aligned with an expert pilot's demonstrations for the same aircraft model and robust against slight deviations from the desired course. After developing a model for a specific aircraft design, the next issue will be transferability to a different aircraft model with less effort compared to the previous modeling process. Besides all these features, the pilot behavior model should be able to run in real time in the flight simulator.

1.1 Previous Studies

The pilot behavior model computes the cockpit joystick inputs of aircraft which determines required actuator values to perform aerobatic maneuvers as executed by controllers. Therefore, the current state of the art in autonomously controlling an

aircraft, ranging from traditional systems to neural network-based systems is examined in this section.

1.1.1 Traditional control based methods

A physics-based autonomous flight controller that uses the aerodynamic equations of an aircraft is sufficient for simple tasks such as maintaining altitude or level flight. When it comes to controlling an aircraft in agile flight (such as performing aerobatic and combat maneuvers), the complexity of control and the accuracy needed for the model quickly rises.

In [8], a control policy is developed by employing feedback and feed-forward control techniques simultaneously for position, attitude tracking and speed,altitude regulating for fixed-wing UAV. Knife-edge maneuver which interacts with a RRT based motion planner is simulated. This paper [9] presents a methodology for automating agile maneuvers for a small fixed-wing unmanned aerial vehicle. The approach involves solving optimal control problems off-line to generate a set of reference trajectories and feedforward control inputs and using a dynamic time warping-based interpolation process to add robustness to the maneuver. Feedback control laws are then used to stabilize the aircraft about the reference trajectory. The methodology is demonstrated on a variety of maneuvers, including an aggressive turn-around and transition maneuvers between straight and level flight. The approach is computationally efficient due to the off-line calculations. In these studies [8,9], feedforward and feedback controller were used together.

Bulka E. and Nahon M. [10,11] presented a physics based single control system to perform agile maneuvers with a fixed-wing UAV. The single control system discussed is modular, meaning that it is made up of individual sub-components that can be tuned separately. The maneuver generator produces reference attitude and position values, which are then used by the position controller to adjust the aircraft's orientation and reduce errors in its position. The attitude controller then ensures that the aircraft maintains this new orientation. This modular design simplifies the tuning process. The control system described can be used to perform a range of aerobatic maneuvers, including knife-edge, rolling Harrier, hover, and aggressive turnaround. It can also

handle the transitions between these maneuvers.

In this study [2], the maneuvers are obtained by recording pilot inputs and corresponding system responses during an actual flight.A hybrid method that consists of dividing the complex maneuvers' trajectories into time-parameterized curves was used.Such maneuvers, however, have to be hand-engineered and are thus not very dynamic.

Although it is uncommon, there is some research that uses control laws for autonomous air combat. In this work Ure and Inalhan [1,12], proposed a multi-modal framework that covers the full range of flight dynamics for an aircraft. The multi-modal control framework decomposes arbitrary maneuvers into a set of maneuver modes 1.1 and corresponding parameters. This approach has several advantages. One is that it allows the feasibility and control problems to be addressed individually for each sub-maneuver, rather than the entire maneuver, reducing the complexity of the problem. Additionally, the control part is simplified by designing specific controllers for each mode and transitions between them, which enables the system to track the sequence of maneuver modes rather than a single controller for the entire flight envelope.

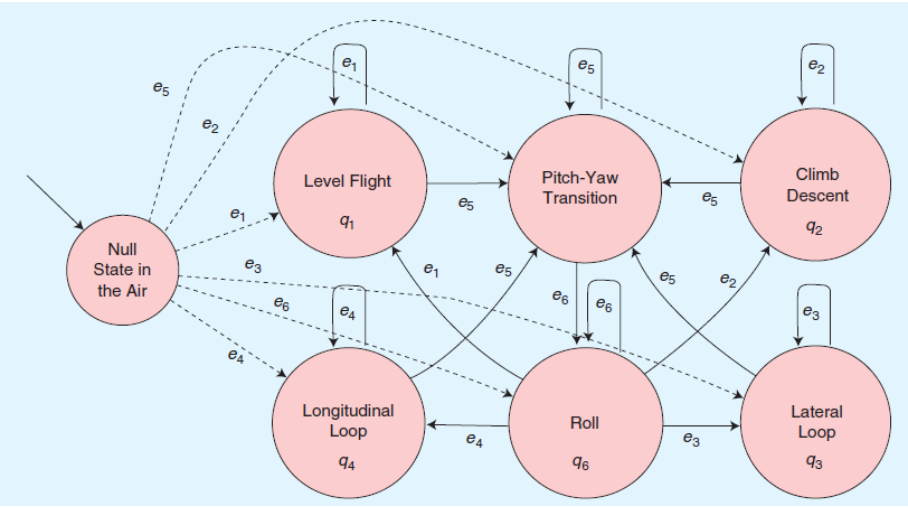


Figure 1.1 : The maneuver modes with transition diagrams [1]

A switching nonlinear control scheme which is based on sliding mode and higher-sliding mode control was introduced. The effectiveness of the control design

was demonstrated through the execution of aerobatic maneuvers, as well as by integrating it with a high-level flight path planner (rapidly-exploring random tree (RRT)) in an urban environment.

1.1.2 Learning based methods

In the previous section, we showed that a traditional controller has to have high detail a priory information about the aerodynamic model and physics of the aircraft especially in order to perform an aerobatic maneuver. Moreover, traditional control algorithms put limits on performance that aren't even close to what an experienced human pilot can do. Although skilled fighter pilots are superior in terms of speed and maneuverability to conventional automatic control systems, they are limited by a shortage of bandwidth 1.2.

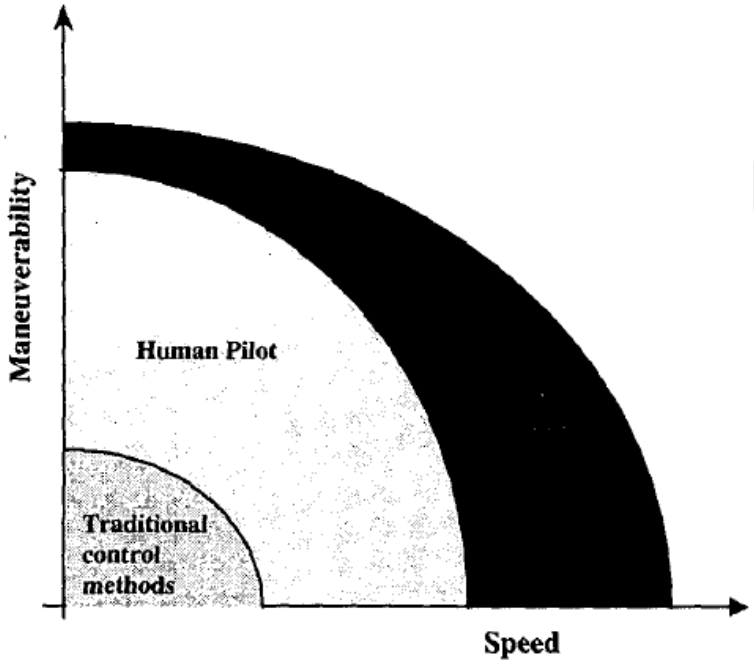


Figure 1.2 : Traditional control performance vs human pilot performance [2]

Main advantage of using learning-based techniques is that it allows for the creation of a model that can replicate the actions of an expert, potentially leading to improved performance. This can be especially useful in situations where it is difficult or impossible to explicitly program the desired behavior such as fighter pilot behavior modeling.

In recent literature ,there has been a rapid increase in the number of studies on this topic.

Shukla et al. [13] proposed a technique for training an artificial neural network (ANN) model to improve its robustness. The authors use the Dataset Aggregation (DAgger) algorithm to introduce perturbed data to the training process, which allows the ANN to maintain its accuracy in the presence of unvisited states. The ANN's output is used to control a fixed-wing aircraft in a simulator, and any deviations from the desired behavior are labeled by an expert policy and added to the training dataset. The authors already had a Guidance Navigation Control (GNC) system that worked as an expert to enabling for faster data generation. The article presents a method for controlling the aircraft to following a square flight track defined by four waypoints using the ANN. The inputs to the ANN include the North-East-Down (NED) distances for each waypoint, the velocity, roll angle, and pitch angle, and the output is the throttle, elevator, aileron, and rudder values. The results show that standard supervised learning techniques are not effective in training the ANN to fly the aircraft through the defined path. Instead, the authors develop a sophisticated algorithm known as "Moving-Window DAgger" that yields successful results.

The goal of this dissertation [14] is to develop an automatic pilot that can imitate the behavior of a real pilot by training a model to perform complex aerobatic maneuvers. The Immelmann turn maneuver with a parameterized target altitude is used to test the capabilities of the learned model. To address this problem, the authors use a behavior cloning approach, where the model learns from the demonstrations of a pilot. This allows the model to imitate the pilot's behavior and perform the desired aerobatic maneuvers. The relevant data is collected using a plugin for Microsoft Flight Simulator (FSX), and the results show that the LSTM model is able to perform the Immelmann turn to any reasonable desired altitude. The trajectory produced by the model is smooth and does not exhibit the oscillations commonly seen in human pilots.

In this work [15,16], a feed-forward neural network that is trained on a limited amount of data collected from human fighter pilots is presented. This data was recorded as the

pilots maneuvered a fixed-wing fighter aircraft in a flight simulator. The goal of the work is modeling the behavior of the pilots by using a behavior cloning technique.

This works [15,16] presents an approach to modeling human pilot behavior using a feed-forward neural network and transferring it to a different model.Unlike previous work [13], this approach uses data collected from human pilots instead of data from an autopilot, and it focuses on capturing human-like flight behavior rather than skilled pilot maneuvers. Additionally, the input state vector has been designed to be direction invariant, which allows the model to perform well with a smaller amount of training data. It is also shown that a pre-trained neural network requires less training than an untrained model for adapting to changes in flight dynamics.These results suggest that transfer learning with pre-trained models could be useful for quickly developing AI-powered aircraft for a wide range of applications.

However, the robustness of the model in the presence of unvisited states has not been discussed in these studies [14]–[16]. This is an important consideration, as the accuracy of behavior cloning models can decrease when they encounter states that were not present in the training data. It is not clear whether the model developed in this works would be able to maintain its accuracy in such situations.

1.2 Contributions

Although there are many studies on the pilot behavior modeling concept, not all of the desired features can be achieved. The robustness of the model in the presence of unvisited states has not been discussed in these studies [14]–[16].This is an important consideration, as the accuracy of behavior cloning models can decrease when they encounter states that were not present in the training data. It is not clear whether the model developed in this works would be able to maintain its accuracy in such situations. Although,it is interesting to explore the potential for trained policies to be applied in other flight simulators [15,16], there is not much more work on transfer training for aerobatic maneuvers.

These issues are addressed as follows:

- **Robustness of Model:** The DAgger algorithm is a method for aggregating data that involves adding new, modified training data that is labeled with inputs and outputs from an expert policy to deal with the cascading and compounding error problems in sequential decision-making. This algorithm makes the model more flexible and adaptable, and able to generalize better to new situations or environments. In this thesis, DAgger was further developed and the Limit-DAgger algorithm is presented to select the most appropriate demonstration data for training, and the efficiency of the training process was improved.

- **Transferring to Another Model with Limited Data:** Transferring the pre-trained model reduces the amount of time and resources required to train a model on the target task. Transfer learning allows us to use training data that may not be independent and identically distributed with the test data, which can be useful when we have a limited amount of training data. This makes transfer learning a promising approach for addressing the issue of insufficient training data.

2. BACKGROUND

2.1 Training Environment

Overall, the interaction between the expert pilot and the flight simulator involves the pilot taking actions based on its current state, and the simulator responding to these actions by transitioning to a new state.

In this thesis, A high-fidelity 6-DOF F-16 model has been used as a simulator flight dynamic model, and NDI as an expert to produce control commands for the desired maneuver.

2.1.1 F-16 model

F-16 is a single-engine, supersonic fighter aircraft. It is equipped with several control surfaces that allow the pilot to maneuver the aircraft. The primary control surfaces include the elevators, ailerons, and rudder. The elevators are located on the tail section of the aircraft and control the pitch, or the up-and-down movement of the nose. The ailerons are located on the wingtips and control the roll, or the tilting of the aircraft from side to side. The rudder is located on the vertical stabilizer at the back of the aircraft and controls the yaw, or the left-and-right movement of the nose. By using these control surfaces together, the pilot can maneuver the F-16 in a variety of ways, including turning, climbing, diving, and rolling.

In this thesis, a high-fidelity 6-degree-of-freedom model of an F-16 aircraft was used. More information about the model can be found in reference [17]. The model includes control surface actuators, leading edge flaps, and an extensive database of aerodynamic forces and moments. The aerodynamic data has been modified to allow for the use of independent and asymmetric actuators.

The state vector of the model is given in Equation 2.1. V_t, α, β are respectively magnitude of velocity, angle of attack and side-slip angles. ϕ, θ, ψ are the Euler angles.

P, Q, R are the body axis angular rates. p_n, p_e, p_d are the tangential plane coordinates of the aircraft.

$$x = [V_t, \alpha, \beta, \phi, \theta, \psi, P, Q, R, p_n, p_e, p_d]^T \in \mathbb{R}^{12} \tag{2.1}$$

The control input vector of the model and threshold values are given as:

$$u = [\delta_T, \delta_{ele}, \delta_{ail}, \delta_{rud}] \tag{2.2}$$

$$0 \leq \delta_T \leq 1$$
$$-25^\circ \leq \delta_{ele} \leq 25^\circ$$
$$-21.5^\circ \leq \delta_{ail} \leq 21.5^\circ$$
$$-30^\circ \leq \delta_{rud} \leq 30^\circ$$

where δ_T is throttle position, δ_{ele} is elevator deflection, δ_{ail} is aileron deflection, δ_{rud} is the rudder deflection. Then, in equation 2.3, nonlinear F-16 model dynamics can be written as:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) \tag{2.3}$$

2.1.2 NDI controller

NDI, or nonlinear dynamic inversion, control is often used in applications where the dynamics of the system are complex and cannot be accurately modeled using linear techniques. The key idea behind NDI control is to invert the nonlinear dynamics of the system, which allows the controller to directly calculate the control inputs needed to achieve the desired behavior.

In our simulation studies, the NDI is used for angular rate control 2.1.

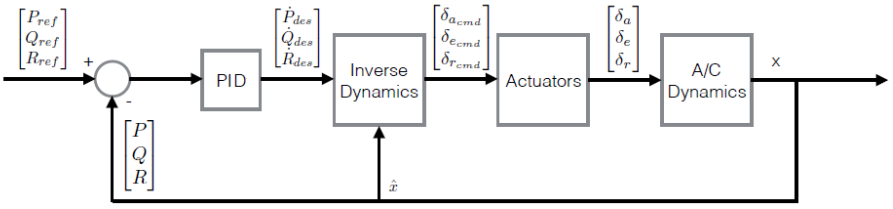


Figure 2.1 : Block diagram of the NDI controller

2.2 Split-S Maneuver

The Split-S is a maneuver used by fighter aircraft, such as the F-16, to quickly reverse the direction of flight. It involves rolling the aircraft inverted and then pulling back on the control stick to cause the aircraft to dive towards the ground. As the aircraft reaches the bottom of the dive, the pilot levels out and begins climbing back up in the opposite direction. This maneuver is used to quickly reverse the aircraft's course and can be useful in air-to-air combat or to evade an enemy missile. It is called a "Split-S" because the aircraft's flight path resembles the letter "S" when viewed from above. Split-S maneuver segments are defined as follows 2.2;

- Null State
- Level-Flight
- Pitch-Yaw Transition
- 180-Roll
- Longitudinal Loop

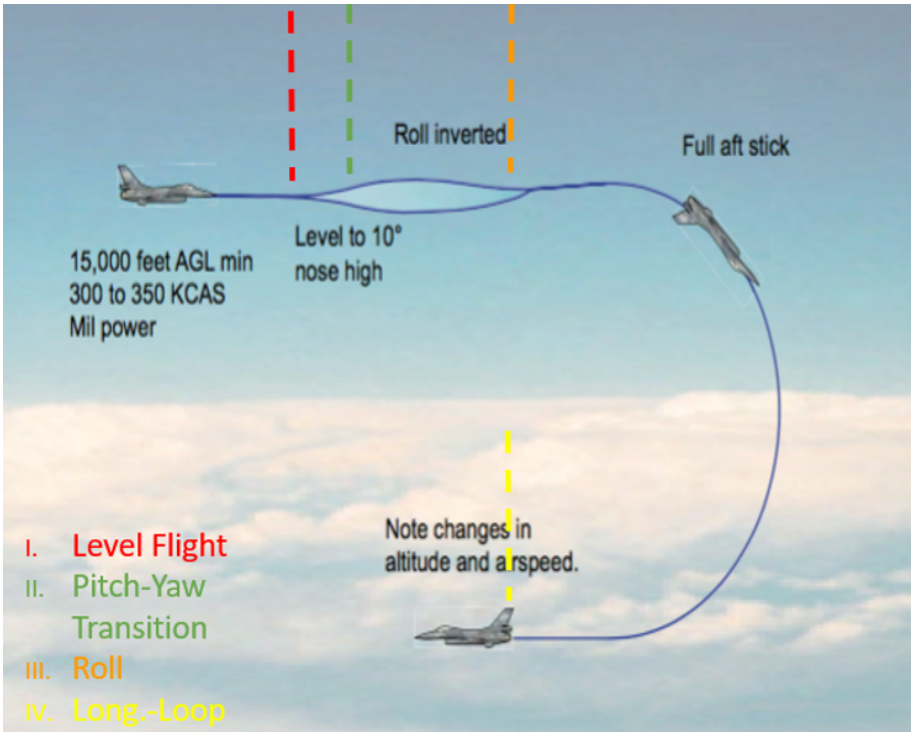


Figure 2.2 : Split-S maneuver

Vertical maneuverability is crucial for performance assessment and design for a high-trust fighter aircraft [18] and split-S is a standard vertical turning maneuver. For the analysis in the flight simulator, it is very important to have a model that can do the Split-S maneuver. Due to these reasons, the Split-S maneuver has been chosen to test the learned model's capabilities.

2.3 Neural Network Models for Time Series Predictions

There are several types of neural network models that can be used for time series predictions, including:

1. Recurrent neural networks (RNNs), which are specifically designed to work with time series data and are able to capture the dependencies between observations in the data.
2. Convolutional neural networks (CNNs), which can be used to identify local patterns in time series data and are often used in combination with RNNs.
3. Long short-term memory (LSTM) networks, which are a type of RNN that is able to learn long-term dependencies in time series data.
4. Autoregressive integrated moving average (ARIMA) models, which are a class of statistical models that are commonly used for time series prediction

Overall, the best model to use for time series prediction will depend on the specific characteristics of the data and the prediction task at hand. In some cases, a simple RNN or ARIMA model may be sufficient, while in other cases, a more complex model like an LSTM network may be necessary.
In this thesis, we aim to develop a model that can perform aerobatic maneuvers, and we preferred to use LSTM models, as used by the studies in the literature [14,19].

2.3.1 LSTM models

Long short-term memory (LSTM) is a type of recurrent neural network (RNN) that is well-suited to modeling time series data [20]. It is called "long short-term memory"

because it can remember information for long periods of time, while also being able to capture short-term dependencies in the data.

LSTM networks are composed of cells that operate on input sequences and maintain an internal state that can be passed from one cell to another. These cells are equipped with gates that can control the flow of information in and out of the cell, allowing the network to selectively remember or forget information from the past. This ability to retain past information and selectively update the internal state makes LSTM networks well-suited for time series prediction. A single LSTM cell [1] structure is given in Figure 2.3:

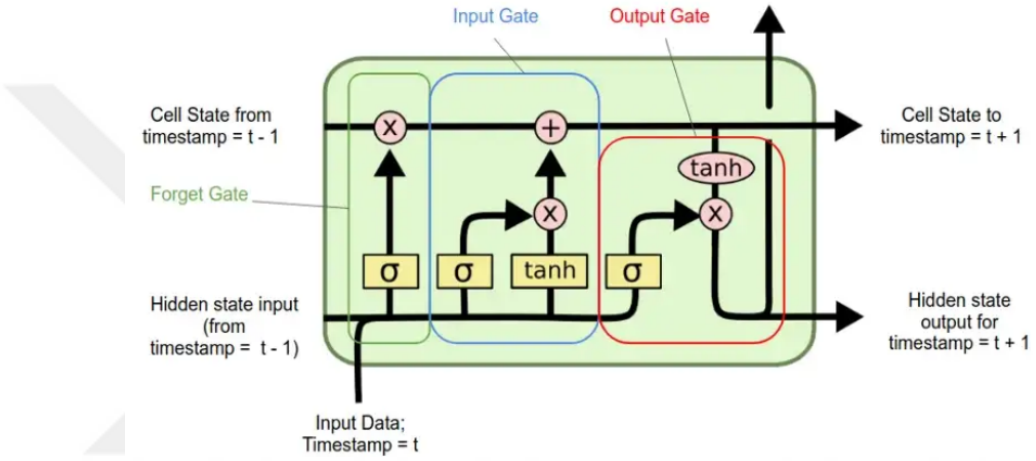


Figure 2.3 : A single LSTM cell

To use an LSTM network for time series prediction, the time series data must first be transformed into a suitable format for the network. This typically involves splitting the time series into subsequences of fixed length, called windows, and framing the problem as one of predicting the next value in the time series given the values in the current window. The LSTM network can then be trained using these windows of data as input and the corresponding future time series values as the desired output. After training, the network can be used to make predictions by providing it with a window of time series values and having it output a prediction for the next value in the sequence.

[1] https://www.turing.com/kb/comprehensive-guide-to-lstm-rnn

2.3.2 Auto-Encoders

An autoencoder is a type of neural network that is used to learn a compressed representation of data, often for the purpose of dimensionality reduction or feature learning. Autoencoders are composed of two parts: an encoder and a decoder. The encoder takes the input data and maps it to a lower-dimensional representation, called the latent representation or code. The decoder then takes this latent representation and attempts to reconstruct the original input data. The illustration of the network is given in Fig 2.4

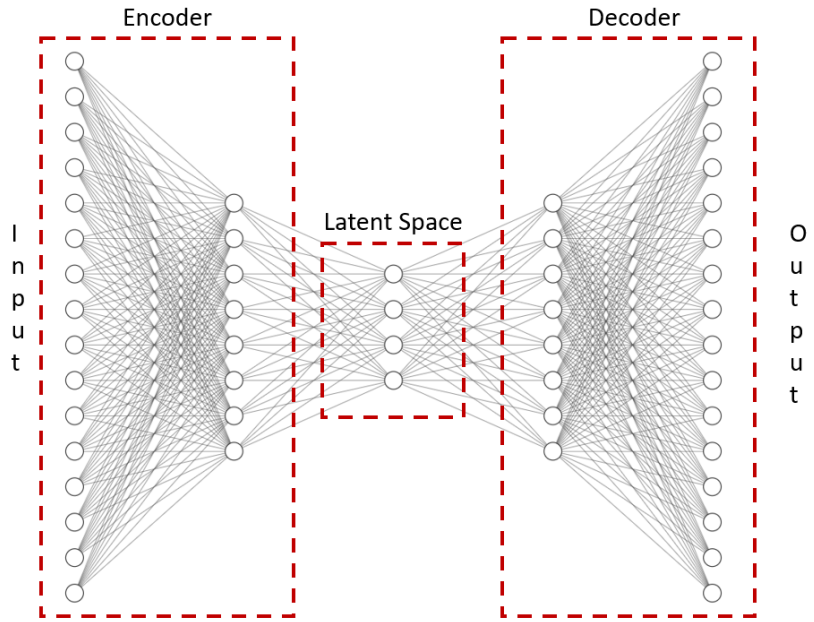


Figure 2.4 : Encoder-Decoder illustration

The goal of training an autoencoder is to learn a representation (encoding) of the input data that captures the most important information in the data while losing as little information as possible. This is typically done by minimizing the reconstruction error between the original input data and the output of the decoder.

Autoencoders can be used for unsupervised learning tasks, such as anomaly detection and data denoising, and can also be used to improve the performance of other machine learning models. The biggest challenge in working with autoencoders is getting the model to learn a meaningful and generalizable latent representation of the data.

2.3.3 LSTM autoencoders

An LSTM autoencoder is a type of autoencoder that uses a Long Short-Term Memory (LSTM) neural network as the encoder and decoder portion of the network. In the case of an LSTM autoencoder, the LSTM network is used to encode the input data into a lower-dimensional representation, and then the decoder LSTM network is used to reconstruct the original input data from this encoded representation. This allows the LSTM autoencoder to capture the most important information from the input data and reduce the dimensionality of the data while preserving the important information.

A prediction LSTM autoencoder combines these two architectures by using an LSTM network as the encoder and a feedforward network as the decoder. The LSTM encoder processes the input sequence and produces a fixed-length latent representation, which is then passed to the feedforward decoder to generate a predicted output sequence. This architecture can be used for tasks such as sequence-to-sequence prediction, where the goal is to predict the next element in a sequence given the previous elements.

In our previous work [3], LSTM autoencoder 2.5 based target trajectory predictions for agile attack patterns are presented. We demonstrated that deploying such networks for targets capable of agile mobility is in fact more effective than alternative methods.

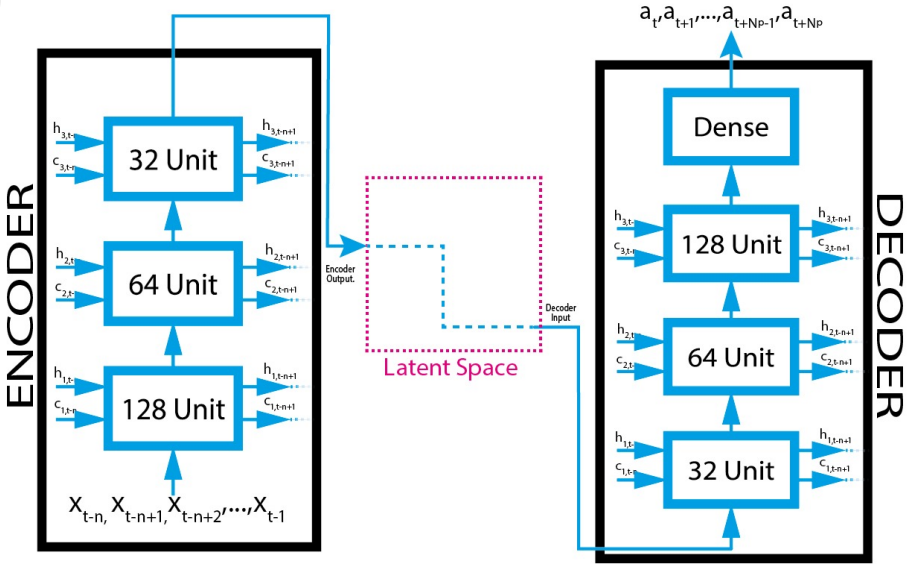


Figure 2.5 : LSTM autoencoder for sequence prediction [3]

2.3.4 LSTM model architecture for extreme event prediction

In the context of extreme event prediction, an LSTM model might be trained to forecast

the likelihood of rare or unusual events occurring in a given time series. For example, an LSTM model might be used to predict the occurrence of extreme weather events, financial market crashes, or cyber attacks.

A recent study [4] at Uber AI Labs showed how the automatic feature learning capabilities of Long Short-Term Memory (LSTM) networks, combined with their ability to process input sequences, can be used in an end-to-end model for forecasting driver demand during rare events like public holidays. This demonstrates the potential of LSTM networks as a powerful tool for demand forecasting in situations where data is scarce or irregular. Model illustration is given in Figure 2.6.

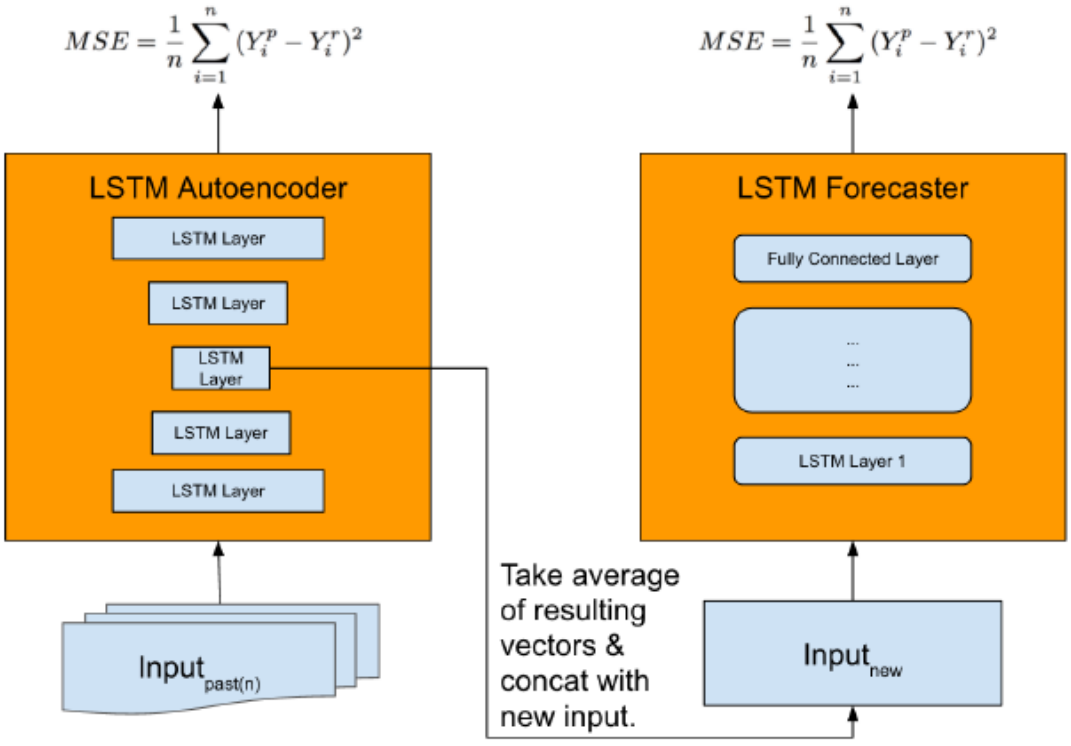


Figure 2.6 : Feature extraction and forecast model [4]

3. IMITATION LEARNING

Deep learning techniques have been widely applied, achieving state-of-the-art results in various fields of study. Learning control policies with deep networks is mainly discussed on two paradigms: Deep Reinforcement Learning (DRL) and Imitation Learning(IL). DRL relying on learning from trial and error which requires significant amount of system interaction time. Moreover, designing a proper reward structure is not straightforward in complex scenarios. Imitation learning is a powerful and practical alternative to DRL for learning control policies by replacing hand-designed reward functions with expert's demonstrations [21].
In recent years, imitation learning has been investigated as a way to efficiently and intuitively program autonomous behavior [22,23]. Imitation learning also known as learning from demonstrations has benefited from computational progresses brought on by deep learning and increased availability of demonstration data. It is aimed to emulate desired behavior in a given task. An agent is trained to learn mapping between observations and actions by utilizing demonstrations [24].

This study aims the development of a pilot behavior model which is capable of autonomously performing agile maneuvers over its full flight envelope and is able to replace expert pilots' demonstrations.

This section is organized by dividing imitation learning into behavior cloning (BC) and behavior cloning via interactive demonstrations.

3.1 Behaviour Cloning

BC is implemented as a first step of "BC with Interactive Demonstrations".

Behavior cloning is a type of machine learning technique that involves training a model to replicate the behavior of an expert or a human demonstrator. It is often used to teach a robot or an artificial intelligence (AI) system to perform a task by learning from a human expert or from a dataset of demonstration examples.

One advantage of behavior cloning is that it allows a robot or an AI system to learn a new task quickly and efficiently, without the need for extensive programming or manual intervention. It can also be used to teach a robot or an AI system to perform tasks that are difficult or impossible for a human to teach manually, such as tasks that require precise timing or coordination. In this step, Split-S maneuvers executed by NDI in various trim conditions are used in BC as supervised learning.

3.1.1 Data generation

To achieve optimal performance, the training data must include a wide range of states and actions. This cannot be achieved with a small number of maneuver demonstrations by expert pilots. However, the trajectory can be reproduced using angular rates (P, Q, R) of expert maneuvers in various trim conditions. The "expert" NDI controller was used for tracking of pilot's angular rates on high fidelity nonlinear 6-DOF F-16 model simulations. Data generation strategy for behavior cloning is represented in Fig3.1.

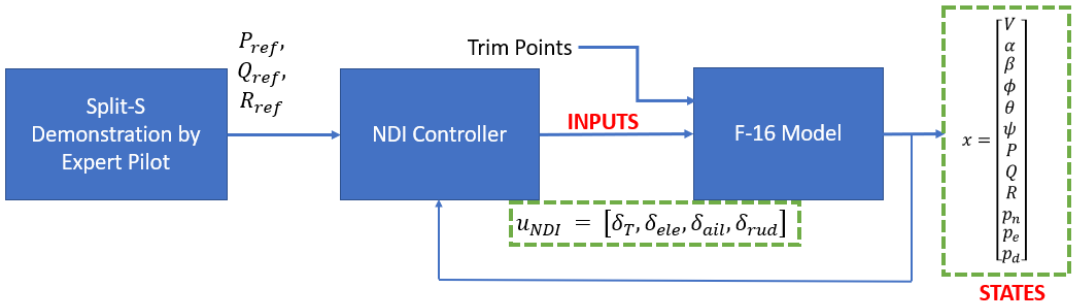


Figure 3.1 : Data generation flow for behavior cloning

Dataset obtained from;

- Velocity range [600; 900] ft/s with the 50 ft/step
- Altitude range of [10000; 20000] ft with the 500 ft step

The training dataset consists of 56 simulations, each with 705 samples and 18 states, total sample number reaches 39480.

Actuator commands for various trim conditions are shown in Fig3.2.

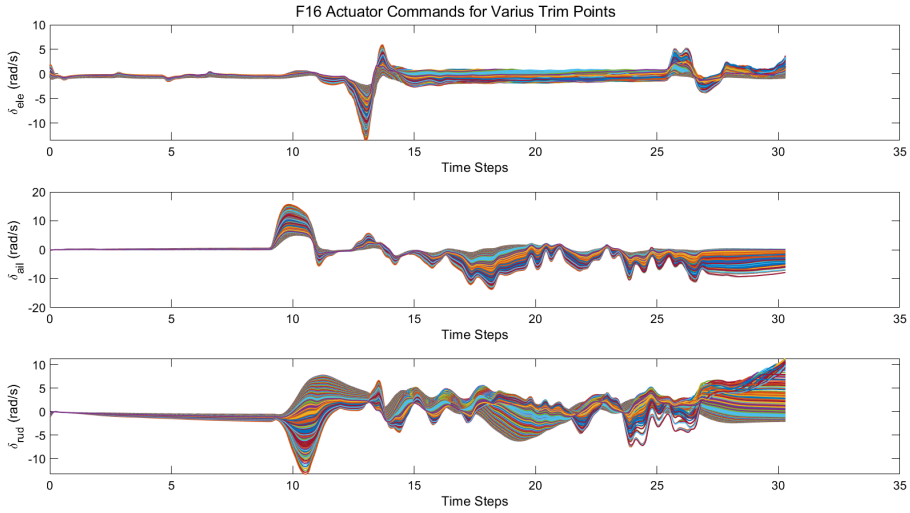


Figure 3.2 : Actuator commands for various trim conditions

Angular rates for various trim conditions are shown in Fig.3.3

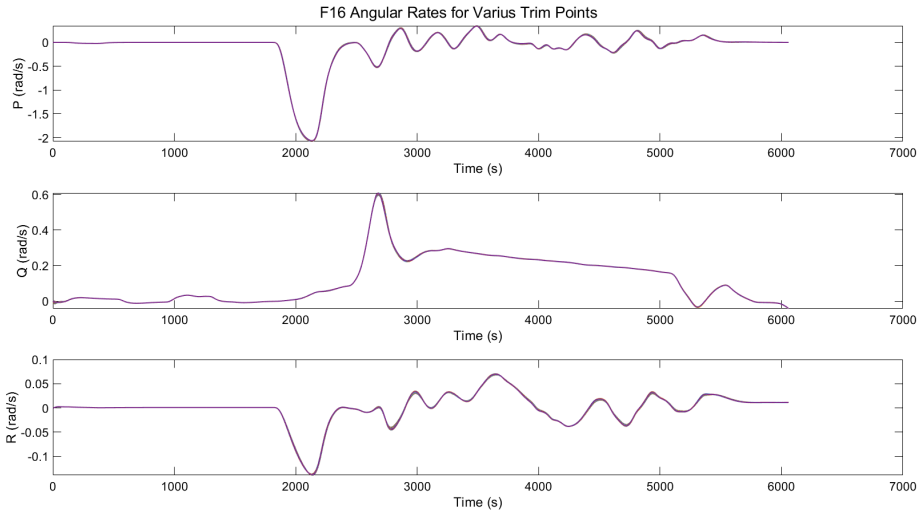


Figure 3.3 : Angular rates for various trim conditions

3.1.2 Proposed NN architecture

A supervised learning algorithm that trains an artificial neural network to control a F-16's trajectory as an expert pilot by learning a direct mapping between sensor states and control input values/actuator commands is required.

Agile maneuvers, such as Split-S, require rapid and significant changes in the control inputs used to execute them.

In order to show these rapid changes, control inputs of Split-S maneuver at 15000ft and 750ft/sec are given in Figure 3.4, and rapid peaks are highlighted with a red cylinders.

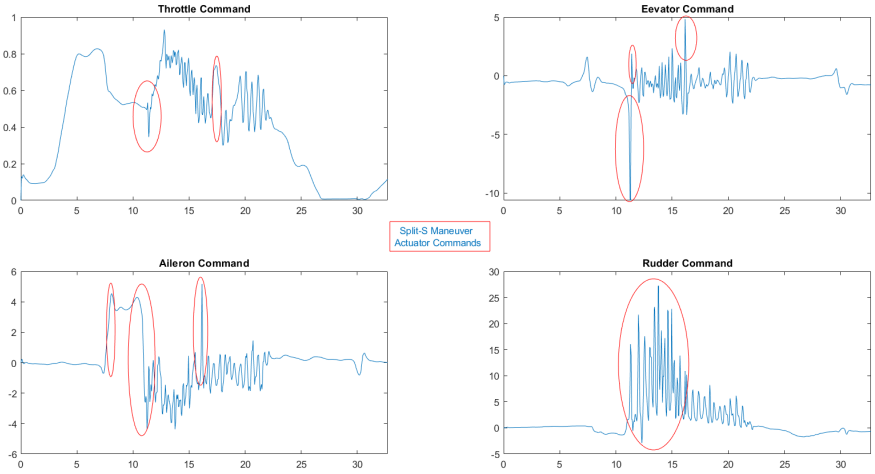


Figure 3.4 : Split-S actuator commands for 15000 ft,750 ft/sec

When the training set is analyzed, it is seen that the number and length of time series and correlation among them are also high.

The network model to be used should be able to deal with all these characteristics of the data.

In that point, the study entitled as"Time-series Extreme Event Forecasting with Neural Networks at Uber" has been influential for us [4]. Accurate prediction of completed trips during special/rare events(e.g., holidays) at Uber can lead to more efficient allocation of drivers, resulting in shorter wait times for riders.

The problem of accurately predicting rare events discussed in [4] is very similar to the problem of predicting rapid changes in control commands applied during aerobatic maneuvers. Also, the number and length of time series and correlation among them

are high in both dataset. As a result, composite LSTM Autoencoder for sequence reconstruction and prediction is used in our case. Details are given in Section2.3.4.

Training network model showed in Figure 3.5:

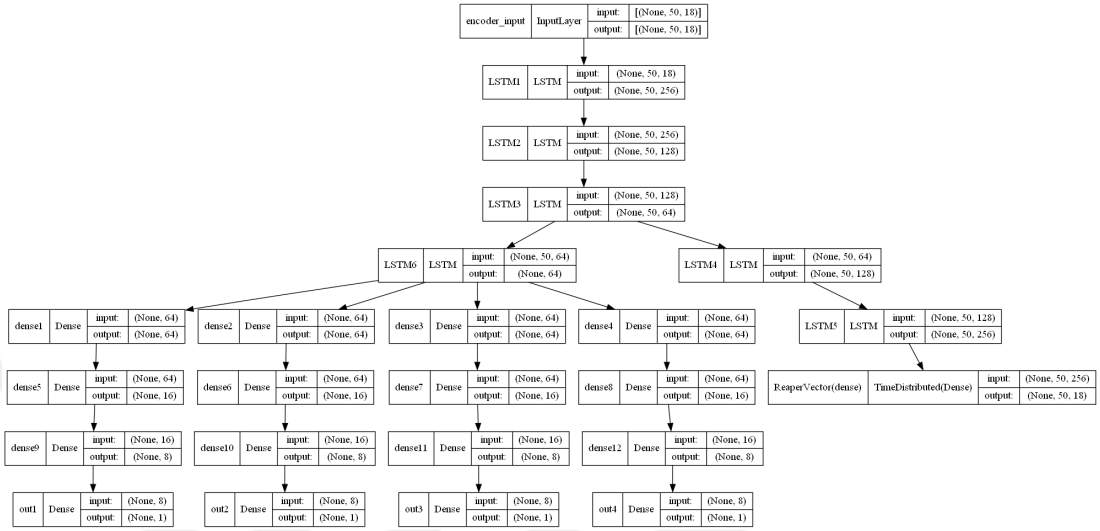


Figure 3.5 : Composite LSTM autoencoder for sequence reconstruction and prediction

3.1.3 Training processes and results

Training process of supervised learning summarised as follows:

- The raw simulation data collected as;

  $x_{training} = [V_t, \alpha, \beta, \phi, \theta, \psi, P, Q, R, p_n, p_e, p_d, a_n, \delta_T, \delta_{ele}, \delta_{ail}, \delta_{rud}, t] \in \mathbb{R}^{18}$

- The data is normalized to [0,1] interval

- Sampled by using sliding window method (lookback size =50 samples)

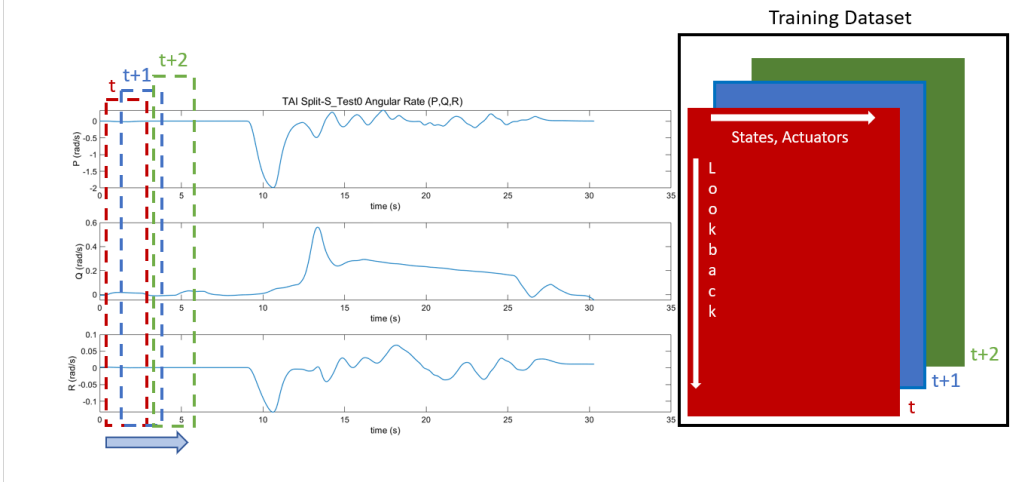
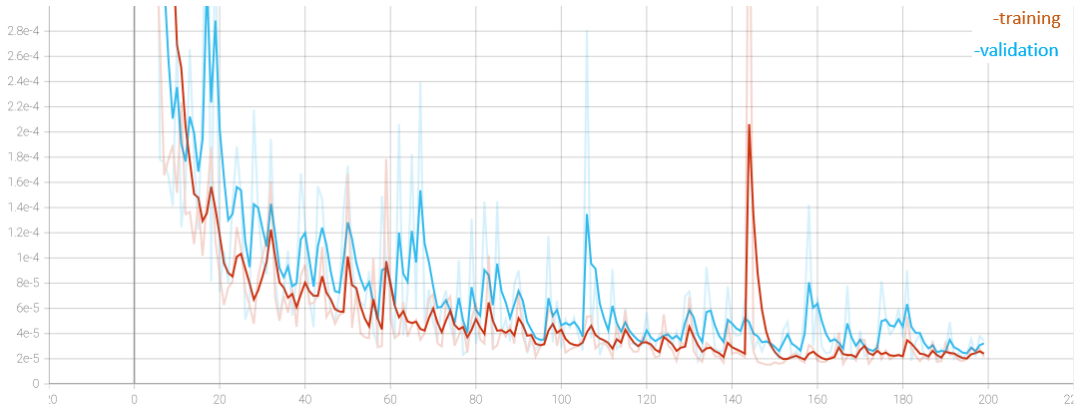


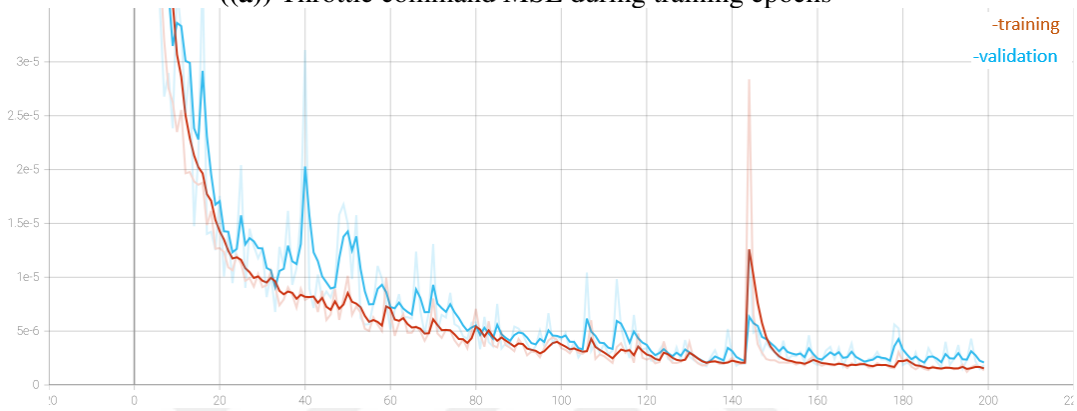
Figure 3.6 : Batch generation for training

- Training data partitioned as; %70 for training, %25 for validation, %5 for test.

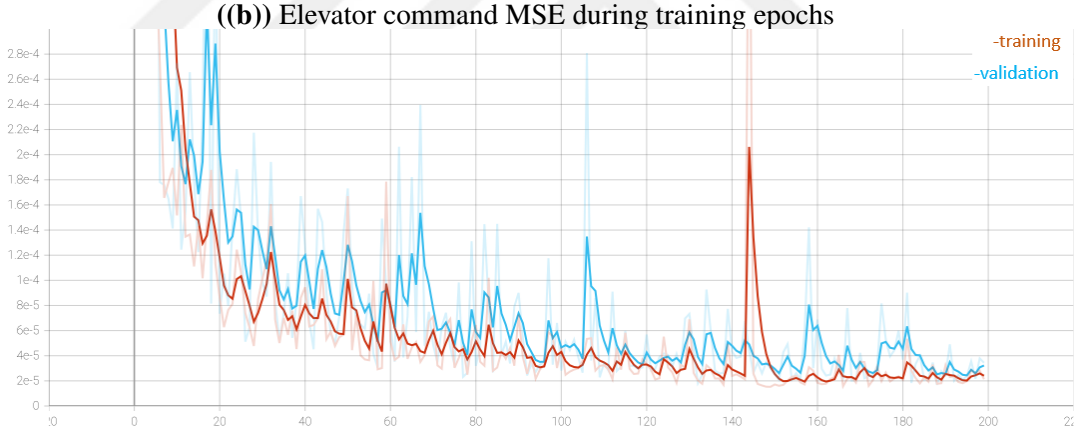
Behavior cloning methods require a measure of the difference between the behavior demonstrated by the expert or human demonstrator and the policy learned by the model, typically expressed as a loss function. The MSE loss function is the most common choice for replicating the demonstrated behavior as closely as possible. The belief is that the more the maneuver produced by the deep control policies resembles a real pilot maneuver, the more it mimics behavior in the operational environment. The MSE losses for actuator commands during training epochs given in Figure 3.7.



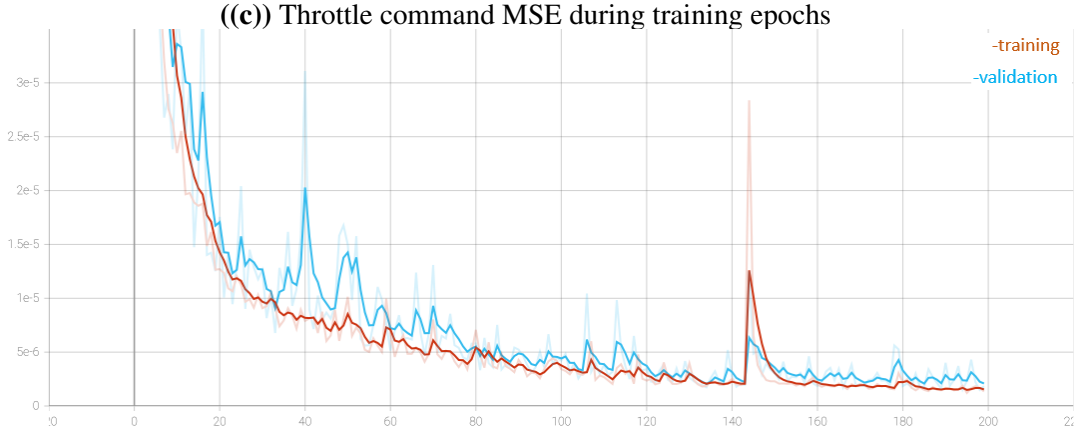
((a)) Throttle command MSE during training epochs



((b)) Elevator command MSE during training epochs



((c)) Throttle command MSE during training epochs



((d)) Elevator command MSE during training epochs

Figure 3.7 : BC MSE errors for actuator commands

The trajectory obtained as a result of BC training is given in the Figure 3.8.

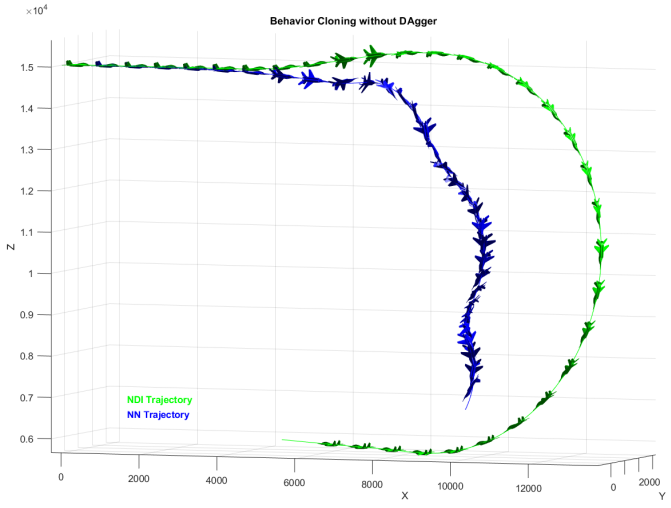


Figure 3.8 : Split-S maneuver by BC model

3.1.4 Discussion

In Figure 3.8 shows that,behavior cloning can be problematic because the errors made by the model during training can cascade and compound, leading to poor performance. This is because the training and test data sets are not necessarily drawn from the same distribution, violating the assumption of independent and identically distributed data that is typically used in supervised learning [23,25]. One potential solution to this cascading and compounding error problem is to collect data for all possible states or conditions, but this is often impractical due to the large number of potential inputs.

3.2 Behavior Cloning via Interactive Demonstrations

Advantage of behavior cloning with interactive demonstrations is that it allows the model to learn from its own mistakes, rather than just relying on the demonstration examples provided by the expert. This can make the model more flexible and adaptable, and able to generalize better to new situations or environments. Ross et al. [26] proposed an approach called DAgger (Data Aggregation) which reduces imitation learning to supervised learning with interaction, in an effort to address this compounding error problem.

Here's how the DAgger algorithm works:

- The BC agent is initialized with a policy that is trained on expert demonstrations.
- The agent then interacts with the environment and collects data from its own actions.
- The expert demonstrations and the data collected from the agent's actions are used to train a new policy.
- This process is repeated multiple times until the agent's policy converges to the expert policy.

One of the main benefits of the DAgger algorithm is that it can significantly reduce the amount of expert demonstrations required to train an RL agent. This is because the agent is able to learn from both expert demonstrations and its own actions, allowing it to learn more efficiently and effectively. Additionally, the DAgger algorithm has been shown to improve the robustness and generalization of RL agents, as it allows the agent to learn from a diverse set of expert policies.

3.2.1 Limit-DAgger algorithm

The DAgger algorithm is a method for aggregating data that involves adding new, modified training data that is labeled with inputs and outputs from an expert policy. The goal of this approach is to improve the performance of a NN model by incorporating additional, expert data.

But what is actually needed for training is not more data, but critical data from which it can learn to recover from deviations in the desired trajectory. Also the use of incorrect actions during training could have negative consequences on training performance.

In this paper [27],the CBA (stands for "Confidence-Based Autonomy") method is presented to select the most appropriate demonstration data for training a machine learning model in situations where it is important to avoid using incorrect actions during the training process. This concept served as the foundation for our Limit-DAgger algorithm.

The Limit-DAgger algorithm consists of three components;

1.) **Finding Limit PQR Tracking Error:** find the critical pqr error where an expert pilot cannot recover the maneuver.

2.) **Decision for Switching:** apply the NN model outputs to the system until it reaches a critical pqr error.

3.) **Correctional Demonstration:** perform corrective demonstrations when an incorrect action is selected by NN model.

3.2.2 Data generation

As first step, NN will be initiated with BC model and data will be collected using Limit-DAgger Algorithm for Limit-DAgger iter-1. For the next iterations, the NN model will be replaced with the latest trained NN model and loops continue until getting satisfactory results. Data generation flow is illustrated in Figure 3.9.

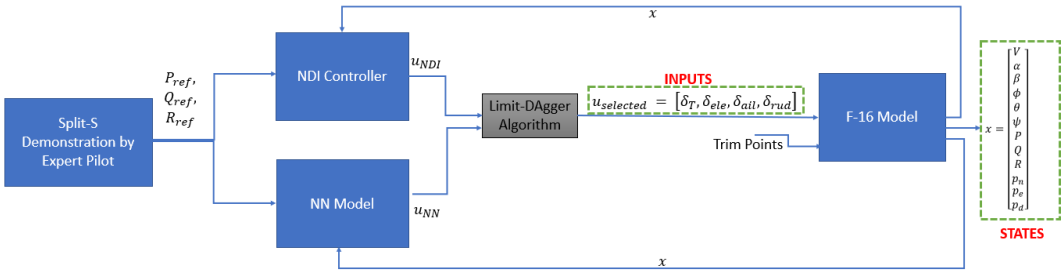


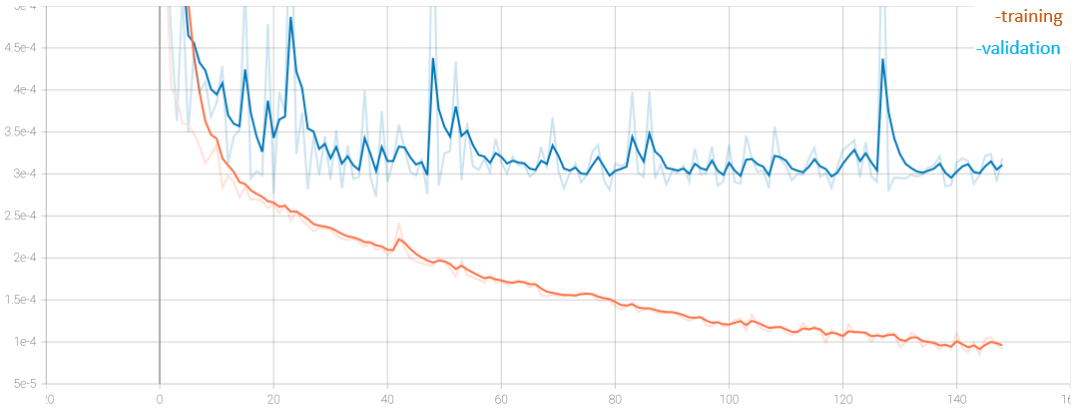
Figure 3.9 : Data generation flow for Limit-DAgger

3.2.3 Training results

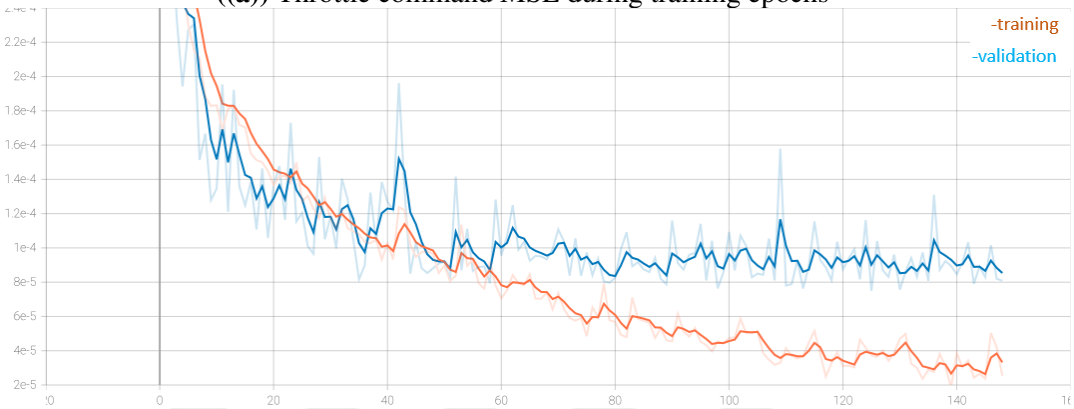
Training results for each iteration and related training losses are presented.

3.2.3.1 Limit-DAgger iteration-1

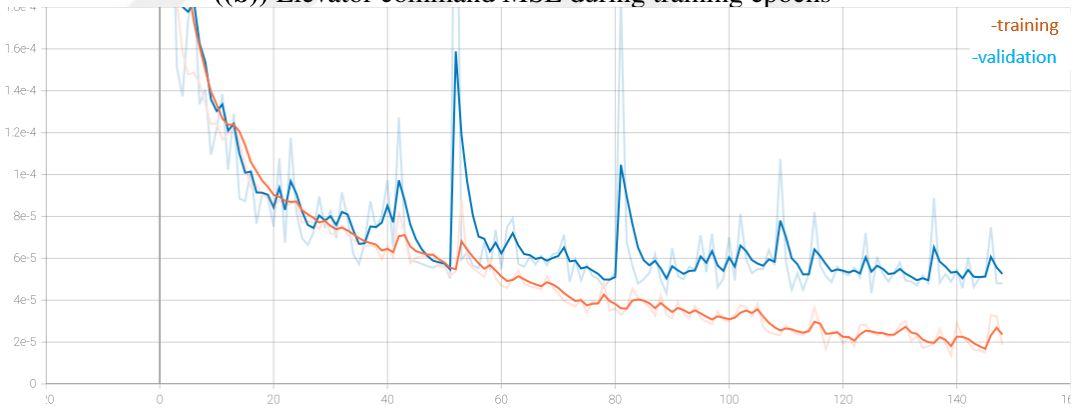
For the Limit-DAgger Iteration-1, the MSE losses for actuator commands during training epochs given in Figure 3.10.



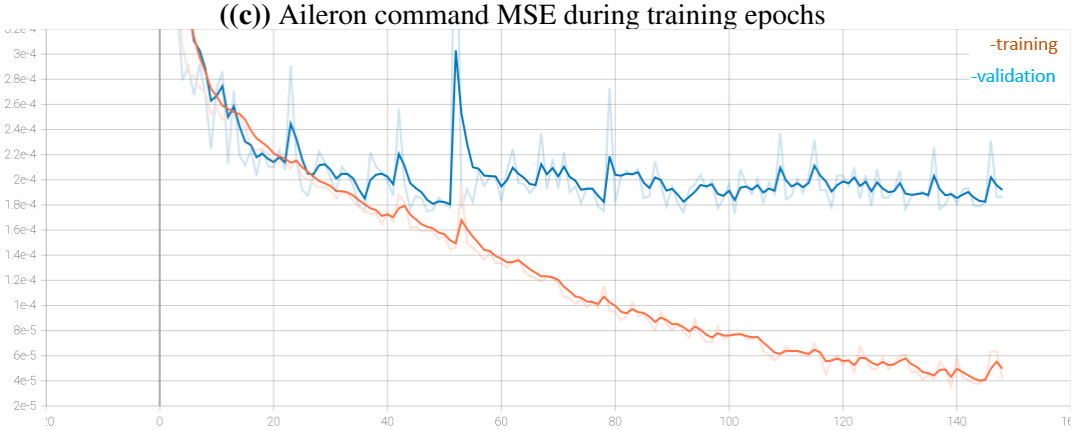
((a)) Throttle command MSE during training epochs



((b)) Elevator command MSE during training epochs



((c)) Aileron command MSE during training epochs



((d)) Rudder command MSE during training epochs

Figure 3.10 : Limit-DAgger iteration-1 MSE errors for actuator commands

The trajectory obtained as a result of Limit-DAgger Iteration-1 training is given in the Figure 3.11.

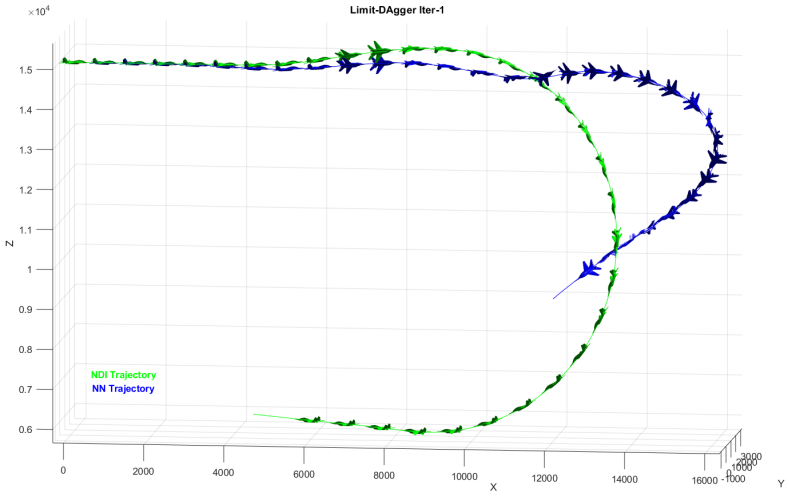
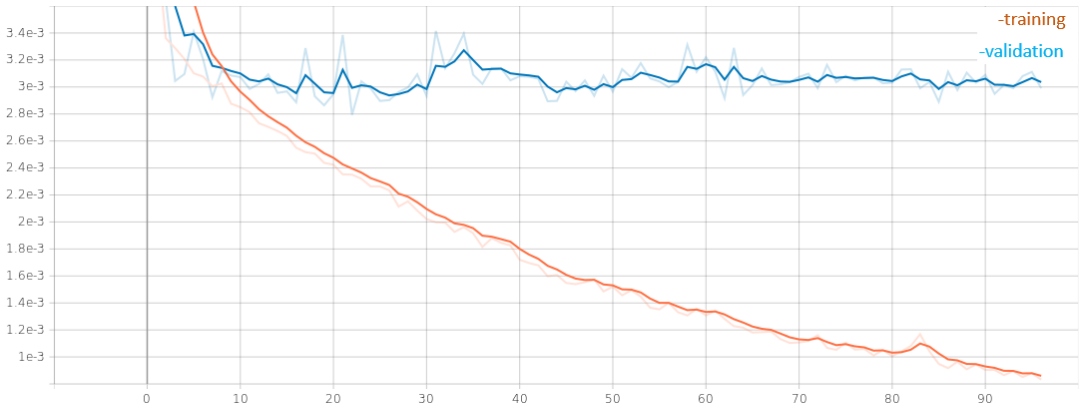


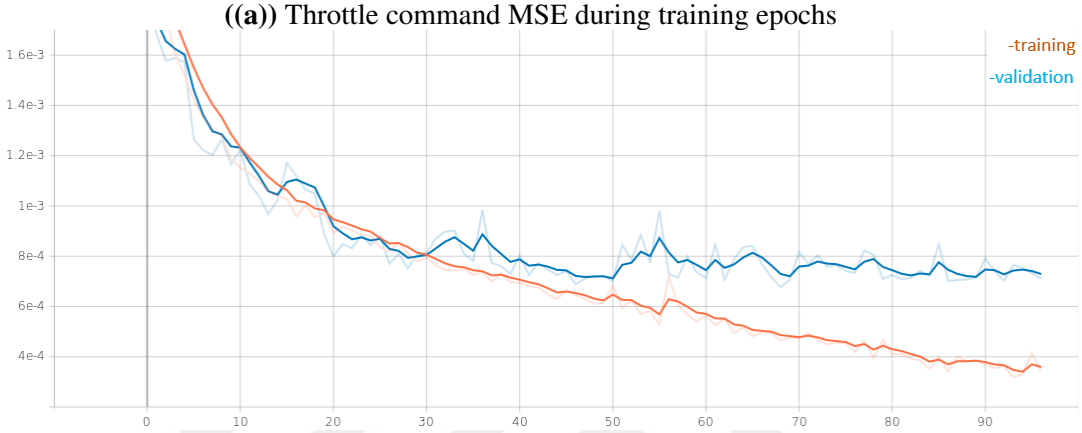
Figure 3.11 : Split-S maneuver by Limit-DAgger iteration-1

3.2.3.2 Limit-DAgger iteration-2

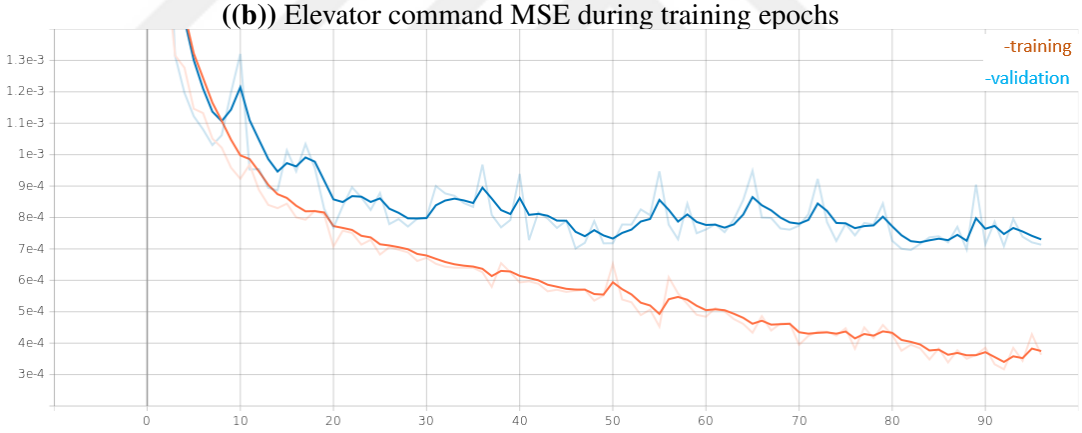
For the Limit-DAgger Iteration-2, the MSE losses for actuator commands during training epochs given in Figure 3.12.



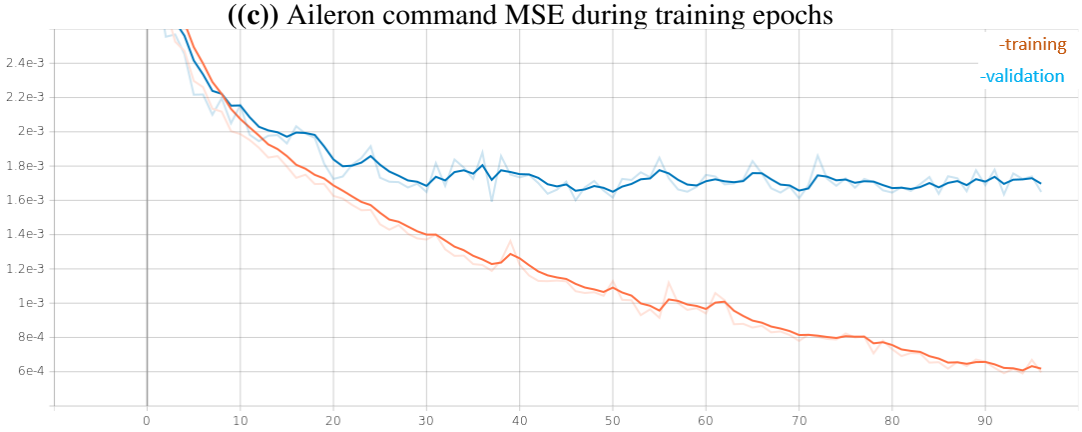
((a)) Throttle command MSE during training epochs



((b)) Elevator command MSE during training epochs



((c)) Aileron command MSE during training epochs



((d)) Rudder command MSE during training epochs

Figure 3.12 : Limit-DAgger iteration-2 MSE errors for actuator commands

The trajectory obtained as a result of Limit-DAgger Iteration-2 training is given in the Figure 3.13.

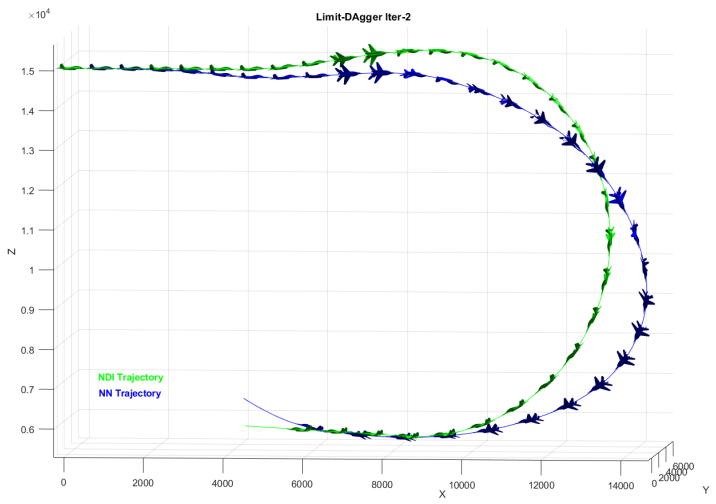
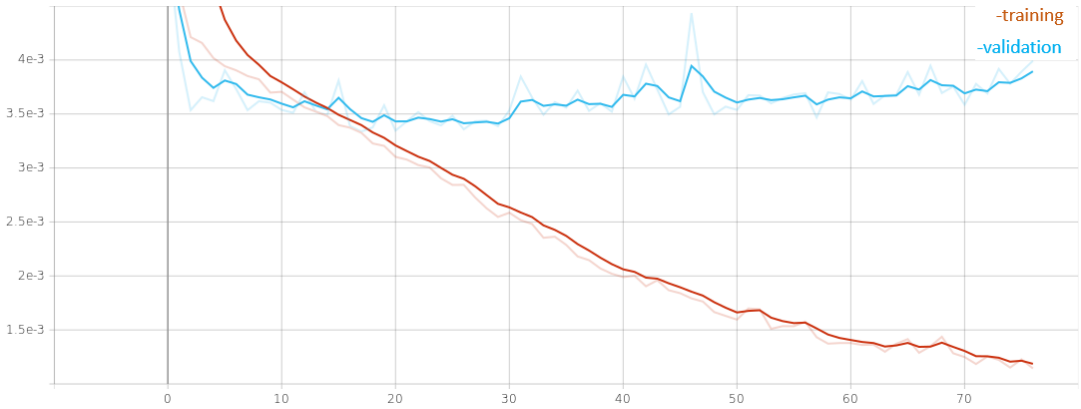


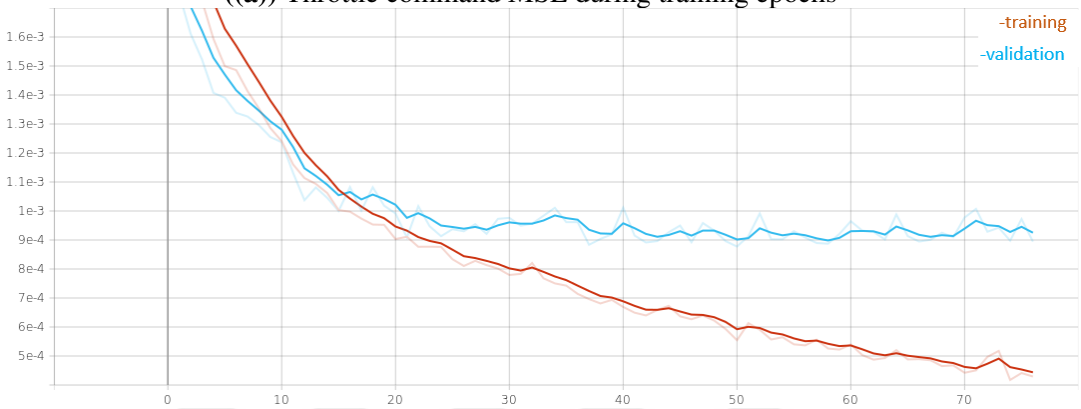
Figure 3.13 : Split-S maneuver by Limit-DAgger iteration-2

3.2.3.3 Limit DAgger iteration-3

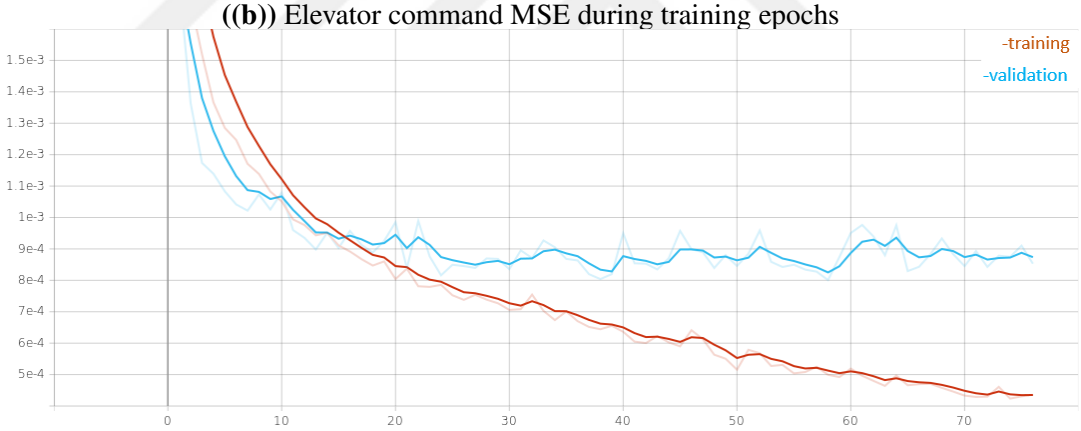
For the Limit-DAgger Iteration-3, the MSE losses for actuator commands during training epochs given in Figure 3.14.



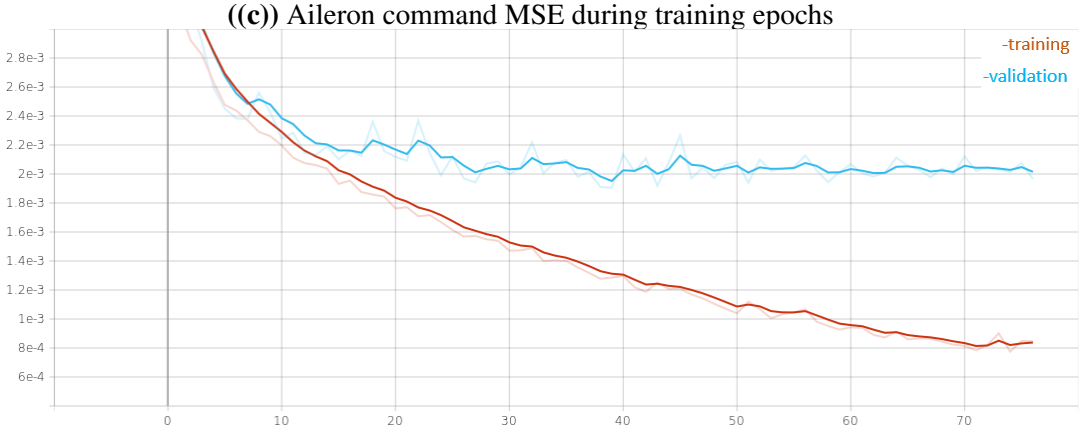
((**a**)) Throttle command MSE during training epochs



((**b**)) Elevator command MSE during training epochs



((**c**)) Aileron command MSE during training epochs



((**d**)) Rudder command MSE during training epochs

Figure 3.14 : Limit-DAgger iteration-3 MSE errors for actuator commands

The trajectory obtained as a result of Limit-DAgger Iteration-3 training is given in the Figure 3.15.

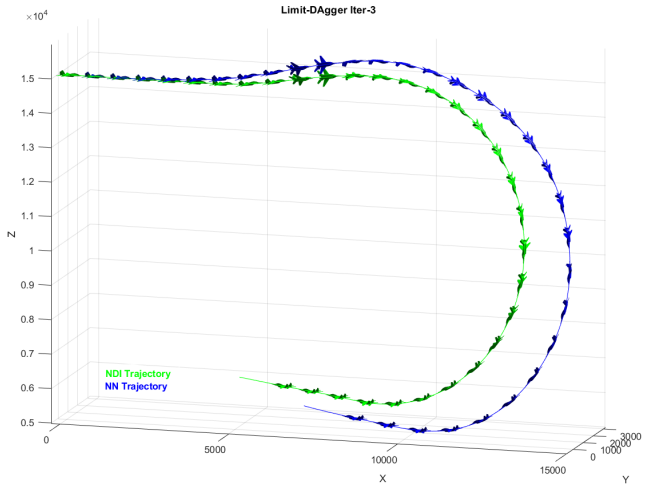
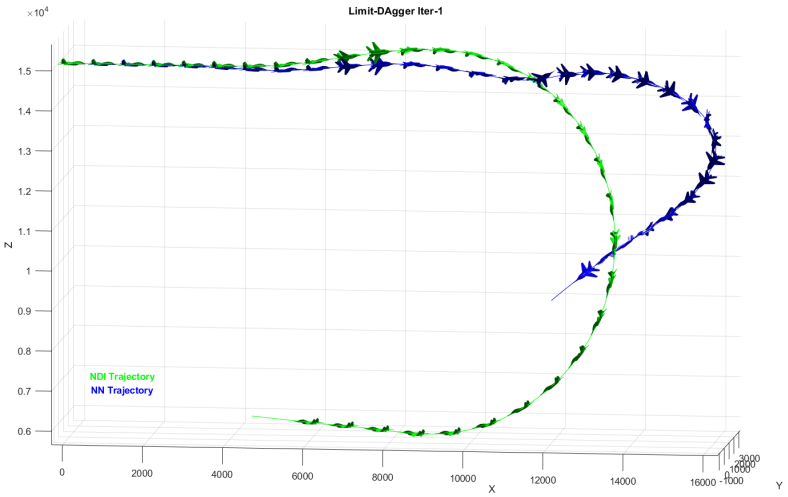


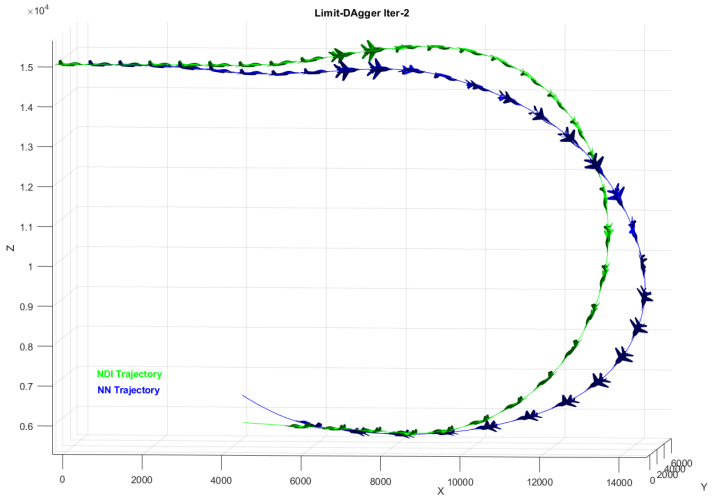
Figure 3.15 : Split-S maneuver by Limit-DAgger Iteration-3

3.2.4 Discussion

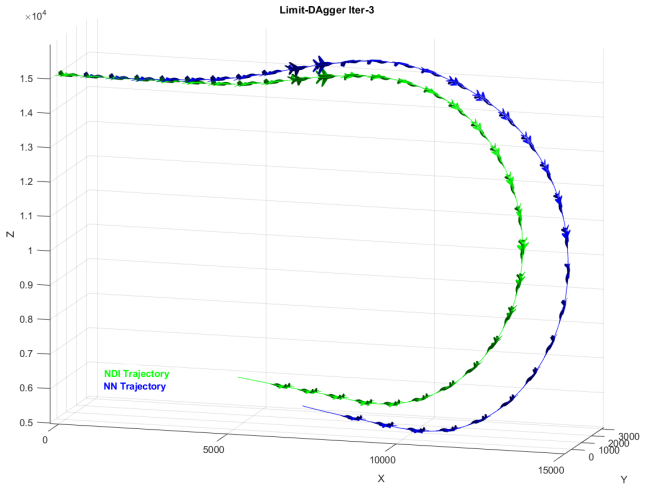
In Figure 3.16 shows the superiority of the Limit-DAgger algorithm and our NN architecture. The Split-S maneuver was learned just in three iterations and this shows how efficient the method is.



((a)) Split-S maneuver by Limit-DAgger iteration-1



((b)) Split-S maneuver by Limit-DAgger iteration-2



((c)) Split-S maneuver by Limit-DAgger iteration-3

Figure 3.16 : Development of NN model performance during training

4. TRANSFER LEARNING

Transfer learning is a machine learning technique where a model trained on one task is re-purposed on a second related task. This can be advantageous for a number of reasons:

1.) Data availability: It may be easier or more cost-effective to gather a large amount of data for one task, rather than multiple tasks. Transfer learning allows you to leverage this data to improve performance on a related task.

2.) Fewer training examples: Transfer learning can be particularly useful when the target task has a small amount of training data. By starting with a model that has already been trained on a related task, you can initialize the model with better weights, which can make it easier to learn the new task with fewer examples.

3.) Feature learning: Transfer learning can be effective because it allows the model to learn features that are relevant to the target task, rather than starting from scratch and learning features from scratch. This can make it easier for the model to generalize to the new task.

4.) Improved performance: In some cases, using transfer learning can improve the performance of the model on the target task. This can be especially true when the target task is related to the source task, as the model will already have learned some relevant features that can be useful in the new task.

5.) Reduced training time: Transfer learning can also reduce the amount of time and resources required to train a model on the target task. This can be especially useful when the target task requires a lot of data or computation to train a model from scratch [28,29].

4.1 Training Process

Only the last LSTM layer and dense layers coefficients are open for training. The initial learning rate started with 10^{-5}.

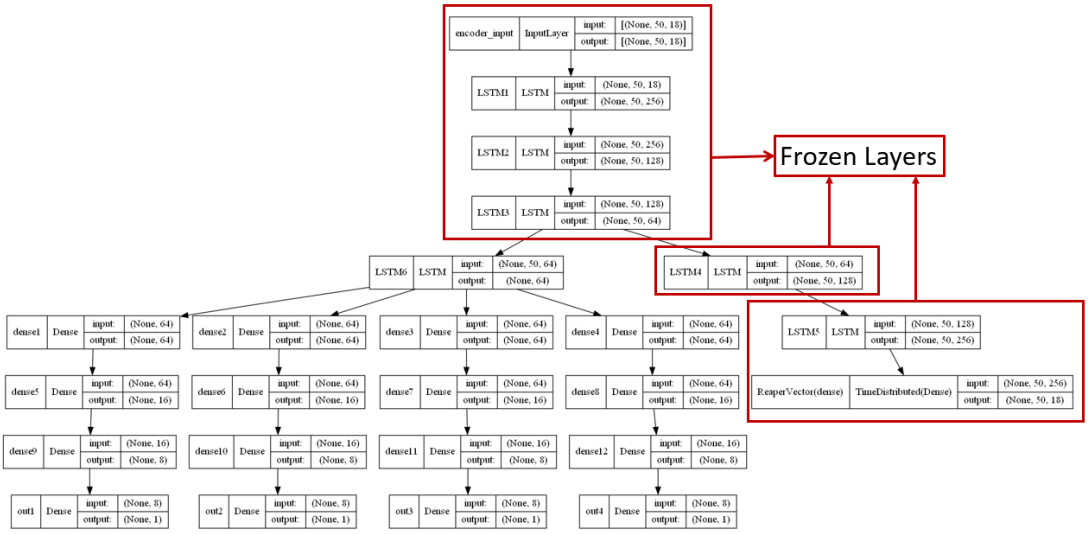


Figure 4.1 : Transfer learning with pre-trained F-16 model

4.2 Training Results

In order to measure pilot behavior model performance in a different dynamic model, the NN model's outputs are compared with the expert pilot maneuver that is demonstrated for this new model. The results for split-s maneuvers executed by the NN model and expert pilots are given below. Figure 4.2 shows the expert pilot and NN model stick commands on the new dynamic model. It is shown that the NN model produces almost same commands with expert pilot's.

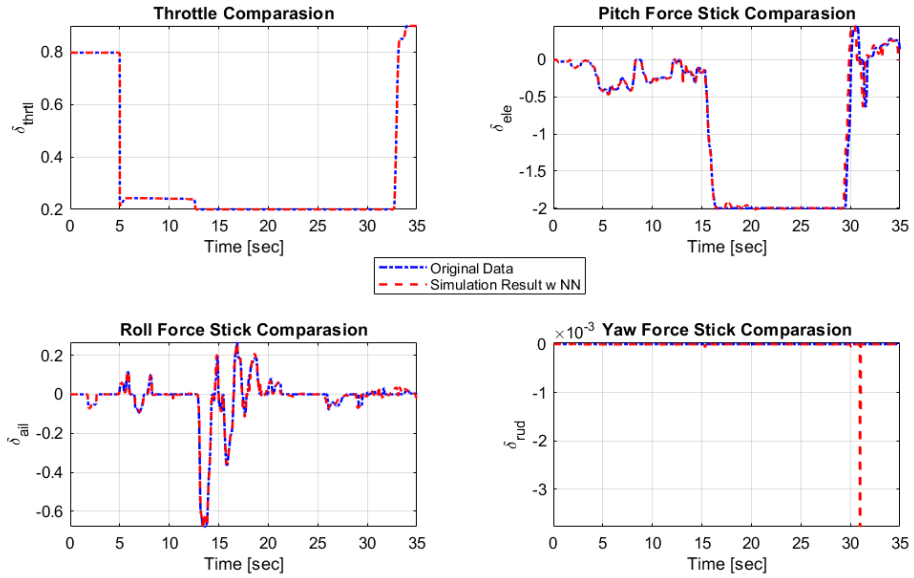


Figure 4.2 : Split-S - stick force commands for TL

The comparison of P, Q, R in Figure 4.4,ϕ, θ, ψ in Figure 4.3 and V_t, α, β in Figure 4.5 which are corresponding these commands are shown.

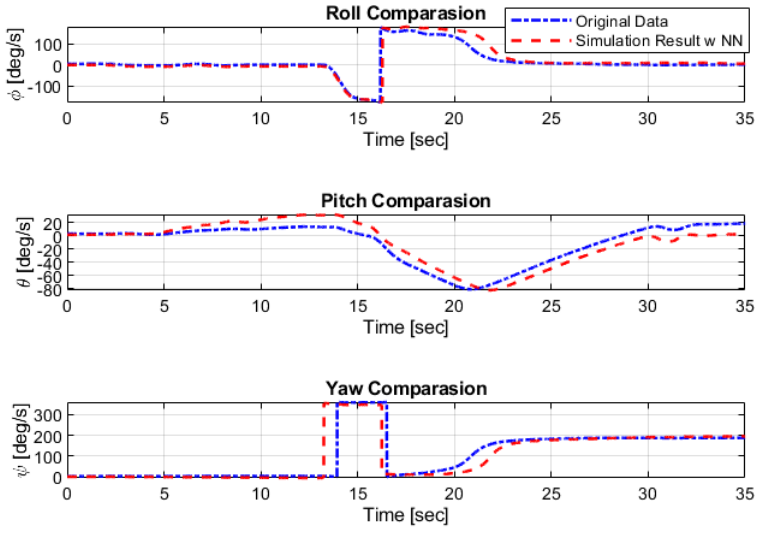


Figure 4.3 : Split-S - euler angles values for TL

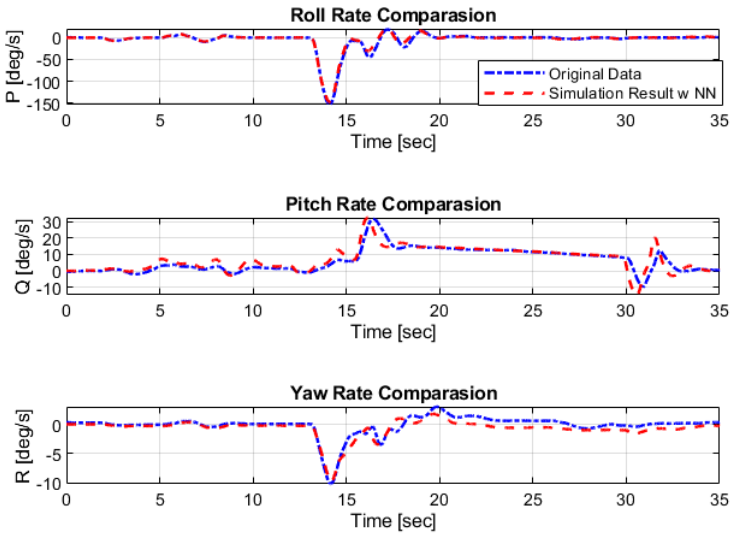


Figure 4.4 : Split-S - angular rates values for TL

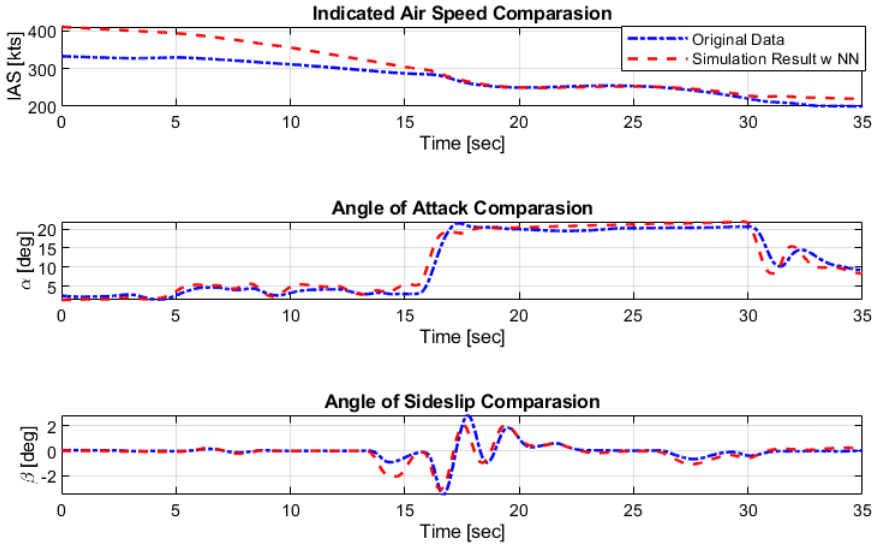


Figure 4.5 : Split-S - angle of attack-α,Sideslip-β, IAS-V for TL

In Figure 4.6 and 4.7, the 3D trajectory of the expert pilot and NN model are compared. The trim condition of the recorded maneuver is not an exact match, which is indicated by the difference in IAS-indicated airspeed plots in Figure 4.5. However, the resulting trajectory has the same characteristic behavior due to NN's generalization capability, as observed in 4.7.

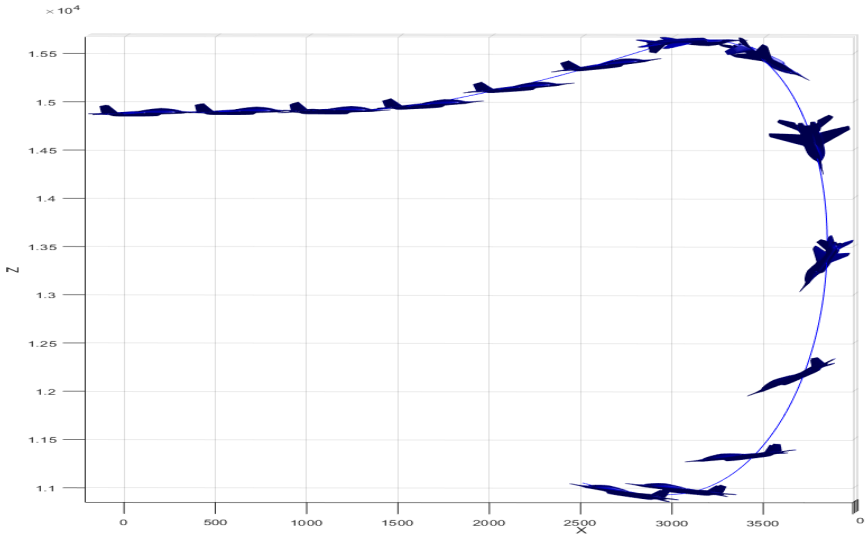


Figure 4.6 : Split-S maneuver by expert pilot

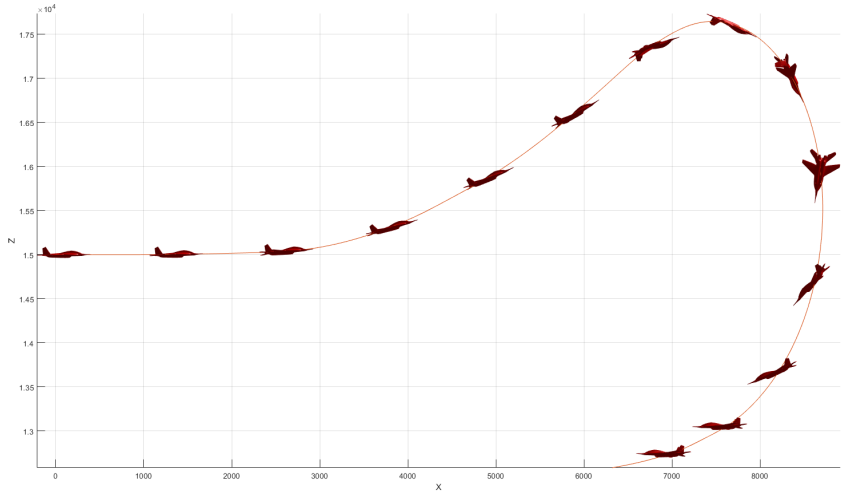


Figure 4.7 : Split-S maneuver by NN model

5. CONCLUSIONS

In this thesis, a fighter pilot behavior model capable of performing agile aggressive maneuvers was developed and implemented on a high-fidelity F-16 model. At first, Split-S maneuvers executed by NDI in various trim conditions and data was collected. The "expert" NDI controller was used for tracking of expert pilot's angular rates(to mimic the real pilot behavior) on high fidelity nonlinear 6-DOF F-16 model simulations . When the high-fidelity F-16 model was not available, a robust controller against parametric uncertainties should be preferred. When the Split-S dataset was analyzed, rapid and significant changes in the control inputs are observed. The composite LSTM autoencoder for sequence reconstruction and prediction is proposed as a solution of the problem on predicting rapid changes in control commands applied during aerobatic maneuvers. Then the resulted supervised training gives a BC model. The results showed that behavior cloning can be problematic because it can lead to poor performance if the model's errors during training are not properly addressed. In such cases, the errors made by the model may compound and cascade, resulting in poor performance.

Next, the DAgger algorithm which is a method for aggregating data that involves adding new, modified training data that is labeled with inputs and outputs from an expert policy to deal with the cascading and compounding error problems in sequential decision-making was further developed and Limit-Dagger algorithm proposed. Unlike the DAgger algorithm, the "Limit-DAgger" algorithm also offers efficiency in the training process. As shown in the simulation results, the Split-S maneuver could not be performed with BC, but it could be performed in 3 iterations using the Limit-DAgger algorithm.

Also, this model was transferred to other aircraft with limited data using transfer learning techniques.

REFERENCES

[1] **Ure, N.K. and Inalhan, G.** (2012). Autonomous control of unmanned combat air vehicles: Design of a multimodal control and flight planning framework for agile maneuvering, *IEEE Control Systems Magazine*, *32*(5), 74–95.

[2] **McConley, M.W.**, **Piedmonte, M.D.**, **Appleby, B.D.**, **Frazzoli, E.**, **Feron, E. and Dahleh, M.A.** (2000). Hybrid control for aggressive maneuvering of autonomous aerial vehicles, *19th DASC. 19th Digital Avionics Systems Conference. Proceedings (Cat. No. 00CH37126)*, volume 1, IEEE, pp.1E4–1.

[3] **Satir, A.S.**, **Demir, U.**, **Sever, G.G. and Ure, N.K.** (2021). Nonlinear Model Based Guidance with Deep Learning Based Target Trajectory Prediction Against Aerial Agile Attack Patterns, *2021 American Control Conference (ACC)*, IEEE, pp.2607–2612.

[4] **Laptev, N.**, **Yosinski, J.**, **Li, L.E. and Smyl, S.** (2017). Time-series extreme event forecasting with neural networks at uber, *International conference on machine learning*, volume 34, sn, pp.1–5.

[5] **Ackerman, K.**, **Pelech, S.**, **Carbonari, R.**, **Hovakimyan, N.**, **Kirlik, A. and Gregory, I.M.** (2014). Pilot-in-the-loop flight simulator for NASA's Transport Class Model, *AIAA Guidance, Navigation, and Control Conference*, p.0613.

[6] **Lone, M.M. and Cooke, A.K.** (2013). Pilot-model-in-the-loop simulation environment to study large aircraft dynamics, *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of aerospace engineering*, *227*(3), 555–568.

[7] **Vidakovic, J.**, **Lazarevic, M.**, **Kvrgic, V.**, **Vasovic Maksimovic, I. and Rakic, A.** (2021). Flight Simulation Training Devices: Application, Classification, and Research, *International Journal of Aeronautical and Space Sciences*, *22*(4), 874–885.

[8] **Levin, J.M.**, **Paranjape, A. and Nahon, M.** (2017). Agile fixed-wing uav motion planning with knife-edge maneuvers, *2017 international conference on unmanned aircraft systems (ICUAS)*, IEEE, pp.114–123.

[9] **Levin, J.M.**, **Paranjape, A.A. and Nahon, M.** (2019). Agile maneuvering with a small fixed-wing unmanned aerial vehicle, *Robotics and Autonomous Systems*, *116*, 148–161.

[10] **Bulka, E. and Nahon, M.** (2017). Autonomous control of agile fixed-wing uavs performing aerobatic maneuvers, *2017 international conference on unmanned aircraft systems (ICUAS)*, IEEE, pp.104–113.

[11] **Bulka, E. and Nahon, M.** (2019). Automatic control for aerobatic maneuvering of agile fixed-wing UAVs, *Journal of Intelligent & Robotic Systems*, *93*(1), 85–100.

[12] **Ure, N.K. and Inalhan, G.** (2008). Design of higher order sliding mode control laws for a multi modal agile maneuvering UCAV, *2008 2nd International Symposium on Systems and Control in Aerospace and Astronautics*, IEEE, pp.1–6.

[13] **Shukla, D.**, **Keshmiri, S. and Beckage, N.** (2020). Imitation learning for neural network autopilot in fixed-wing unmanned aerial systems, *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, pp.1508–1517.

[14] **Medeiros, C.M.N.** (2021). Learn to Fly: Cloning the Behavior of a Pilot.

[15] **Sandström, V.** (2021). *On the Efficiency of Transfer Learning in a Fighter Pilot Behavior Modelling Context*.

[16] **Sandström, V.**, **Luotsinen, L. and Oskarsson, D.** (2022). Fighter Pilot Behavior Cloning, *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, pp.686–695.

[17] **Nguyen, L.T.** (1979). *Simulator study of stall/post-stall characteristics of a fighter airplane with relaxed longitudinal static stability*, volume12854, National Aeronautics and Space Administration.

[18] **Shinar, J.**, **Merari, A.**, **Blank, D. and Medinah, E.** (1980). Analysis of optimal turning maneuvers in the vertical plane, *Journal of Guidance and Control*, *3*(1), 69–77.

[19] **de Freitas, H.M.** (2022). Learn to Fly II: Acrobatic Manoeuvres.

[20] **Hochreiter, S. and Schmidhuber, J.** (1997). Long short-term memory, *Neural computation*, *9*(8), 1735–1780.

[21] **Tai, L.**, **Zhang, J.**, **Liu, M.**, **Boedecker, J. and Burgard, W.** (2016). A survey of deep network solutions for learning control in robotics: From reinforcement to imitation, *arXiv preprint arXiv:1612.07139*.

[22] **Hussein, A.**, **Gaber, M.M.**, **Elyan, E. and Jayne, C.** (2017). Imitation learning: A survey of learning methods, *ACM Computing Surveys (CSUR)*, *50*(2), 1–35.

[23] **Osa, T.**, **Pajarinen, J.**, **Neumann, G.**, **Bagnell, J.A.**, **Abbeel, P.**, **Peters, J.** *et al.* (2018). An algorithmic perspective on imitation learning, *Foundations and Trends® in Robotics*, *7*(1-2), 1–179.

[24] **Hussein, A.**, **Gaber, M.M.**, **Elyan, E. and Jayne, C.** (2017). Imitation learning: A survey of learning methods, *ACM Computing Surveys (CSUR)*, *50*(2), 1–35.

[25] **Bagnell, J.A.** (2015). An invitation to imitation, **Technical Report**, Carnegie-Mellon Univ Pittsburgh Pa Robotics Inst.

[26] **Ross, S.**, **Gordon, G. and Bagnell, D.** (2011). A reduction of imitation learning and structured prediction to no-regret online learning, *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, JMLR Workshop and Conference Proceedings, pp.627–635.

[27] **Chernova, S. and Veloso, M.** (2009). Interactive policy learning through confidence-based autonomy, *Journal of Artificial Intelligence Research*, *34*, 1–25.

[28] **Tan, C.**, **Sun, F.**, **Kong, T.**, **Zhang, W.**, **Yang, C. and Liu, C.** (2018). A survey on deep transfer learning, *International conference on artificial neural networks*, Springer, pp.270–279.

[29] **Pan, S.J. and Yang, Q.** (2010). A survey on transfer learning, *IEEE Transactions on knowledge and data engineering*, *22*(10), 1345–1359.

CURRICULUM VITAE

Name SURNAME: Gülay SEVER

EDUCATION:

- **B.Sc.:** 2014, Yıldız Technical University, Faculty of Mechanical Engineering, Mechatronics Engineering

PUBLICATIONS ON THE THESIS:

- Satir, A. S., Demir, U., **Sever, G. G..**. (2021, May). A Nonlinear Model Based Guidance with Deep Learning Based Target Trajectory Prediction Against Aerial Agile Attack Patterns. *In 2021 American Control Conference (ACC),* 4(pp. 2607-2612). (IEE)

OTHER PUBLICATIONS:

- Demir, U., Satir, A. S., **Sever, G. G..**,Yikilmaz, C., & Ure, N. K. (2022). Scalable Planning and Learning Framework Development for Swarm-to-Swarm Engagement Problems. *arXiv preprint,* (arXiv:2212.02909).