DEVELOPMENT AND EXPERIMENTAL EVALUATION OF AN EMBODIED COGNITIVE MODEL FOR PILOTING OF AIR VEHICLES IN MISSION FLIGHTS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS INSTITUTE
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

YASİN KAYGUSUZ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE
OF
DOCTOR OF PHILOSOPHY
IN
THE DEPARTMENT OF COGNITIVE SCIENCE

JANUARY 2023

DEVELOPMENT AND EXPERIMENTAL EVALUATION OF AN EMBODIED COGNITIVE MODEL FOR PILOTING OF AIR VEHICLES IN MISSION FLIGHTS

Submitted by Yasin Kaygusuz in partial fulfilment of the requirements for the Master of Science in Cognitive Science, Middle East Technical University by,

Date: 27th JANUARY 2023

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, Last Name: **Yasin Kaygusuz**

Signature: _____

ABSTRACT

DEVELOPMENT AND EXPERIMENTAL EVALUATION OF AN EMBODIED COGNITIVE MODEL FOR PILOTING OF AIR VEHICLES IN MISSION FLIGHTS

Kaygusuz, Yasin
Department of Cognitive Science
Supervisor: Assist. Prof. Dr. Murat Perit Çakır

FEBRUARY 2023, 80 pages

Various types of engineering methods have been successfully applied for the automated control of attitudes and flight of manned or unmanned air vehicles. Some of these methods included "pilot models," which are generally implemented in the form of transfer functions and are not necessarily cognitive in nature. Additionally, symbolic and connectionist pilot models have been developed to control the maneuvering of air vehicles. While some of these models have been successful, they are mostly focused on attitude control and do not take into account higher-level decisions such as attacking or chasing an opponent. The current modeling efforts in cognitive science do not provide enough information on the structure of those high-level decision-making processes. This study aims to create and test a dynamic cognitive fighter pilot model in a simulated environment that can make high-level decisions using the Neural Engineering Framework and employing spiking neuron models as encoding units. The model is connectionist, embodied, and situated and uses semantic pointers to decode the outputs into semantic wordings. The study also includes the comments and evaluation of real fighter pilots on the results of the model.

Keywords: Spiking Neuron Networks, Neural Engineering Framework, NEF, Cognitive Pilot Model, Decision-making Model.

ÖZ

GÖREV UÇUŞU GERÇEKLEŞTİREN HAVA ARAÇLARI İÇİN VÜCUTLAŞTIRILMIŞ BİLİŞSEL BİR PİLOT MODELİNİN GELİŞTİRİLMESİ VE DEĞERLENDİRİLMESİ

Kaygusuz, Yasin
Bilişsel Bilimler Bölümü
Tez Yöneticisi: Dr. Öğretim Üyesi Murat Perit Çakır

ŞUBAT 2023, 80 sayfa

İnsanlı ya da insansız hava araçlarının uçuş ve durumlarının otomatik kontrolü için geliştirilmiş pek çok başarılı Mühendislik yöntemi mevcuttur. Bu yöntemlerin bazıları "pilot model"leri içermekle beraber, bu modeler çoğunlukla transfer fonksiyonu şeklinde tanımlanmış olduğundan genellikle bilişsel bir içeriğe sahip değildir. Bunlara ek olarak, sembolik ya da bağlantıcı/yapay-sinir-ağı temelli bilişsel modeller de hava araçlarının manevra kontrolünde kullanılmıştır. Bu modellerin bazıları son derece başarılıyken, genellikle odaklanılan nokta durum kontrolüdür ve bir rakibin takip edilmesi, avlanması ya da kaçılması gibi kritik üst seviye kararlar üzerinde çok durulmamıştır. Dolayısıyla, bilişsel bilimlerdeki bugünkü modelleme çabaları havacılık bağlamında üst seviye karar verme süreçleri ve bu kararların hakkında yeterli bilgi sağlamamaktadır. Bu çalışmada ise, NENGO Sinirsel Mühendislik Çerçevesi kapsamında atım yapan nöron ağlarını temel alarak üst seviye kararları alabilen dinamik bir bilişsel avcı uçağı pilotunun modellenmesi ve benzetimli bir ortamda değerlendirilmesi amaçlanmıştır. Model bağlantıcı, bedenlenmiş ve durumlanmış özellikler taşımakta ve semantik işaretçiler vasıtası ile model çıktılarının semantik söylemlere çevrilmesine imkan verebilmektedir. Çalışma ayrıca son bölümünde gerçek avcı pilotlarının model çıktılarına dair yorumlarını da içermektedir.

Anahtar Kelimeler: Atımlı Nöron Ağları, Sinirsel Mühendislik Çerçevesi, Bilişsel Pilot Modeli, Karar Verme Modeli.

To My Family.

ACKNOWLEDGEMENT

The study provided in this thesis is supported by Turkish Aerospace Industries' Technology Management Department. I express sincere appreciation to Turkish Aerospace Industries and all its executives on duty, starting from Prof. Dr. Atilla Doğan and Ms. Nezaket Güneri Orbay, for their encouraging and inspiring attitude toward employees' academic studies. It may be acknowledged that this supportive environment has resulted from the innovative changes inflicted in the dynamics of Turkish Aerospace with the visions set by Prof. Dr. Temel Kotil as the president and CEO. Of course, I may thank Dr. Murat Perit Çakır as my thesis supervisor and the person who directed me towards modeling in cognitive science for all his contribution, aid, and patience. My teammates in Turkish Aerospace also provided me with great encouragement. Finally, to my wife Deniz, my daughters Sare and Elif Bade, and my son Kaan Sadi I thank you from my heart for your patience in my long absences and continuous faith in me. Without them, I would not be as motivated to understand human cognitive behaviour.

TABLE OF CONTENTS

ABSTRACT iv
ÖZ v
ACKNOWLEDGEMENT vii
TABLE OF CONTENTS ix
LIST of Figures xi
LIST of TABLES xiii
CHAPTERS
1 INTRODUCTION 1
2 A REVIEW OF METHODS IN COGNITIVE MODELLING AND PILOT MODELS 7
2.1 Control Theory Models 7
2.2 Models Based On Symbolic Architectures 9
2.3 Neural Network Models 11
2.4 Models Employing Neural Fields And Neuron Models 13
2.1.1 The Neural Engineering Framework and NENGO 17
2.1.2 Artificial Intelligence and Robotics Applications Of NEF And NENGO 22
3 CONTENT OF THE MODEL: UNDERSTANDING THE FIGHTER PILOT DECISIONS AND AERIAL COMBAT 25
4 CONTENT OF THE MODEL: METHOD AND STRUCTURE 35
CHAPTER 55 PILOTS' EVALUATION, RESULTS, DISCUSSION AND CONCLUSION 47
5.1 Effects Of Noise in The Functionality Of Spiking Neuron Networks 47
5.2 Symbolic Graph Methods and SNNs 57
5.3 Pilots' Evaluation, Results, Discussion And Conclusion 64
REFERENCES 69

APPENDIX 1: TRANSCRIPTION OF AN ENGAGEMENT SCENARIO WITH A PILOT 79

LIST OF FIGURES

Figure 1. A representation of the Johson and Pritchett's Generic Pilot Model (2002) as represented in Kaygusuz (2015). 8
Figure 2. Hutchins' Information Flow Model from Kaygusuz (2015). 9
Figure 3. Architecture of ACT-R 5.0 by Anderson et al. (2004). 11
Figure 4. Response and tuning curves of a neuron ensemble. 20
Figure 5. Inputs multiplied by weights to reach outputs, a classical neural network example from SimBrain. 21
Figure 6. NENGO GUI and Python script sample for a basic model. 22
Figure 7. BVR to WVR ranges comparison for three different scenarios. 27
Figure 8. Representative Radar Cross Section in a specific frequency. 28
Figure 9. Antenna Training Angle and Aspect Angle definition for engagements. 29
Figure 10. WVR Head Up Symbology for western aircraft provided with permissions of Turkish Aerospace Industries. 29
Figure 11. A brief summary of major engagement flow and decisions. 34
Figure 12. A Complete Aerial Mission Pilot Model Architecture. 36
Figure 13. Usage of inhibitory and excitatory networks together. 38
Figure 14. Overall model structure. 39
Figure 15. Inventory evaluation network. 40
Figure 16. The Identification evaluation network. 41
Figure 17. Range Evaluation Network. 42
Figure 18. Set of 4 SPs from NENGO-GUI. 42
Figure 19. A set of inputs for an example scenario. 46
Figure 20. The output voltage difference of two spiking neurons where one is subjected to white noise (Carter and Mancini, 2018) 49
Figure 21. IAC model for inventory evaluation. 50
Figure 22. Noise applied to gun, Synaptic weight X = -0.8. 52
Figure 23. Noise applied to gun, Synaptic weight X = 0. Both nodes are excited by only self. No external stimulus. 53
Figure 24. Noise applied to gun, a heavy inhibitory Synaptic weight X = -2. 53
Figure 25. PES learning model for squaring function implementation. 54
Figure 26. Squaring function learning under noise is represented. Left hand side graphic shows the case where there is no noise in the input A.. 56

Figure 27. Applied Gaussian noise amplitude in peak is 10 times higher than the given in previous example.....56
Figure 28. Decision Forest Illustration.....59
Figure 29. The Decision Tree Model for Pilot's High Level Decisions 62

LIST OF TABLES

Table 1. Summary of Scenarios. 43
Table 2. Decisions by the model. 64
Table 3. Pilot evaluation status and disagreement scenarios. 65

CHAPTER 1

1 INTRODUCTION

In the Embodied Approach version of the Handbook of Cognitive Science, Berkeley (2008) presents a good representation of embodied cognitive robotics using a CAJUNBOT robotics problem. The environment needs to be perceived by sensors in the hardware layer, which needs to be encoded into the software layer for further processing. Then, the encoded data is used in a path planning algorithm whose output path segments are decoded by the actuators to generate mobility. The depicted loop is a perception-cognition-action flow unfolding in interaction with the environment. In the case of a human agent situated in much more complicated task environments, Hutchins (1995; 2000) provided very similar models covering both the naval and flight deck settings. The model proposed by Hutchins (1995) claims an internal representation or understanding of the external system acquired by the agent where the representations are manipulated in interaction with other agents and instruments that altogether constitute a cognitive system (1995), with a further clear description, a cognition distributed across individuals, tools and the environment. In fact, Berkeley's model is a much more abstract and simplified version of Hutchins' model, whereas Hutchins' model is focused on a higher level of complexity. Whatever the level of complexity or the problem in the context, the agents are always depicted in interaction with the environment to perceive it, to cognitively compute about or on the data of it, and act in or on it. Therefore, the model or agent developed in this dissertation is intended to perform in a similar vein. Nevertheless, this time the subject of the model is the human pilot, and the environment is a simulated aerial scene, where the intention is not only to perform operations on encoded environment data but also to provide semantic outputs which can be transferred as communication outputs in the future to guide subsequent actions/decisions.

In the related literature, several mathematical models that emulate a human pilot's[1] abilities to control an aircraft have been proposed in aerospace engineering since the 1950s. Most of these are functional models that provide a transfer function constructed

[1] In the modern scientific literature, the term Pilot is used in a very wide context and different meanings. In this study, the term Pilot refers to the human agent who flies or conducts an air vehicle.

upon control theoretic principles such as root loci and Bodé plots[2], primarily based on data obtained from pilot questionnaires or flight test instrumentation. Many examples of such models are available in Blakelock (1991). In particular, these models belong to the first among two main modeling paradigms, which focus on performing a specific task using a pilot transfer function to control some specific aspect of aircraft attitude. Such models treat the pilot as a set of control equations in the control loop of an aircraft function. Although these control models are highly functional in solving control problems of systems, they are primarily geared towards controlling the aircraft as part of an autopilot system rather than providing tools for understanding human piloting behaviour.

Despite their conventional naming, control theory based models are not proper pilot models; in fact, they are mostly autopilot models for attitude or speed hold modes. These models can control an aircraft in a position, attitude, or speed which is externally selected by an agent but cannot adapt to a situation and cannot change the intended attitude. Therefore, control theoretic models lack cognitive capabilities such as decision-making, planning, and adaptive control by their very nature.

On the other side, during an aerial mission, the "pilot in control" evaluates current inputs or knowledge in a partially observable environment, decides on an action under high uncertainty conditions, and performs the selected action. When inputs to the situation or the overall system state change during the action, the pilot's decision may also need to be re-evaluated and changed. Therefore, a pilot model may include higher-level functionalities in comparison with an autopilot in which the intended attitude values are manually selected by a real human agent.

In this dissertation, the purpose has been to develop, implement and evaluate a cognitive pilot model, which can assess certain aspects of the environment and the opponent's status, decide what to do from a set of possible behaviours, and finally provide a semantic representation of the decisions. Therefore, since the environment and the partner/opponent's action in interaction will continuously change, the pilot model in control shall also be continuously updating its decisions and behaviour to adapt to the situation. To be able to perform the above functions with various sensory inputs, the model shall rapidly evaluate these inputs and be able to react to the detected changes. Therefore, such an agent may not only be adaptive but also embodied in some manner to achieve the required level of responsiveness.

In this dissertation study, the Neural Engineering Framework (NEF) (Eliasmith, 2013) will be employed as a computational modelling framework that provides spiking neurons as computational units. With that method, the study aims to answer questions on the possibility and plausibility of such a high-level decision model, benefiting from the dynamical characteristics of the spiking neuron model and the semantic pointers provided by NEF. Employing the Semantic Pointer Architecture (SPA) to decode the syntactic output into a semantic wording, i.e., providing a transformation between the computational output and the natural language, is a new approach and a significant benefit provided by NEF. Such a study can be used in the development of aerial robotics applications, computer-generated forces in simulators and training setups, game-based applications, and serious games, as well as operational analysis of aerial warfare. Despite all these positive aspects, the use of spiking neurons has been rare in robotics applications in comparison with classical machine learning methods and deep nets. Additionally, their

[2] Root Loci and Bodé Plots: Root Loci is a graphical representation used in the analysis or roots and poles of a system equation of a control system. Bodé plot is the graphical representation of a system's frequency domain characteristics in logarithmic scale. These two methods are very commonly employed in the control theory for rpbust control of systems.

characteristics in a noisy environment are not investigated in detail, and the current literature utilizes other approaches, such as the symbolic graph methods, for developing decision models that are less susceptible to noise. Therefore, in a modelling study of high-level pilot decisions, a survey of alternative methods, noise, and challenging scenarios may be necessary. Similarly, the outputs of a model shall also be critiqued by human pilots or domain experts to evaluate the accuracy of the decision outcomes and their practical applicability in the industry. This study intends to provide an answer to the above questions by developing and evaluating a model based upon a Spiking Neural Network (SNN) with semantic pointers compared with alternative methods, especially graph-based decision methods (Decision Trees and Decision Forests), surveying the model's performance under noise, and assessing the success of the model by real fighter pilots. Also, the study will provide a cross-check of pilot comments to point out cases of disagreement among human experts and discuss the plausibility of the model's output in those circumstances.

In summary, in this dissertation;

- An adaptive and embodied pilot model was developed,
- Interviews with experienced fighter pilots were conducted to understand the rules of interaction, formation, and engagement, as well as tips and rules of thumbs employed by human pilots,
- Connectionist sub-networks to aid the decision-making process of the overall pilot model were developed,
- The results were evaluated and analysed based on the comments obtained from experienced fighter pilots.

For the planned dynamical, adaptive, hybrid, embodied cognitive pilot model able to act in a limitedly simulated world, most of the above items are intuitively required.

During the history of cognitive modelling studies, various approaches have been used to construct perception-cognition-action loops. The second chapter will provide an overview of existing modelling and simulation approaches in the context of aviation. Modern aviation uses various types of simulation technologies for various purposes. Although there are many types with slight differences between them, the primary uses of simulations may be listed as follows. First, training simulations are popularly used to increase flight crew abilities and workload assessment. Secondly, most aviation systems, including avionics, are simulated in the early phases of development projects to decrease costs by anticipating potential design issues. Lastly, to plan efficient aerial operations, models of the operational scene are often used, including "blue and red" aircraft[3] teams in the presence of surface-based forces. It is evident that in cases where an enemy or friendly aircraft is flying under the control of a computer, there is a practical need for computerized intelligence to create pilot behaviour to control the aircraft. The second chapter will provide a summary of previous studies that focus on pilot modelling, and motivate the need for a better understanding of the information flow within a cockpit.

To understand the high-level decisions shaping aerial warfare, a description of the theatre and the relevant low-level decisions is needed, in addition to action possibilities for the agents. Such information may be useful for defining high and low-level decisions and understanding the content of the training sets. The third chapter provides a summary of

[3] Red and Blue Aircrafts: Red and Blue terms are definitions origining from the cold war era, where aircrafts of the nations in the Sovietic Union are considered as REDs and western block as BLUEs. The terms are still used in the western aerial exercises where blue trainees are combatting against reds to master their skills.

aerial engagements from top to bottom along several dimensions. The chapter provides a description of high and low-level decisions. Additionally, the link between the decisions and the manoeuvres is also summarized.

The second chapter contains a short summary of the Neural Engineering Framework (NEF) and its foundations, the concept of spiking neurons, spiking neural network (SNN) models, and applications of SNNs. It provides a comparative analysis of some of the prominent methods in the literature, such as dynamic field theory (DFT) based models or symbolic architectures such as ACT-R and SOAR. Also, the chapter contains a summary of the mathematical roots of the spiking neuron models. Clearly, a pilot model constructed upon a neural network in which multiple functionalities and subnetworks are implemented is a very complex undertaking, and such a model will be difficult to implement without an appropriate framework. Although existing frameworks such as NEF provides already implemented solvers for spiking neuron equations, due to a lack of support for training SNNs, several equations needed to be solved individually in a functional way to develop a higher-level pilot model. The NEF/NENGO framework provides embedded solvers and a graphical modelling environment in conjunction with a Python interpreter enabling the user to solve complex SNNs and offering capabilities such as semantic pointers.

Finally, the fourth chapter contains a detailed description of the main pilot model, including the subnetworks, the semantic pointers, and the modelling methodology. The model is structured via four subnets; each being structured in conjunction with two individual networks. In this chapter, the decoding of the inputs into the model, the training sets, the relation between the inputs and the outputs of the subnetworks, the cognitive decision layer, and the transcription of the syntactic output to the semantic level (i.e., the action) are summarized. At the end of each mentioned subnetwork, a semantic pointer is used to decode the syntactic output into verbal information. Therefore, the model is able to output some elementary sentences, such as "valid target, employ a missile" or "invalid target, abort."

In nature, "noise" exists inherently; any cognitive system reacts to the noise in its own behaviour. In this context, noise refers to the signal noise, which can be observed in the sensory/perceptual layers or globally all around the perception-cognition-action loop. The noise resilience of signal processing systems is a critical topic of concern in the related literature. Chapter six contains a survey and a modelling study of the noise effects in SNNs. Neuron and network-level examples are distorted with noise, and the effects are reported to better understand what might be expected from an SNN under noise. The first paragraphs of the results, discussion, and conclusion chapters provide a commentary on the effects of noise on spiking neurons and SNNs.

If one neglects the necessity of the embodied input, i.e., the seamless flow between sensory/perceptual and cognitive faculties, the cognitive system can be defined by resorting to alternative methods. The employed method can be symbolic or connectionist, cognitive or control theory based, Bayesian or deterministic. A charming variety of potential methods and approaches enriches modern cognitive science and robotic studies. On the other side, the alternatives to an SNN approach differ in some important aspects. First of all, symbolic architectures are left out of the scope of this study since, symbolic applications allows the user to implement every functionality in a symbolic architecture when providing recursive functionality and if-then possibilities, even complex mathematical functions can be solved using computational capabilities. Additionally, while capable of implementing every functionality, the method is not addressing problems of adaptability and environmental or spatial-temporal continuity. However, high-level decisions are much more amenable to graph methods such as decision trees,

which are outputs of cognitive selections between predictable sets of options. The final chapter discusses symbolic graph methods, such as decision trees and random forests, with SNNs regarding their biological plausibility and robotic applicability. The chapter argues for the view that SNNs are more applicable to robotics and far more biologically plausible than symbolic methods.

Despite all the above content, the output of the model may always have the probability of being not correlated with the decisions made by the human agents. This aspect needs to be studied carefully to bring scientific plausibility and acceptability to a proposed cognitive model. Additionally, the pilots may not always agree among themselves about the interpretation of a given situation and the resulting decisions. Therefore, a survey of multiple pilots on the model outputs may be beneficial to understand the variations among human pilots' high-level decisions and evaluate the model's success in the eye of human pilots. Therefore, the final chapter provides the comments of the pilots on the scenarios and a comparison between human agent decisions and the model outcomes. Agreements and disagreements between the pilots' comments and the model's output were analysed and discussed to conclude the study.

In the embodied and situated cognition context, the perception-cognition-action loop is mostly used for low-level robotics applications such as computing manoeuvring decisions or solving sense and avoid problems. On the other hand, as summarized above, this dissertation provides an embodied, high-level pilot decision-making model implemented using SNNs, compares the SNNs' aerial robotic applicability with symbolic graph methods, investigates the effects of the local and global noises in the SNN models, and provides a comparison of the human agent decisions with the model outputs. The model is focused on high-level decisions that precede the lower-level decisions in the chronological order of an aerial engagement. Therefore, the developed model can be classified or categorized as an embodied model of hierarchically organized, high-level pilot decisions, so the terminology of high-level decisions will be used throughout this dissertation.

CHAPTER 2

2 A REVIEW OF METHODS IN COGNITIVE MODELLING AND PILOT MODELS

Various methods and approaches have been developed to model human perceptual, cognitive, and motor skills at behavioural and neural levels. This study's focus is mostly on providing an embodied and connectionist pilot model to provide higher-level or strategic decisions. The modelling efforts for the pilot as a topic can be classified into four major groups; control theory models, symbolic architectures and other symbolic methods, connectionist methods (classical neural networks), and the neural engineering and neural field-based models that are considerably closer to the biological substrate of the human mind. It shall be noted that the third and fourth groups can be distinguished from each other with the properties of their modelling units. The third group employs a very abstract idea of a network where the modelling units are basic summers and very different from the last group's biologically inspired computing units, such as spiking neuron models or neural fields. These two major groups are indeed close relatives of the family of neural networks. Understanding these specific approaches in terms of their interrelationships and their historical context may lead to a better understanding of the value of the modelling approach undertaken in this study.

2.1 Control Theory Models

Firstly, control-theory based models have relatively less relevance to cognitive science. In such models, the pilot is represented as a transfer function either in time or frequency domains. The most well-known and frequently used models that served as a source of inspiration for the relatively newer models can be listed as follows:

- The Crossover Model, by McRuer and Jex (1967) in frequency domain with parameters *jw* and by Blakelock (1991) being the same model in Laplace domain with variable *s*.
- The Paper Pilot Model, a further developed version of McRuer and Jex (1967) with a performance-based parameter estimation method and the addition of a second-level control loop (Anderson, 1970).

- The Optimal Pilot Model, Pollard (1975), which aims to provide a task independency superior to Paper Pilot and brings additional units such as noise or displays.
- The Generic Pilot Model, by Johnson and Pritchett (2002), which is very similar to currently used dual loop autopilot models.

Control theory models are products of great engineering efforts and useful tools for aerospace development studies. The very first examples are used in the development of the Vertical Take Off and Landing Aircraft specifications regarding the checklists and questionnaires filled by conventional aircraft pilots (Blakelock, 1991). Figure 1 contains Johnson and Pritchett's Generic Pilot Model as an example.

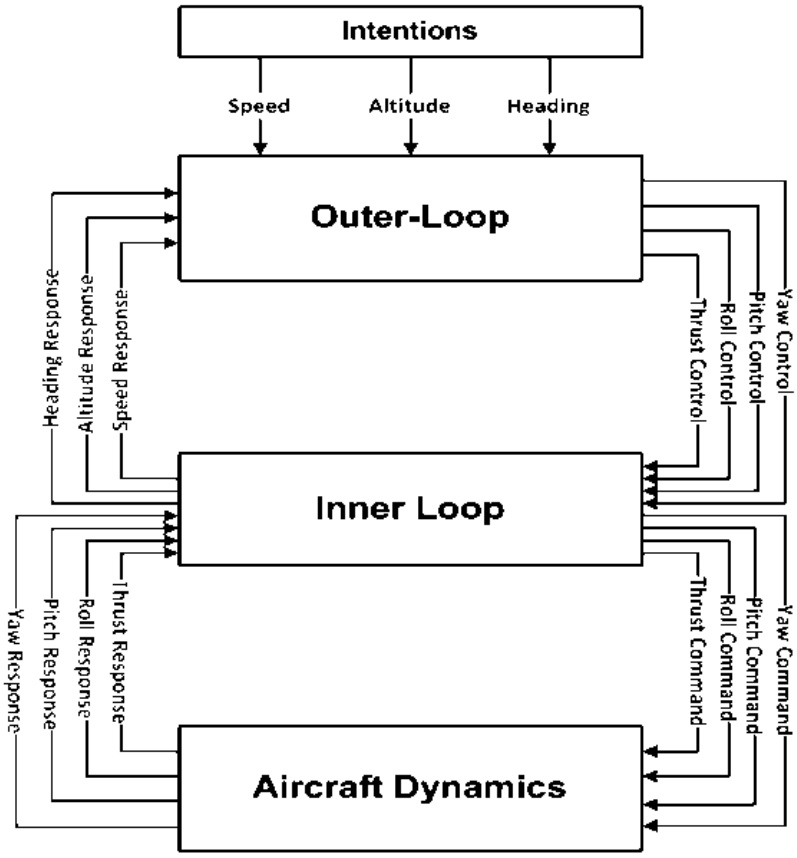


Figure 1. A representation of the Johson and Pritchett's Generic Pilot Model (2002) as represented in Kaygusuz (2015).

These models or all similar control theory based models lack focus on specific cognitive attributes and do not have such a claim or purpose[4]. On the other side, there are various studies, especially by Hutchins, aiming to understand the pilot's cognitive functionality, namely Information Flow models, where the pilot is represented as a continuously

[4] The interested reader may refer to Kaygusuz (2015) for further information and mathematical foundations of control theory and information flow models. Especially given reference contains a summary of various modelling approaches.

interacting and fully conscious disembodied agent, and the flight deck or cockpit displays are observed as an interaction layer between the aircraft and the pilot (Hutchins, 2000; Hutchins & Klausen, 1996).

According to Eliasmith, the primary problem of cognitive modelling is defining the transparent and vague relation between the external and internal realms of human cognition (Eliasmith, 2013), while the pioneering name of the Information Flow Models, Hutchins (2000) describes the flight deck as the interface between the pilot and the aircraft systems. The control theory based models given in this section as well as most models provide a mathematical understanding or pretending of pilot behaviour. Oppositely, the information flow models do not contain such approaches. Although such models neglect mathematical representations of the behaviour in most examples, they still provide useful and valuable conceptual structures for understanding pilot behaviour from an information-processing approach and may be a basis for strong engineering models. Given in below Figure 2, Hutchins' model may provide all components of an engineering model pretending as a pilot.

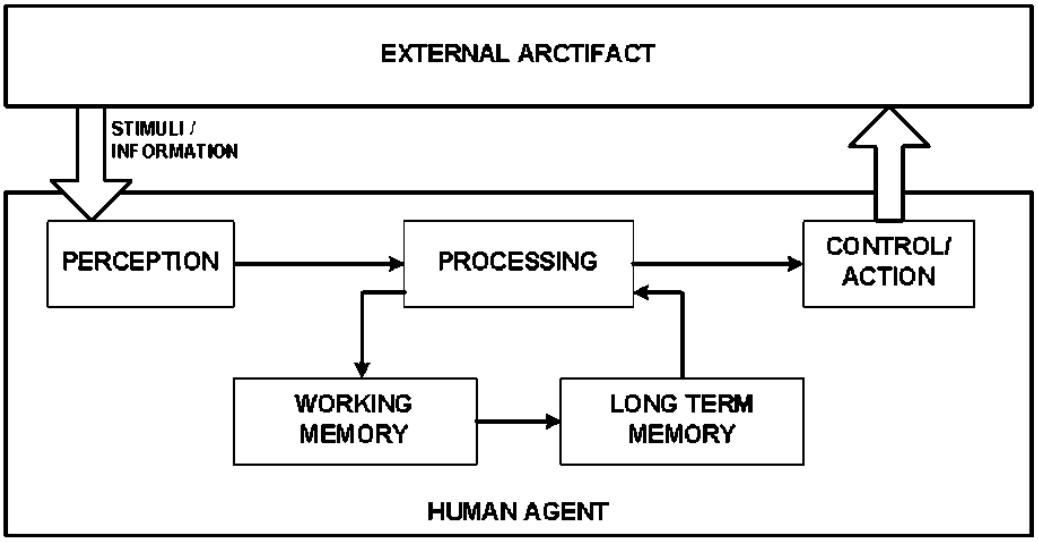


Figure 2. Hutchins' Information Flow Model from Kaygusuz (2015).

2.2 Models Based On Symbolic Architectures

Following the control theory and information flow models, it should be noted that later groups are mostly focused on not only understanding the behaviour but also simulating the human agent's cognitive faculties. The very first group is the symbolic models or symbolic architectures. Symbolic architectures use functional blocks of equations or faculties set to simulate or form cognition. The idea goes back to early studies of Alan Turing and Newell & Simon and, lately, David Marr (Anderson and Taatgen, 2009). The Turing Machine (Turing, 1950) and the General Problem Solver (Newell and Simon, 1963) may be considered the first examples of this approach.

An "architecture" can be a schematic representation or hierarchical abstraction of the faculties' distribution on the blocks or units of a system and the relations such as interfaces between those blocks. The architecture can represent different aspects of a system, e.g.,

physical architecture which shows the physical organization of the system, or functional architecture which shows the functional organization. An interface defines the relationship, functional, physical, or electrical, between units of the system and the external interactions. Interfaces need to be appropriately identified in an architecture so that the information flow can be described.

In the modern era of cognitive architectures, SOAR and ACT-R can be considered two important exemplars of this paradigm. The SOAR (States, Operators, and Reasoning) (Laird, Newell, and Rosenbloom, 1987) architecture is an extension of the General Problem Solver, whereas ACT-R (Adaptive Control of Thought-Rational) is grounded on Anderson and Bower's work on the human associative memory (HAM). Although ACT-R and SOAR cognitive architectures have been widely used in the cognitive modelling literature, there are also notable alternatives, such as EPIC, CLARION, 4CAPS (Wray and Jones, 2006). Anderson et al. (2004) provide the details of the modules contained inside ACT-R symbolic architecture. ACT-R provides a framework for modelling cognitive functions using a symbolic approach. The paper contains details of the modules such as the perceptual-motor module, declarative memory module, procedural memory, and goal modules. Each module is implemented using functionalities included inside ACT-R. Equations of the models are given, and an experimental comparison of the model performance is summarized for an anti-air warfare controller. Symbolic architectures provide an alternative approach for modelling cognitive behaviours. Nevertheless, ACT-R's on-board implementation to a flying machine would be quite difficult due to the limited possibilities for integrating ACT-R models with embedded software tools.

SOAR is another cognitive architecture that can be characterized as a pure symbol processing system, which is more rule-based and domain knowledge oriented in comparison with ACT-R (Lehman, Laird, and Rosenbloom, 2006). A key application of SOAR in aerospace is the TAC-Air-SOAR model, where a knowledge base in the form of domain expert-derived rules (more than 8000 rules) is stored. Domain expert rules are applied to the operators and verified by the actors. SOAR could be a valuable resource in the control of aerial mission scenarios due to its capabilities in composing rule chains via means-ends analysis and learning new rules through a chunking process. However, representationalist architectures such as ACT-R and SOAR require the embodied control functionalities to be reformulated in symbolic form so that rules can be applied on them, which ultimately leads to the grounding problem where the symbolic content loses the embodied and situated aspects of dynamic control. For that reason, so far, TAC-Air-SOAR has been only deployed in a flight simulator environment.

In this study, the goal is to develop a high-level or strategic decision model using spiking neurons, as described before. The difference between SOAR and ACT-R can be clarified with the functional decomposition of both approaches. In fact, ACT-R has a subsymbolic dimension while containing a Bayesian network and a rewarding functionality different than the SOAR. In a symbolic architecture, components of the system are faculties where syntactic operations are performed via symbol processing. Despite being cognitively meaningful and pretending the human mental faculties, at the low level, the components are just blocks of symbol processing. The functional blocks can be seen in Figure 3. On the other side, a connectionist model is focused on devising the functionality from the embedded operations which are performed through the network.

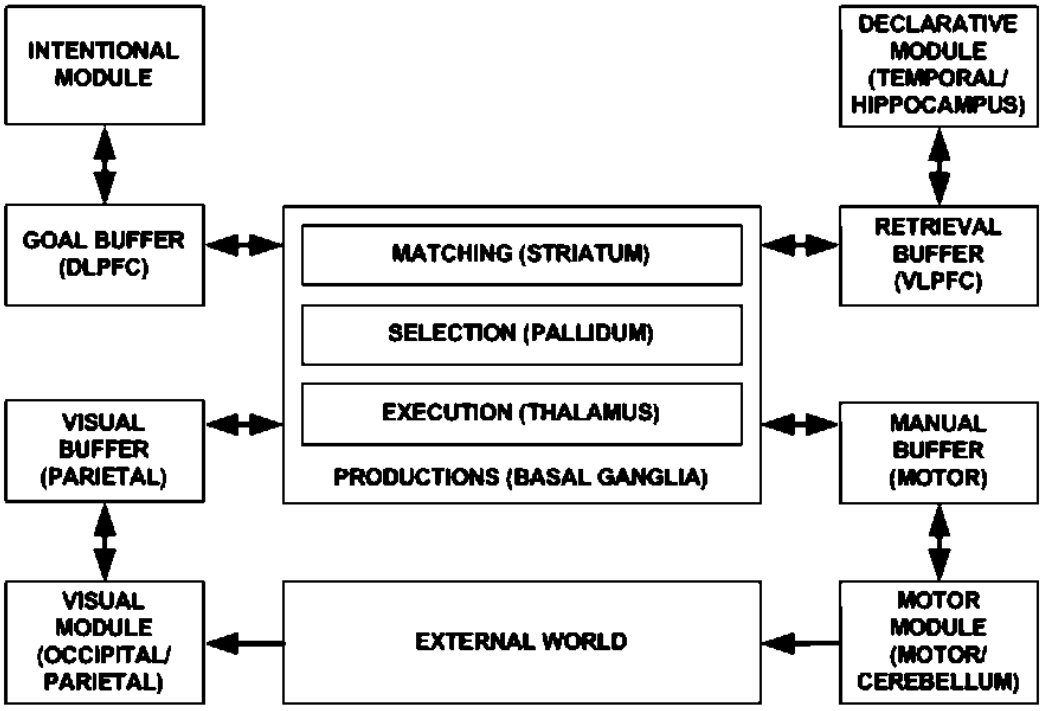


Figure 3. Architecture of ACT-R 5.0 by Anderson et al. (2004).

In Figure 3, it can be clearly observed that each functional block is inspired by human cortical organization. On the other hand, the connections between blocks and each block's internal content are primarily symbolic operations.

Symbolic models do not require to be solely function-oriented. There are graph-based models, namely symbolic graphs such as decision trees or forest types, which are symbolic in terms of processing but graph-based in terms of their functional approach. Such methods will be described in chapter 6 of this study.

2.3 Neural Network Models

Finally, parallel to the history of Cognitive Science, there has always been a school of connectionism, which claims that cognition arises from the network (Fodor & Plyshyn, 1988). From the early idea of perceptron, neural networks have been widely used to simulate or model human cognition. For the specific task of modelling the pilot, there are neural network-based flight control or autopilot models such as Enns and Si's (2004) helicopter flight control with Neural Dynamic Programming and Kaneshige and Burken's (2008) neural in-flight control model. Despite being s-domain control models implemented in neural networks, these studies provide cognitive functionalities such as "intentions" to the table (Kaneshige, Bull, and Totah, 2000). Kaneshige, Bull, and Totah (2000) describe a neural flight control algorithm without providing the aircraft's specific parameters. The main principle of the developed model is to use reference models for handling qualities

characteristics of the pilot model and compare these reference models with the model outputs to obtain reference-like characteristics. The model is, in fact, a control circuitry with feedback mechanisms. The only neural parts are the neural networks that are used to generate adaptive error functions in the feedback loop of the controller. Three different feedback neural networks are used, each respectively belonging to different Euler axes of the aircraft, namely P, Q, and R. The method uses a very simple idea of using reference behaviour models for control behaviour targets of the pilot model. This part of the idea is very common in complex robotic control. Similar ideas are also used in Dynamic Field Theory (DFT) models.

De Souza et al.'s (2005) study proposed another neural network based model to simulate aircraft behaviour. In the classical methods, an aircraft is generally modelled using the 6 DoF (degrees of freedom) approach, where aircraft Euler angles and moments are solved with equations of motion. The equations of motion are differential equations showing aircraft properties under external loads. De Souza et al. employed a classical 12 equations 6 DoF model where the differential equations are solved using a 4th-order Runge-Kutta integrator. After achieving a satisfactory aircraft simulation, a recurrent neural network (RNN) is trained using the outputs of the 6 DoF model. After the training, the final neural network is tested with various conditions to check whether the outputs are consistent with the original model and if the RNN can correctly estimate the aircraft envelope for the conditions it was not previously trained on. The manuscript shows that a 6 DoF aircraft model can be trained with an RNN.

In a related study, Muratov, Lakhman, and Burtsev (2014) used a method named neuro-evolution to model a controller with sigmoid neurons. The model is formed with less than 25 neurons, while each neuron exhibits a sigmoid characteristic. Neurons are trained for a multiple-goal navigation task using a reward-based training method with a reward function taking older goals also into account. The training method is similar to genetic training, where the parents are not killed. In the end, the resulting task performance demonstrated that the simple neuro-controller perfectly performs the navigation tasks. Promising results were obtained in a simple multi-goal 2D navigation task.

Connectionist models have been utilized in the context of decision-making and response compatibility, which can be related to pilot performance. For instance, Zorzi and Umilta (1995) applied a neural network based model of the Simon task, which is a popular spatiotemporal cognitive task focusing on the response-compatibility phenomenon. The Simon effect refers to an improvement in the response time of a subject when he/she is asked to respond to the spatial location that is compatible with the presented visual cue. When the subjects are demanded to respond to a spatially incompatible stimulus, their response time slows down. The network model of the Simon task consists of simple neuron models where weights can be positive and negative (inhibitive) to create a competition between the left and right nodes of the task. Each node's activation is a weighted sum function of its current state and the effect of the external input. The network is trained by giving the potential stimulus and desired response at the same time and adjusting the weights. After the first run of the simulation, regarding the results, the network's sparse connection is improved with all-to-all connections for better results, and the network is retrained. The effect of change from sparse connection to all-to-all connection is creating a competition in between output nodes when inhibitory weights are used. The simulation results suggested that the method described above could generate response characteristics compatible with the Simon effect. This study can be considered another indicator of how simple connectionist models can be used to simulate complex behaviours. Connectionist models offer important advantages in simulating situated/embodied processes due to their fault-tolerant, distributed, and dynamical nature.

Yıldız, Agogino, and Brat (2014) use the Game Theory to make tactical-level decisions on an airfield. The airfield is a simple grid of squares where simple aircraft can move diagonally or straight. Aircraft are simple particle mass models under velocity control. While the setup is quite simple, the method of using game theory with reinforcement learning provides a robust training where in the end, the aircraft in the airfield are preventing themselves from collisions. The simulation has been run with various scenarios where in one of them, a self-navigating (maverick) aircraft has entered into a perfectly structured airspace. The authors use a pilot reward function reinforcing the prevention of separation violation, decreasing the probability of separation violations, staying on its pre-planned trajectory, and finally minimizing the effort. Each of these properties is assigned a reward and the system is trained by using the reward function of each action. The manuscript also uses the Level-K theory where; Level-0 intelligence makes straight flight in its trajectory, Level-1 intelligence makes sense-and-avoid action assuming other aircrafts are Level-0 agents, and Level-2 intelligence makes complex moves assuming other aircrafts are Level-1 agents and will make sense-and-avoid actions. Musavi et al. (2017) is a very similar study to Yıldız, Agogino, and Brat's (2014) study. The main goal is to make a game and Level-K theory-based decision model trained with reinforcement learning. The model's intention is to find a safe method to integrate unmanned aerial vehicles into the national airfield, which also includes manned aircraft.

Although not related to the aviation domain, Backhaus et al., (2013) utilized Level-K theory-based strategy development in a similar way in the context of a game simulation over a smart grid (i.e., a power distribution network). The attackers are randomly attacking the network, and defenders are unaware of the timing of the attack. As a result of this, the defenders are trained to make probabilistic assumptions on the timing of the attack. Therefore, in some aspects, this game has similarities with aerial warfare, where the patrolling aircraft (defenders) are unaware of the attack's direction and time. This article pointed out that an optimal defender would be capable of attributing various levels of intelligence to the attacker and developing different strategies to counter each level.

Deep learning models extend the classical connectionist models that can accommodate more complex network structures and non-linear relationships. LeCun, Bengio and Hinton, (2015) provide a review study summarizing the fundamentals of deep learning, starting from the very basics of supervised learning, backpropagation, and convolutional networks. The main advantage of a deep learning network is that, if feature engineering is performed properly, such a network can learn any function most of the time. Backpropagation training methods generally perform with great success on multi-layer networks. In rare instances, local minima pockets can encapsulate the solution before reaching global maxima points. Most of the current deep-learning examples focus on image-processing applications. Such networks are typically used in tandem; for instance, after feature extraction via a convolutional net, a recurrent neural network (RNN) can be used for the tagging of the extracted features. Therefore, it may be said that deep nets are used in most applications as a joint network of multiple types.

2.4 Models Employing Neural Fields And Neuron Models

As mentioned in the first paragraph of this chapter, classical neural networks and biologically inspired methods are very close relatives of a family of Neural Networks. Neural network based models can also be classified under three main groups. The first group is the above-mentioned classical network models, where neurons or nodes are not

biologically inspired. The second group contains a different level of granularity. Dynamical Field Theory (DFT) uses mathematical equations by Amari (1956) representing the electrical activity generated by a group of neurons. In other words, although DFT uses neural fields of the neuron ensembles as modelling units, they do not contain the neuron itself. The strength of the DFT approach lies in its attractor dynamics that enable the modeling of cognitive faculties such as memory formation or motor action control. A considerable literature on the application of DFT on autonomous robotics exists; where some examples are Erlhagen and Bicho, (2006), Schöner, Bicho, and Mallet, (2000), and Kaygusuz (2015). The final group of the neural networks and the second group of the bioinspired models are the neural engineering models, which will be introduced in the subsequent paragraphs.

For all of the above approaches and some other biologically inspired models, Kaygusuz (2015) provides a detailed categorization and summary of the mathematical foundations. On the other hand, this study will be using a different approach called the Neural Engineering Framework (Eliasmith & Anderson, 2003) to model the pilot's high-level or strategic decisions. The following paragraphs provide a summary of some of the major modelling studies leading up to NENGO and its successor, the Semantic Pointer Architecture (Eliasmith, 2013).

Feldman and Ballard (1982) provide a detailed review of connectionist models and their properties, including technical and philosophical details. The authors classify the previous models, mostly representationalist, under a category named information processing models. Due to their requirement for enormous processing power and resulting latency, they find connectionist models more beneficial where a single neuron does not process the information, but the overall network makes the processing. This work belongs to a period where the neural field or neural engineering approaches had not yet been developed. The authors pointed out the possibility that cortical electrical activity could be modelled beyond the level of granularity of single neurons (Feldman & Ballard, 1982, p. 209). Similarly, during that period, research on neural networks was also at a halt, which was about to regain momentum with the eventual publication of the Parallel Distributed Processing (PDP) books (Rumelhart et al., 1986). Feldman and Ballard (1982) proposed an abstract symbolic processing language to demonstrate logical operations between units of computing. Despite the immaturity of neural networks, the authors proposed notions such as stable states and convergence for mental faculties, especially for decisions. Finally, Feldman and Ballard also proposed winner-take-all (WTA) networks as useful methods for cognitive decision modelling.

Another key conceptual leap in this era is the proposal of the interactive activation and competition (IAC) model (McClelland, 1981; McClelland & Rumelhart, 1981). The IAC model brings the concept of free connotation or association of subject properties in human memory. The model depends on the idea that similar properties excite relevant neurons and competing properties inhibit each other to suppress activation. Using this approach in a fully connected network, activating one property may freely excite agents holding similar properties in their neurons.

Johnson and Busemeyer's (2010) review summarizes various cognitive models of decision-making under risk and uncertainty. Among the mentioned models, there are purely mathematical ones with no biological plausibility, as well as symbolic/representationalist and connectionist models. Some models intend to use probabilistic approaches to justify decisions under uncertainty (e.g., rank-dependent theories, configural weight theories). Additionally, there are heuristics models, such as Thorngate's Heuristics for Gamble Forms, The Adaptive Decision Maker Hypothesis, Gigerenzer's Adaptive Toolbox, and decision field theory. The current thesis study will

utilize a connectionist approach for modeling the decision layers, such as the Leaky Competing Accumulator Model, Coherence-Seeking Network Model, or the IAC model.

A leaky competing accumulator model contains the alternative choice options' competition in a network, where the preference may reach a holistic result for similar cases in time. The idea of competing options for a decision seems very applicable to the current study. In this study's task, the pilot model implemented may decide whether to attack, flee or ignore the opponent. Such a decision may be assumed as the result of competing options, namely "attack," "flee," "ignore," and "chase," similar to the leaky competing accumulator model. But such vital decisions (in fact, a matter of life and death) do not accumulate leakily to the same decision in general. Therefore, what is needed for the decision layer is not a leaky model but a robust model incorporating competition. For that reason, a modified IAC model is used for this task in this thesis, which will be summarized in the next chapters. Alternatively, a fuzzy neural network should also provide similar competition since fuzzy logic already contains the idea of competition in a network.

Most dynamical field theory (DFT) studies focus on spatiotemporal tasks. Neural fields are clearly quite good in interaction with low-level sensors and spatiotemporal data manipulation tasks. In particular, Cuijpers and Erlhagen (2008) show that neural fields can be used to implement complex and stochastic functions. The authors encode the log-likely hood and the prior probabilities into two different neural fields and then superimpose these two fields to obtain the posterior probabilities. Since the kernels for both fields are not the same, the superposition leads to a nonlinearity. While the idea of super-positioning neural fields is quite simple, the nonlinearity of Amari equations is not easy to solve. Therefore, the remainder of the study focuses on providing a good estimation of the result. In short, Cuijpers and Erlhagen's findings shows that complex functions can be implemented via neural fields.

Erlhagen and Bicho, (2006) provide the basis of a robotic architecture based upon the Dynamic Neural Fields or Dynamic Field Theory (DFT), where cortical areas are represented with neural field equations. Since classical robotic architectures using sensors on-board generally do not contain methods using past and future states of the environment, the action decision simply depends on the present state of the observation. In their study, Erlhagen and Bicho (2006) provide memory-like neural fields to hold the past states and pre-set neural fields to estimate potential future states. Neural fields are in the form of Amari equations (i.e., the activity of the neural field depends on the rate of change of the activity, leading the system to reach an equilibrium state, the resting state, input functions, and the neural activation kernel), and steady state solutions are provided.

Bicho, Mallet, and Schöner (2010) utilized the position acquired by low-level sensors as a dynamic field (a differential equation) where steady states represent desired values and unsteady states represent locations of obstacles in the environment. Unlike Monteiro and Bicho (2008), this manuscript is not making formation control but focuses on an autonomous vehicle that simply navigates. The main weakness of current neural field-based modelling and robotic processes is that a robust reinforcement learning method is not yet developed.

Sandamirskaya and Storck (2014) provide a neural field-based learning method to partly address the issue of learning in DFT. Although the learned behaviour emerges via tuning the connection weights in connectionist models, this approach points out memory modules as blocks of learning. Pre-shaped neural fields are used as memory blocks, and those do not decay over time, even in the absence of input. The main idea here is to compare an input with the pre-shape held in the memory and obtain a result from superposition, subtraction, or interaction of both.

Additionally, the authors use zero-dimensional neural field concepts to obtain decision nodes where the decision is a Boolean function or simply a go-no go decision. Both concepts may be useful in a purely neural field-based model. One of the most important cognitive functions is sequential task following. Generally, humans don't perform many tasks hierarchically or serially. Therefore, a strong cognitive architecture shall be able to provide a sequential task-performing capability. The authors of this manuscript provide an architecture to perform sequential task generation and ordering for neural field models.

Neural field models' ordinal dynamics depend strongly on ordinal node concepts. Those are bistable zero-dimensional neural field nodes, which have mutual inhibition in between. In other words, only one ordinal node can be active at any time. Therefore, for any task in the queue, there shall be an ordinal node, each starting the connected task while active. On the other hand, when a task is completed, a Condition of Satisfaction (CoS) node will be able to acquire this fact by comparing a pre-shape with a connected field's output. The output of the CoS node will activate the next task in the series and suppress the previous one. The concept of CoS is very common on DFT, and ordinal dynamics is widely used in serial task ordering (Sandamirskaya and Schöner, 2010). Up to Kazerounian et al.'s (2013) study, all learning applications on DFT were strongly connected to the pre-shape or memory trace learning. However, this is not enough for a strong task learning demanding environment. In this specific study, the lack of learning methods in DFT, is mostly filled with modified reinforcement learning (RL). Classical RL uses reward functions for each action that the agent performs in the environment, and if the reward function returns higher values, model parameters are improved. The study applies a method named Dynamic Neural SARSA (DN-SARSA(λ)) algorithm to train the architecture. The method mostly depends on a temporal difference value computed in each action.

Mainly, out of the neural field architectures, DN-SARSA holds a table of relations between intention nodes and CoS nodes. Each time an intention node and CoS node lives transition (from one EB to another), the state-action field dedicated to this couple is activated via a step function. This state-action activity is connected to another field, namely TP (transient pulse) field. TP fields are formed of both inhibitory and excitatory fields. Inhibitory fields are used to eliminate the persistence of previous epochs. Each time an input is acquired from the state action, TP shapes a resulting field, and if it is above a threshold, an eligibility trace is shaped. Eligibility trace is the working memory for the learning task of the reinforcement learner (RL) of DFT architectures. In the end, RL is a complex structure in DFT mechanisms.

Koch provides (1999) details of the biophysics of neurons, yet Chapter 4 will provide a simple approach to rate models and common neuron models such as integrate and fire neurons. The method starts from the capacitance equation, where the current in a membrane is the function of the derivation of the voltage in it. On the other hand, such a capacitance system includes leakage currents which are linear functions of voltage and resistance. Therefore, the current is a function of differentiation of the voltage with time and the voltage itself. This is the root of the rate models' idea.

In modern cognitive science literature, there are studies employing various types, methods, frameworks, and granulation levels of modelling. Therefore, connected to the context, there are also different software tools containing various methods and granulation levels employed as a development environment. Some require high costs and contain various pre-developed libraries like MATLAB/Simulink, and some are open source with varying levels of library readiness, like CEDAR (Lomb et al. 2013) of Ruhr University Bochum or WEKA of University of Waikato (Holmes, Donkin and Witten, 1994). Of course, these packages each having their own properties and their own goals are also enriching the effectivity and diversity in the domain. A very recent tool containing an embedded solver

for spiking neuron networks is NENGO (Bekolay et al. 2014). NENGO is used in this study to build a spiking neuron based network to achieve a fighter pilot's high-level decisions model. In this chapter, the Neural Engineering Framework behind NENGO and previous studies employing it will be summarized.

2.1.1 The Neural Engineering Framework and NENGO

All of the studies mentioned in the previous sections of this chapter will employ either integrate-and-fire (INF) or Leaky integrate-and-fire (LIF) neuron models to build the network. Before giving the details of the neural engineering framework, the idea behind spiking neuron models such as INF and LIF is given below.

According to Kaygusuz (2015) spiking neuron models can be used in lieu of complicated conductance-based neuron models when a simple neuron model with relatively good output quality is needed. A spiking neuron model is a classical neuron model generating a spiking behaviour as output activity while neglecting the biologically motivated ionic conductances. In other words, a spiking neuron model approximates the conductance-based spiking behaviour via a mathematical equation without using biological channels. Therefore, spiking characteristics can also be achieved without using conductance-based models.

Above mentioned spiking neuron models are the integrate-and-fire neuron (INF) and the leaky integrate-and-fire neuron (LIF), both of which use the below capacitance equation. The difference between the INF and LIF is that the leakage function F(V) is linear in a LIF. So each LIF is an INF but with a relatively simpler F(V).

$$C\frac{dV}{dt} = -F(V) + I(t) \qquad \text{(Equation 1)}$$

On a network of LIFs, the model provides a spiking behaviour between a resetting voltage and a spike threshold. In other words, a LIF neuron outputs a rising spike starting from the resting potential, rising with external stimuli, up to a specific spike threshold, then decay very rapidly down to a resetting voltage that must be below resting potential. This oscillation between these two values results in the spiking behaviour of the famous spiking neuron.

By integrating equation 1 under a constant input current I_0; a very simple output given in Equation 2 is obtained.

$$V(t) = I_0 + (V_0 - I_0)e^{\frac{-(t-T_m)}{C}} \qquad \text{(Equation 2)}$$

In Equation 2, the T_m represents the spiking time, C is constant, I_0 is a constant input and the V_0 is the initial condition of the $V(t)$. $V(t)$ value reaches to I_0, if the t increases to infinity passing T_m, so it can be deduced that the I_0 is the upper limit of $V(t)$. Therefore, when the input I_0 is below the spiking threshold h, there is no spiking behaviour. If the I_0 is above the threshold, i.e. the given input is strong enough to penetrate to the system, then spiking characteristics arises periodically with an interval between the spikes. Output $V(t)$

augments until *h* (since upper limit I_0 above the *h*), then reduces down to V_0 and starts to augment again. The increasing duration up to next decrease is given in Equation 3.

$$T = C \ln(\frac{I_0 - V_0}{I_0 - h}) \quad \text{(Equation 3)}$$

Kaygusuz (2015) provides detailed information on all well-known neuron models (rate, biological, spiking, or basic), and Bresloff and Coombes (2000) give detailed information on INF neurons and networks.

A neuron model is expected to produce a voltage type output activity value as the result of its characteristic equations set. As expected, the input to the next neuron's synaptic input needs to be the previous neuron's postsynaptic voltage. So an equation representing the current as a function of the synaptic gap or synaptic conductance is possible as in Equation 4:

$$I_s(t) = g_s s\,(V_S - V) \quad \text{(Equation 4)}$$

Equation 4 defines the synaptic current $I_s(t)$, g_s is the synaptic conductance and *s* is a gating variable, where voltage terms, *V* is the input voltage and V_s is the reversal voltage. On following equations 5 and 6, two very common methods to approximate the shape of the current is given:

$$P(t) = \alpha\, e^{-\alpha t} H(t) \quad \text{(Equation 5)}$$

$$P(t) = (\frac{1}{\alpha} - \frac{1}{\beta})^{-1}[e^{-\alpha t} - e^{-\beta t}]H(t) \quad \text{(Equation 6)}$$

The above functions are shapes for postsynaptic response current approximation. The first is an exponential decay with a Heaviside function (given in Equation 7), where α is the decaying rate as a function of time. The second occupies two exponential parameters α and β, which are used to determine rise and fall durations. Below Equation 7, the H(t) is represented.

$$H\left(u(x',t)\right) = \begin{cases} 0, & t < 0 \\ 1, & t \geq 0 \end{cases} \quad \text{(Equation 7)}$$

NENGO uses above neuron models in very fundamental level, while supplying a framework to build further larger models. The Neural Engineering Framework (NEF) which is the idea behind NENGO, is a general methodology that allows the researchers to build biologically plausible, neural models (Stewart, 2012). NEF can be assumed a neural modelling environment or framework with following capabilities:

1. As a modelling environment and compiler, NEF can solve the neural models outputs for user specified neuron properties, values to be presented and the functions to be computed and specific connection weights.

2. Different than many modelling tools, can be used to compute recurrent functions allowing the user to build up complex dynamics such as integrators, oscillators or recurrent filters.

3. While being weak in large scale network training, local realistic error driven rules can be applied for optimization.

To be able to provide a representation of a specific value, either scalar or vector, NEF uses LIF neurons each having specific tuning curve. This is very parallel to the neuroscientific methods which uses a specific tuning curve for each neuron to make it fire in a specific direction. NEF assumes a linear proportionality between the input current of a neuron and the value being represented. Therefore, each neuron fires to a different direction, resulting in the varying activity of different neurons for different inputs. In below Equation 8; G is the neural non-linearity, α_i the gain parameter for the neuron *i*, and b_i is the constant background bias for the neuron *i*, then neural activity for a given x is;

$$a_i = G(\alpha_i e_i x + b_i) \qquad \text{(Equation 8)}$$

Equation 8 can be applied to any neuron model in literature, such as LIF or rate models if the relation between the input current and neuron activity is strictly defined. NEF employs a basic approach to code this mapping into neuron tuning curves. See the Figure 4 for tuning and response curves of a 100 neurons ensemble.

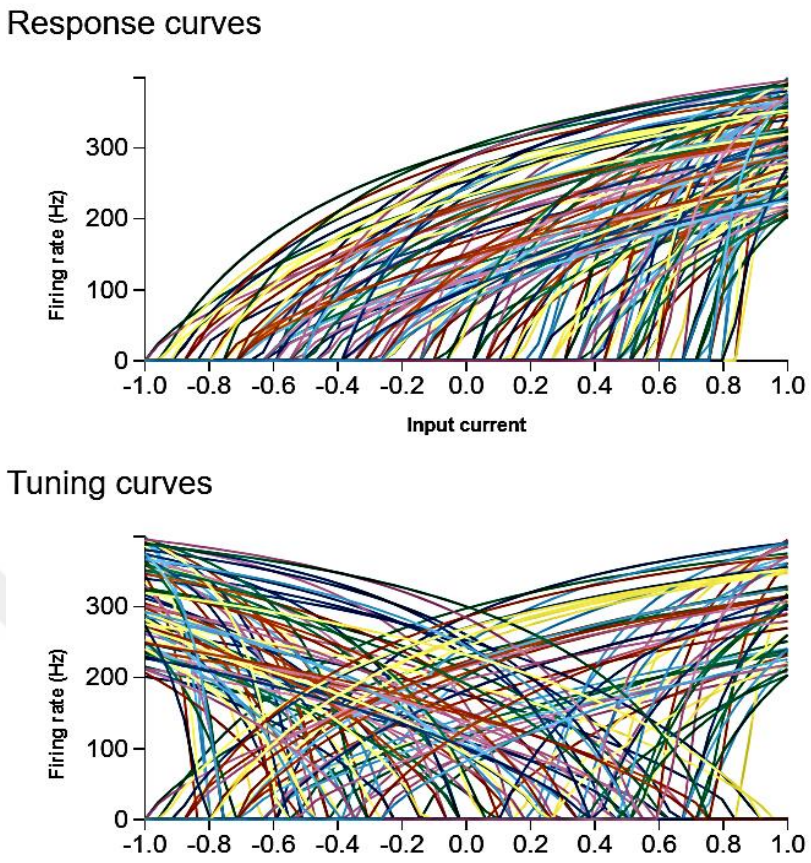


Figure 4. Response and tuning curves of a neuron ensemble.

As seen above, via Equation 8, a value can be encoded into spikes of an ensemble. On the other side, to be able to obtain readings from the network, values can be read out from the neurons to the output. Equation 9 gives a linear estimator d_i to decode out output value X from activity a_i.

$$X^v = \Sigma\, a_i\, d_i \qquad \text{(Equation 9)}$$

The d is a vector set that can be solved via a least squares minimization problem to reduce the error between X and the estimate.

A classical neural network, even when having no hidden layers, has in fact multiple stages, first, the inputs, connected to the weights, the second stage of inputs (multiplied with weights), and outputs. This can be seen in Figure 5 extracted from SimBrain, an open source neural network tool. The example is a very classical one in machine learning lectures. Here, it is used as an example of classical neural networks.

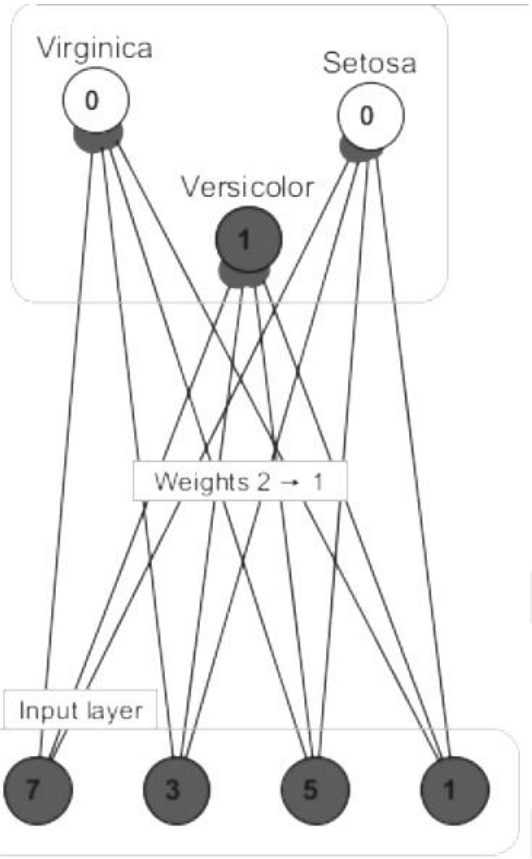


Figure 5. Inputs multiplied by weights to reach outputs, a typical neural network example from SimBrain.

Since NEF uses the linear decoder set d for decoding outputs, weights are directly coded into the reverse transformation, so different than the classical neural networks (like the one in Figure 5), a NEF network is more direct, so every function can be implemented in this type of networks such as trigonometric functions or integrals and the famous XOR problem (Stewart, 2012).

NEF and its open source implementation NENGO allows the researchers to use the Semantic Pointer Architecture (SPA) to provide a decoding between syntactical operations via network to semantic derivation. According to Eliasmith (2012), NENGO employs SPAs as high dimensional vectors to provide fuller semantic content and claims that this functionality is already a continuous occurrence in the biological neural networks. In this study, SPAs are used to obtain semantic definition of the pilot's decisions arising from the network.

NENGO allows the user to define nodes as inputs or outputs. These nodes can be used via sliders to change the input or with lambda calculus inputs to define mathematical inputs. The GUI contains a Python interpreter to script functions. A sample GUI from python and a script sample in Python is given in Figure 6.

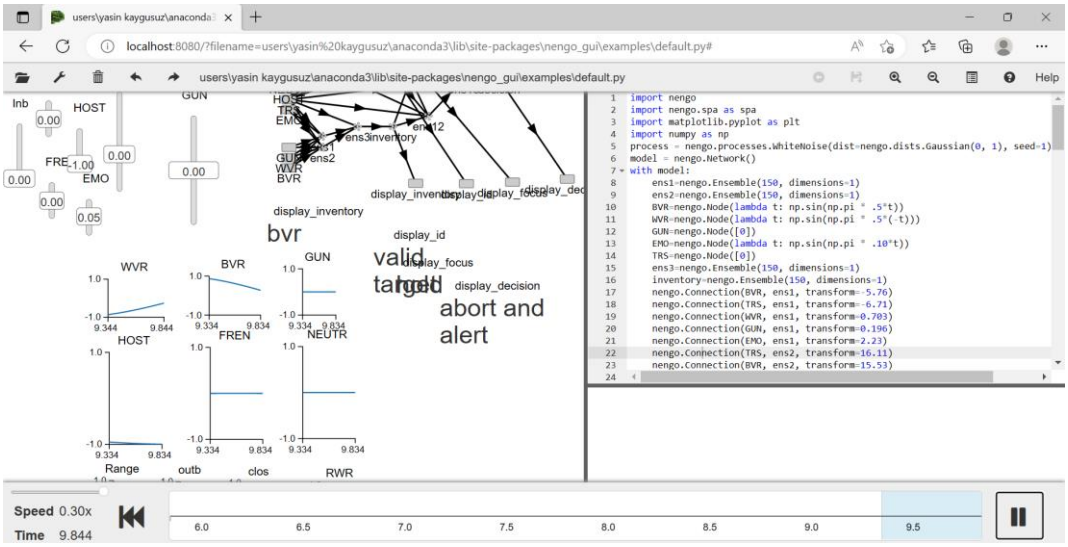


Figure 6. NENGO GUI and Python script sample for a basic model.

Figure 6 provides a screenshot of the NENGO graphical user interface. On the right-hand side, there is a Python interpreter embedded in the GUI. The left-hand side is a graphical environment where each ensemble or network property can be seen, and some arrangements are allowed. This tool is used for model development and testing purposes in this dissertation study. With its embedded solvers and models, the tool provides an environment that allows the user to focus on the model and neglect the low-level mathematics.

2.1.2 Artificial Intelligence and Robotics Applications Of NEF And NENGO

Neural Engineering Framework (NEF) and NENGO, as a software tool employing NEF, have various applications and evaluations in the literature. Bekolay et al. (2014) provide an open-source software tool for spiking neuron based computational models. On one side, those models are neuron-based models, are connectionist in some aspects, and are biologically plausible. On the other side, such models are quite complex and may not always be suitable for on-board applications such as UAVs. NENGO is an open software modelling environment with a scripting interface allowing the users to add their own Python scripts and uses spiking neurons as building blocks. The block parameters and characteristics, such as linearity or response time constants, are manipulated via menus. If NENGO could be run on board as embedded software, it would be a good alternative for on-board robotics applications and studies.

Stewart and Eliasmith (2011) built a spiking neuron based cognitive model using 150,000 Leaky Integrate and Fire (LIF) neuron models and tuned the model parameters using original values from cortical regions. The modules of the model are shaped with biological inspiration, and the represented cortical areas are associated with 18 expert rules for this specific task. The model includes the brain parts, namely the cortex, thalamus, and basal ganglia, each dedicated to the original function of the human agent. The model could

successfully perform the tower of Hanoi task. The main idea in the implementation of area-specific rules is to implement if-then rules in a connectionist way in a neural network of spiking neurons. In another study, Hurzook, Trujillo, and Eliasmith (2013) focused on achieving a biologically plausible visual cortex model to process motion information to deduce a target's velocity. The model contains V1, middle temporal area, and lateral intraparietal areas to process motion information from the visual sensor. This is a spiking neuron based model. To ease the computational complexity, the LIF (Leaky Integrate and Fire Neurons) models are used as building blocks. The application is used with Eliasmith's new approach, namely the Neural Engineering Framework (NEF). The study gives a mathematical summary of LIF neurons and vector estimations about the velocity using the population dynamics of neurons.

The cognitive aspects of regular and irregular verb segregation for English language users and child development is an old but important topic in cognitive science. Plunkett and Juola's (1999) manuscript takes its side in the famous past tense debate by developing a simple perceptron neural network to be trained with the error back propagation algorithm and to be used on the derivation of past forms of English verbs after training. The network contains 130 input, 160 output, and 200 hidden neurons. The authors intend to show that the dual route approach claiming separate profiles for syntactic and semantic developments is not meaningful. CHILDES and Brown's corpus are used in this study(Plunkett and Juola, 1999). In the end, the neural networks have been able to learn the correct inflection of 946 different verbs and 2280 distinct nouns with accuracy above 99%. The study shows the potential of neural networks in achieving complex task performance. Secondly, the study uses a binary interface between the neural network and the human interface; i.e., semantic and syntactic layers are properly interfaced, and both connectionist and symbolic structures are used together, given an appropriate vector representation for the words and their inflections. The context is not directly relevant to the study at hand, but the method can be employed in case of necessity.

In another related study, Hemion (2013) uses a neural field approach to provide a framework for cognitive robotic modelling. In particular, Hemion uses a simulation-based approach, namely "schemata" to analyse the external world to form an understanding of the environment. The main idea is to use an internal model of the external world to obtain a what-to-do analysis without creating any motor action. The idea behind Hemion's approach is the simulation theory. According to this theory, cortical faculties providing actions or observable outputs under normal conditions can also be used to run simulation models without creating any action during decision processes..

In the DFT method, researchers prefer to model the highway instead of modelling cars that use the highway to obtain the emerging traffic behaviour. In other words, in DFT the electrical activities of the cortical columns or slices are modelled and used as building blocks of larger models. Opposite to DFT granularity which is ensemble level, i.e. cortices or nuclei are modelled, one type of the most complex neuron level models is the spiking neuron models where spikes are used as the measure of the electrical activity. In the study given in Eliasmith et al. (2012) uses a 2.5 million spiking neuron models (equal to the size of a cat brain) under a framework named Semantic Pointer Architecture Unified Network (SPAUN) to generate a cognitive model that performs specific tasks. Eliasmith et al.'s model contains various simulations of human cortical faculties such as the premotor cortex or ventrolateral prefrontal cortex. Whatever the biological focus, the model provides a faculties organization and architecture containing visual input, information encoding and decoding, working memory, action selection, reward evaluation, motor processing, and motor outputs. Choo and Eliasmith (2013) mentioned SPAUN architecture is performing 28 different tasks which can be grouped under eight major task categories with no modification on its structure. Therefore, the architecture contains a level of flexibility and

generality among cognitive tasks. Eliasmith et al.'s (2012) model contained 2.5 million spiking neuron models. In the study by Choo and Eliasmith (2013), the authors focus on creating a high-level interpretation encoding to communicate with SPAUN to be able to reach a task independent cognitive model by achieving a task declaration interface. In other words, the main purpose is to reach a point where tasks different than the original eight tasks can also be declared to SPAUN with high-level communication. It shall be noted that, while their granularity levels are completely different, still both the DFT and SNN methods employs rate codes in the encoding of spatio-temporal values.

Up to all connectionist models that can be surveyed in the connectionist literature assumed the semantics would arise or emerge from the operations of the network.Similarly, representationalist approach assumes symbols or representations carry the semantic intuitively. Within the development of the cognitive science it shall be deduced that both approaches are not maturely defining the roots of the semantics and are not yet successfully providing a semantic backbone. On the other side, semantics is an important part of the cognition and modelling studies may also take this issue into account when possible. Both Eliasmith, Tang, and DeWolf (2013) and Eliasmith, Stewart, and Rasmussen (2013) propose a new approach, namely the Semantic Pointer Architecture (SPA) for constructing a link between encoded information in the network, the embedded operations and the relevant semantics. Regarding Eliasmith's approach, pointers can be connections between the syntactic operations performed via the neural network and a multi-dimensional semantic space. In other words, operations can be connected to their topics via pointers. Pointers can also be neural engineering transformations such as decoding. In this method, a SPA can connect a linguistic label field to a complex operation in a neural engineering framework or neural network, and the label field can change its content regarding the output or state of the network. For example, if the network is inhibited, the SPA can carry this output to the semantic dimension and change the label field content to "none," and when the network is functioning under excitation, the same SPA can change the content of the field to "hurray!".

Without the employment of the SPAs, a model could only provide an output in the engineering units and a verbal output cannot be provided. On the other side, with a strong study of SPAs, the network can output verbal and meaningful outputs in natural language. The model developed for this dissertation contains four subnetworks, each providing outputs via a SPA so that the outputs provided by the network form sentence-like structures for each observer. The chapter four will provide the details of the networks and the SPAs.

CHAPTER 3

3 CONTENT OF THE MODEL: UNDERSTANDING THE FIGHTER PILOT DECISIONS AND AERIAL COMBAT

A fighter pilot, or the "warfighter" in modern aerial combat literature, is a phenomenon with ultimate popularity in media. In certain aspects, the warfighter is demonstrated as a very strong athlete and a hero, and in some cases, as a very sophisticated and well-educated chess player. But is aerial combat a game like chess where the players see the opponents' all tools and the game board clearly? The sky is vast and broad compared to the detection ranges on board the aircraft and the opponent is just a particle in infinite space. What a pilot observes in the sky is very limited in comparison with the observations of the chess player. A similar question arises on the cliché of the "athlete-hero pilot." Is the physical strength to resist eight or nine G's[5] and wild sustaining turns enough to outcome another pilot with similar abilities? Can physical suitability (clearly a mandatory property for a fighter pilot) be the cue of aerial combat success? Let us further improve our questions. Imagine two pilots of the same air force with similar history (such as graduates of the same year, similar education, flying in the same airbase) and exactly the same model of aircraft, employing similar weapons in an airborne training exercise, coming together as opponents. One may be in the blue force trainees and the other as an instructor in the red aggressors. Under very similar backgrounds and the existence of very similar tools, these two competent pilots generally create a winner and a loser in each combat. Men with the same education, aircraft, and muscular power, but one defeats the other each time. What is the difference? What makes a pilot able to defeat the enemy and survive? The answer may be the "decisions" of the pilots under a very low observable environment. The word "observable" is used as the path planning or motion planning robotic studies. The pilot monitors only a portion of the game field, only some motions of the opponents and only have very few information of the enemy ammunition, aircraft potential etc.

[5] A one G is equal to the gravity acceleration of the planet earth. Eight G means eigth times greater acceleration of earth.

In a study to model pilot decisions or to build up an agent to give pilot-like decisions, even in cases where the agent is not bioinspired, understanding the pilot decision behaviour is mandatory. Therefore, in this chapter, the basics and dynamics of pilot combat decisions will be summarized to provide background information about the functionalities of the model. Understandably, nations' aerial combat guides are classified and not publicly distributed in open sources under normal conditions. Yet, there are some open-source training documents, memories of pilots, simulator handbooks, and similar resources still available. Shaw (1985) is one of the most important resources available open-source. The NATC (2018) training guide is another helpful resource. Gunston (1988) can be a secondary resource since it provides a historical overview of the development of aerial tactics. Yet the following concepts provided in this chapter about the aerial warfight and engagement are mostly a summary of the interviews with test pilots of Turkish Aerospace Industries.

The classical aerial combat pilot has three types of weapons. Potential future weapons like lasers or radio frequency weapons are left outside the scope of this study. These classical weapons are divided into three groups regarding their ranges. The first group with the longest range potential is named the Beyond Visual Range (BVR) weapons, mostly formed of long-range radar-guided semi-active missiles like AIM-120 AMRAAM (Advanced Medium Range Air to Air Missile). The second group has a relatively shorter range and is named Within Visual Range (WVR) weapons, including Infra-Red (IR) guided missiles like AIM-9 Sidewinder. In this context, the visual range does not need to be the pilot's visual range but the missile seeker's range where the missiles are passive (i.e., does not emit any power to the target but reacts to the signature of the target). Finally, the shortest of all three ranges is the gunnery range (GR) which is truly a pilot eye visual range requiring line of sight between the pilot and the bandit. The word bandit is very often used for naming the opponent in air combat literature.

The BVR range may reach distances up to 100 km regarding the model and technology of the missile employed, while the maximum WVR range may decrease to 15 km under normal conditions. Finally, GR may be short as some thousand feet. Of course, at closer distances, the chance to omit a dangerous situation or flee may get harder; therefore, pilots would prefer a valid kill with BVR employment if possible. However, the real life engagements may not allow such a success each time, and the BVR employments may miss the target, and the inventory may end with only WVR missiles existing, or an opponent undetected in BVR range may show up suddenly in WVR range. Identifying the bandit in the BVR range where conditions force the hunter to close up to the WVR range requires a BVR-to-WVR range transition flight which is a truly harsh and difficult flight type. The method will be summarized in later paragraphs.

The GR employment is the most undesired situation where the weapon is not guided and the success of the employment completely depends on both aircrafts' respective abilities. Figure 7 provides a relative demonstration of employment range types. Therefore, a severe combat decision of the employment depends on the range of engagement and the content of the weapon inventory. When the ownship aircraft velocity increases and the engagement is head on, both BVR and WVR ranges may be reached or get close to their maximum. As seen in Figure 4's scenario 1, the maximum closure speed conditions for a head on game, BVR ranges may be more than 20 NM longer than WVR ranges. Therefore, in such conditions, BVR-To-WVR transition flight may endure 20 NM, and with Doppler Notch, this transition may be quite long, difficult, and dangerous.

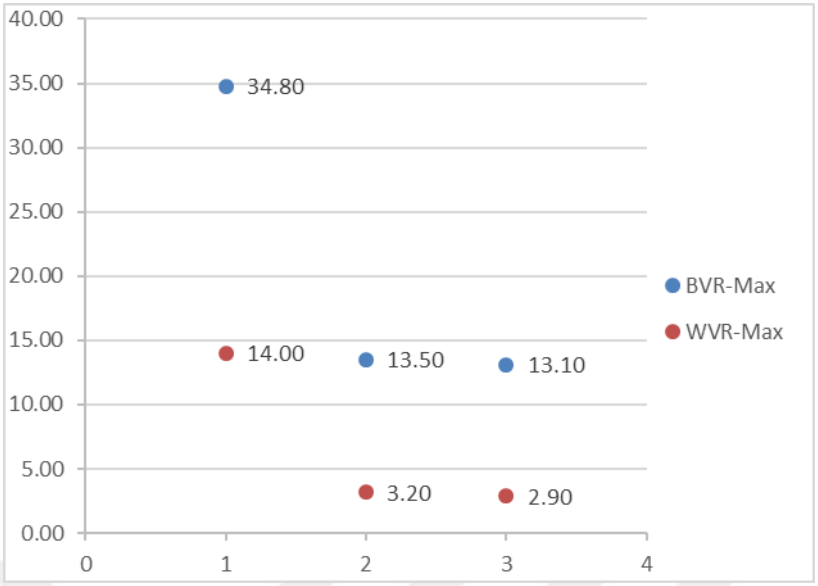


Figure 7. BVR to WVR ranges comparison for three different scenarios.

By the very nature of the rules of engagement, the first decision of the aerial contact is not the employment decision. Before making an employment decision, the hunter needs to be sure of the detected target's identification status, so the very first decision is the identification evaluation. The target may appear on the radar screen of the hunter or just be transmitted from a ground-based post or Airborne Early Warning and Caution asset such as AWACS aircraft via radio datalinks, can be previously identified as hostile, can be a neutral aircraft such as a known airliner or can be unknown or unidentified. In the case of neutral or unidentified target signature, rules of engagement may vary depending on the war cases. Under normal conditions, neutrals are let go, and unidentified aircraft are chased for proper identification. Normal identification tools of a pilot are, first, his/her radar's identification database and capability; second, the identification interrogation system such as IFF (identification of friend or foe, a piece of avionic equipment) and lastly, his/her own eyes. If the target's signature does not exist in the radar database, and the signature does not respond to the identification interrogation system, the pilot may be forced by the engagement rules to close up on the target to see the tail codes or painting to get visual identification (VID). This type of VID may require complex approaches such as stern manoeuvres or intercepts. If the target subject to VID is a hostile aircraft, there is a high probability that it reacts to the approaching hunter's VID efforts before the stern closure ending with a BVR combat. On the other side, if the VID is succeeded but the target is still hostile, a BVR or GR combat may also get started by the hunter. Finally, if the target is found neutral by the hunter, it may be released to continue its route. Under a high-tension atmosphere of a border conflict, identification systems are truly important since VID approaches are very dangerous and undesired. In the last decade, situations like the Tripoli operations of western forces showed very common VID necessities since each nations identification systems were not able to respond to interrogations of the allies flying in the same operation.

After acquiring the identification and the check of the inventory, the pilot has the clues of proper engagement, and is able to select the weapon of the situation. BVR weapons' employment is relatively simpler. The pilot may prefer to lift up or not, but simply checks the range caret in its display (mostly head up one is preferred), uses the engagement

symbology mostly focusing on the employment, not on the manoeuvres. Range caret can be observed on the right-hand side of Figure 11 below. The given symbology (the content of display for specific purposes) is typical and well-known for western aircraft. If the hunter has an angular advantage where the target's head is not directed at the hunter, the BVR missile employment is relatively much easier since the hunter may use the advantage of the enemy's radar signature to lock on from longer distances. In Figure 8 below, a classical figure of radar signature is shown to define why an advantage appears in approaching a target from oblique angles. The RCS in most fighter aircraft increases in oblique angles (30 degrees in the above figure) due to wing sweep and aspect angles. Therefore, approaching such aircraft from this specific angle may increase the detection ability. Additionally, since the hunter approaches from an oblique angle, the bandit's detection possibility will decrease. Figure 9 defines the concepts of angles in an engagement and simplifies the understanding of the relationship between approach angles and RCS. It shall be noted that the RCS of the polar plot given in Figure 8 is asymmetrical, potentially due to the asymmetrical weapons loading of the subject aircraft.

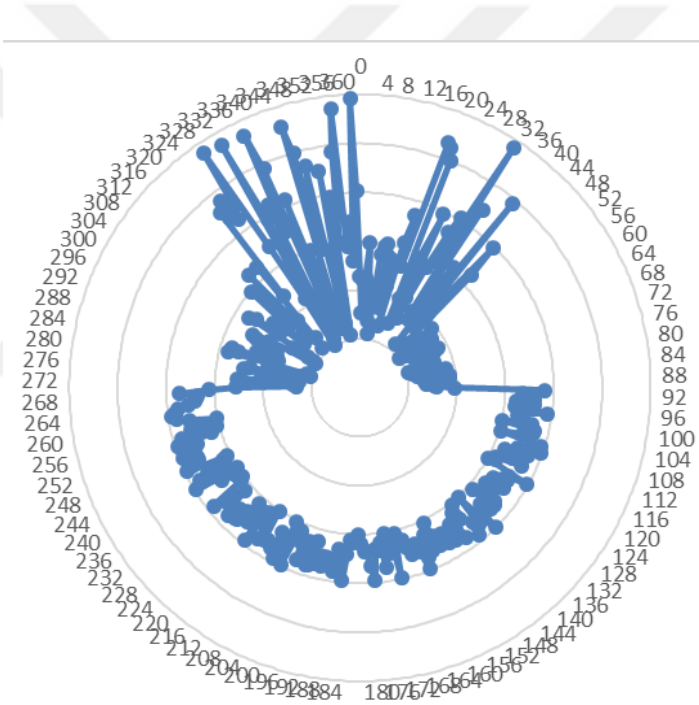


Figure 8. Representative Radar Cross Section in a specific frequency.

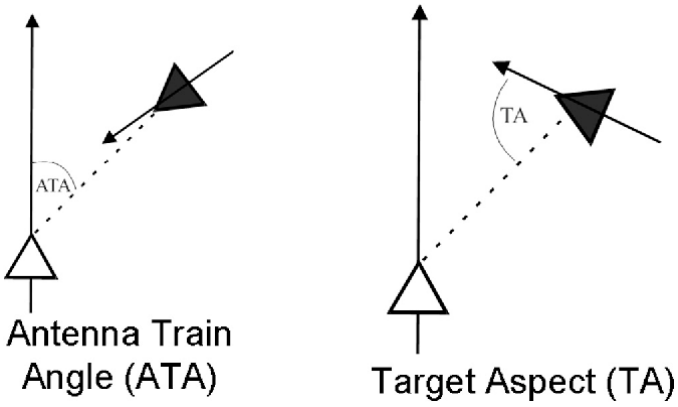


Figure 9. Antenna Training Angle and Aspect Angle definition for engagements.

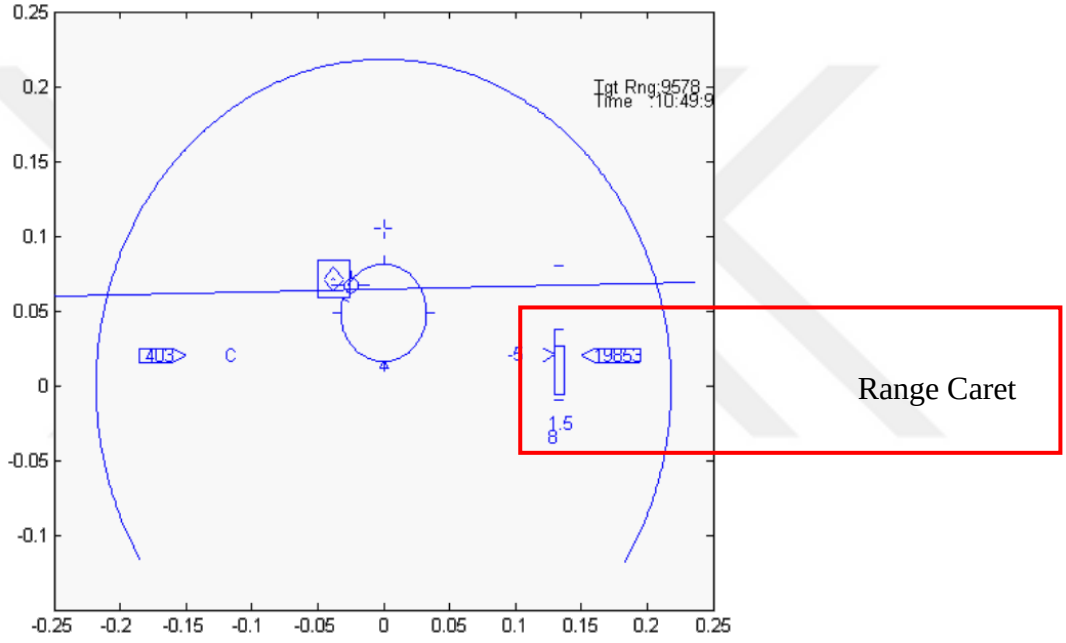


Figure 10. WVR Head Up Symbology for western aircraft provided with permissions of Turkish Aerospace Industries.

BVR engagement may seem relatively simpler since the missiles' ranges are quite long, and there is no line of sight contact with the opponent. On the other hand, the hunter may get Radar Warning Receiver alerts (RWR) showing that the opponent also had a radar lock on the hunter, and it is a difficult situation since it may not mean that the opponent has fired a missile. So this is a partially observed condition where there are some cues such as RWR tone or display and Radar signatures of the opponent, but still, there is no real information about the true actions or intentions of the observed signature.

After employing a BVR missile, since the very modern weapons are generally semi-active radar-guided missiles, the pilot should guide the missile with its Radar until the missile seeker is able to track the opponent itself without the datalink support from aircraft radar. Therefore, for a limited duration, the hunter shall not break the radar lock on the target. Generally, this duration for a classical BVR is less than 80 seconds at most. It should be

noted here that the bandit has most often also a radar, and during the guidance phase, it is possible that a missile may be flying towards the hunter. Regarding this fact, the hunter needs to protect a certain distance relative to closure speed between two aircraft, which is namely the Minimum Abort Region (MAR). The missile guidance shall be over before the minimum abort region so that the hunter may have a chance to break up before entering the missile employment ranges of the bandit. Also, the pilot needs to memorize that his/her aircraft will need a distance for a proper slice back before the MAR, so a separation distance shall be added to the MAR value to find a Decision Range (DR). Therefore, the pilot employs a BVR missile and guides it until it reaches to datalink off range and never enters MAR computing the DR. Clearly, a BVR missile is not a fire-and-forget weapon. Finally, when reaching the DR or undesired MAR, or at the point where the datalink is off, the hunter breaks up, preferably with the slice back manoeuvre, where the lift vector is used to increase the velocity outside the enemy alignment.

Given the generic engagement scenario summarized above, it may be deduced that BVR engagement is mostly a computational task more than an athletic one. In this type of engagement, the pilot uses rules of thumbs and the content (symbology) of his/her displays to employ the missile properly and to keep out of the fatal danger zone. But when the inventory is out of BVR, or the bandit pops up surprisingly in shorter distances or unexpected angles out of the radar scan zone, the engagement may lead towards a WVR.

WVR engagement requires the missile seeker to have a line of sight and proper acquisition of IR signature radiated from the opponent above a detection threshold. So the hunter needs to be inside a detection range and missile flight range which the missile may travel in less than 40 seconds. Modern missiles can be launched in high off-boresight angles without seeker acquisition, but those are left outside of this study since the aim is to understand main pilot decisions. Now, if the hunter needs to fly to the inside WVR range from a BVR range starting point, it is a serious problem. The bandit may still have BVR missiles in his/her inventory and may fire some of them when the hunter is inside the MAR, which may sometimes be larger than the WVR range. Therefore, the pilot needs to omit the bandit's chance of BVR employment with some tricks.

The most common trick to counter a BVR threat is the Doppler notch approach. The Fire Control Radar needs to segregate the signatures of a flying object from the earth's surface which is always behind for low-altitude signatures. A Radar mostly uses the Doppler principle to be able to make this discrimination. Since the earth is truly still in its location in the aircraft reference frame (cancelling the aircraft's own speed at the algorithm level), a moving object can be simply separated from ground clutter via the Doppler principle. However, this discrimination depends completely on the accuracy of the radar's speed detection. Mostly a speed filter is used to eliminate the ground clutter from a flying object, and this filter uses a speed threshold due to measurement accuracy issues. This number may be smaller than 50 knots (Nautical Miles per Hour) down to 10 knots, depending on the technology of the Radar. It can be deduced that if the hunter succeeds in flying slower than the bandit radar's Doppler pulse speed detection limit, then it is lost inside the ground clutter, and a BVR cannot be employed toward him.

Unfortunately, a jet fighter aircraft cannot fly at such lower speeds, but the Doppler notch method is a well-known manoeuvre to break up the radar locks. Doppler notch means a flight at a lower altitude than the opponent, in a circular trajectory where the enemy is put at the centre, and the closure speed to this centre is lower than the mentioned speed threshold. So the hunter may circle around the bandit, almost always perpendicular to the line of sight of the bandit, and may only slowly close up to the WVR range by decreasing the circle's radius with very low closure. Not imagine the hunter need to close up from 50

km range to 20 km using a Doppler notch with a closure speed of 40 km/h (some acceptable Doppler notch kts.). So the hunter may reach the WVR range in a truly long duration for aerial combat, and during this time, many things can occur. First, the bandit's position is never properly known when the hunter's flight path is perpendicular to the bandit's line of sight. So the enemy location is not well known due to the lack of radar contact, and the only clue of the bandit's position is its RWR position. Second, the enemy may not fly in a linear route and may jingle or perform various chase manoeuvres when he/she notices that the hunter is performing a notch. Lastly, in the BVR-to-WVR transition, it is very easy to lose the bandit if the radar lock is broken consciously by the bandit, and the hunter may get trapped. Doppler notch requires well-trained coordination between the flight controls and the RWR or visual observation via the canopy and a very good understanding of the 3D aerial combat geometry. Therefore, it is not an easy manoeuvre or a good preference for a beginner.

Assume that our expert hunter reached the WVR ranges. Now he/she should seek for a proper seeker contact with the enemy, and this is named as the short range combat or in some cases as the dogfight. In Figure 4, the WVR symbols in the Head Up Display (HUD) shows the symbology for a typical WVR missile. The main goal is to get the bandit into the seeker Field of Regard (FOR) and get the missile's reticle caging upon the bandit automatically by radar guidance or manually. When a true seeker lock-on occurs, the missile tone (an aural warning) is heard by the pilot, and now the missile can be launched since it is locked. At the beginning, it may seem easy, but the bandit will make every effort to keep out of the hunter FOR and try to lock up his/her own missiles to the hunter. Therefore, it will be a matter of life and death in the short range manoeuvres.

At the initiation of a short-range aerial combat or WVR engagement, the initial position of the hunter may be either offensive (i.e., having the advantage to attack) or defensive. The situation may affect the decisions. Major decisions can be summarized as follows.

In a defensive state, the main goal is not to give the opponent a proper shot. Therefore, the hunter shall always keep itself moving out of the plane for not giving the bandit a chance to be in the shooting plane. One major method is beaming. i.e., continuously pulling the lift vector towards the bandit so that bandit always loses the line of sight with the hunter. During this process, the hunter's lift vector continuously be towards the bandit so that the bandit never has a chance to be in a lead position. If the bandit changes his/her plane of motion to get out of the beaming of the hunter, then the hunter may have a chance to get the lead pursuit and switch to offensive state. If bandit does not plan its position in 3D space very well while the beaming hunter is approaching, He/She will lose its lead position and fall at least to lag. In case of lag position is not corrected for a while, the hunter will get the lead and the offensive state.

In an offensive state, the hunter has to make an assessment of the opponent aircraft and abilities. The main manoeuvring decision depends on the opponent's rate of turn information. There are two major types of engagement entries. First is the single circle engagements where the attacker prefers to follow up with the defender in the same circle since its ownship's rate of turn is superior and the attacker can protect its lead position. During this lead position, if a proper shot advantage is obtained, a WVR or GR shot can be evaluated.

In case of the defender's (bandit in this case) rate of turn ability is known to be superior to the hunter or assumed to be this way, then the hunter prefers to attack with a double circle engagement where the attacker prefers to turn opposite to pitching direction where the turning circles of both aircraft may coincide only in a single point in 3D space. Of course

such a perfect geometry may not always occur because the turning circles may not always be truly circular, but the same principles also apply to ellipses.

As it can be seen, the main decisions depend on the information known or assumed about the opponent during an air-to-air engagement. First, the detected "unknown" aircraft's intention is labelled as "neutral," "friendly," or "foe" with identification artefacts or VID if necessary. Then, if a bandit exists in the scene, the range is evaluated with ranging artefacts, and depending on the inventory status, an attack munition and method are decided. While the ranges, the content of the inventory, and positions vary, manoeuvring decisions are made mostly depending on the enemy weapon inventory status, enemy actions, and turn rate. The last three items are mostly not known unless visual contact with the opponent is not acquired. In some cases, Missile Approach Warning Receivers (MAWS), if installed on the hunter aircraft, may help in detecting the approaching missiles, therefore, in detecting enemy intention or action. But in many cases, the rate of turn or inventory cannot be known without visual contact or intelligence data. Therefore, many decisions depend on assumptions about the opponent. A fighter pilot always prepares for the worst case and assumes that the opponent's inventory and the turn rate are superior to his/her conditions. When high-level decisions are made, a medium level of decisions exist for manoeuvers; the beaming for evasive actions, Doppler notching for range transition flights, and circle followings are all medium-level decisions to be followed via control tools of the aircraft. The success of the attack or mission depends first on the accuracy of the high-level decisions, then on the correctness of the medium-level decisions, then, if the above ones are correct, on the application quality of selected manoeuvres in a very obscure environment where information are mostly supported with pessimistic assumptions.

In the above paragraphs, a very rough decision tree of a fighter pilot is summarized. The above information creates a very important question. Does such a functionality require a human-like or bioinspired model that will use similar rules, principles, and assumptions like the way that a human pilot does, or may prefer to construct a much more formal model that will use mathematical or computational tools like Monte Carlo analysis for selecting the best manoeuvre or approach angle to the opponent? Can an AI agent select better manoeuvres in comparison with very basic single or double-circle attitudes selected very rapidly by the human pilot? One method will create a fighter pilot model, and the second one will shape a robotic agent that replaces a human pilot with a different model, maybe with superior functionality, but will not be a human pilot model; it will be another model with piloting ability.

Secondly, the method of implementation for the selected model type may vary a lot depending on the tools used and the modelling method. A symbolic modelling approach utilizing any of the existing cognitive architecture tools may simplify the modelling task a lot. ACT-R will be a very good candidate since the rules summarized above are very suitable for formalisation in the ACT-R environment. A very simple decision tree is given in the below Figure. Additionally, being symbolic, these types of architectures may ease the "labelling" tasks of the pilot model where the connectionist alternatives would require additional efforts to build such structures (e.g., The Semantic Pointer Architecture –SPA used in Eliasmith's Neural Engineering Framework is a very good example of purely connectionist modelling approaches requiring additional tools for semantic content). Also the embedded implementation of such a model built using a cognitive architecture will be quite easy to control a robotic agent. On the other side, such a model will not be fulfilling the embodiment and situatedness requirements for unconscious piloting tasks such as detail motor controls on the lowest level of the piloting functionality. The study here aims for an embodied and situated connectionist model that will seamlessly work on a robot's body.

Current literature mostly focuses on lower-level decisions such as manoeuvring selection decisions or estimating the bandit's next action. For an understanding of air combat manoeuvres (ACM) or basic fighter manoeuvres (BFM) many open source and military documentation can be found. A very popular resource is NATC (2018). To categorize, to select or to predict the correct action between the ACM/BFM sets, multiple methods are applied. Rodin and Amin (1989) provided a very early example of artificial neural network usage in ACM manoeuvre selection. The study also provides very clear representations of some ACM manoeuvres for the interested reader. Ghasemi et al. (2005) employs a fuzzy model for dogfight performance of a pilot. The fuzziness mentioned in the method is applied to the pilot's manoeuvre selection decisions. While being symbolic, the model provides proper dogfight manoeuvres. Karlı, Efe and Sever (2017) provides an algorithm to select BFM manoeuvres in between actual flight data. The study provides an abstraction of the pilot decisions, where the strategic decisions are on the top level, and flight physics is the lowest level. The authors define the strategic decisions as the determination of the high-level objectives for tactical decisions. Karlı, Efe, and Sever's definition of strategic pilot decisions is parallel to this study.

Influence Diagrams are also widely used in modern literature for manoeuvre decisions. Virtanen, Raivio and Hämäläinen (2004) provide a multistage influence diagram for sequential manoeuvre decisions. The stochastic approach provided in the manuscript is resulting in advantageous selection of correct manoeuvre. Machine Learning methods are also increasingly used in the ACM/BFM modelling topics. Zhang et al. (2018) provides a Reinforcement Learning based machine learning method to obtain a manoeuvring selection algorithm. The Zhang et al. method is different than the others, since Zhang et al does not select from a set of ACM manoeuvres. Rather it makes a selection between the members of a limited set of turning directions such as "turn left" or "go straight". At the end, the method provides a model of a fighter aircraft able to dogfight partially. Finally, there are also groups focused not to a total pilot model but to partial functionalities modelling. Zacharias et al. (1996) employs symbolic methods to build a situational awareness model for pilot in the loop tests or trainings. Situational awareness is the heart of the pilot's decisions, therefore their attempt may be seen as a facilitator and a great achievement for a complete cognitive pilot model. Despite being not implemented on any cognitive platform, still the Zacharias et al. model provides not only a situational awareness model but also a model for situational assessment, i.e. the model is able to select or predict and action (1996). Therefore, it should be noted that the mentioned model is a decision model.

Present fighter pilot is trained regarding world famous Boyd's OODA loop (Boyd, 1995). While the idea is developed by a United States Air force pilot, Colonel John R. Boyd in late 1950s, it is widely used worldwide, especially by the western air forces and cyber security studies. An OODA loop consists of four blocks sequentially following each other and restarts, namely Object-Orient-Decide-Act. On the other side, the discrimination or distinction between the subchapters of the decision process may not always be that clean and neat. While most literature focuses on the determination of the tactical decisions or manoeuvre selection, high-level decision such as "attack", "flee" or "let go" is mostly left out of scope. Probably, the importance and difficulties of such decisions are quite underestimated. Therefore, most of the models deciding on the manoeuvres, in fact need human supervision to start the action. On the other side, higher level decisions are in fact truly complex and very open to affective manipulation. In this study, a high level decision model for a warfighter pilot is developed and compared by pilot's original decisions. Please see below Figure 11 for a representation of the content of the model developed for this study.

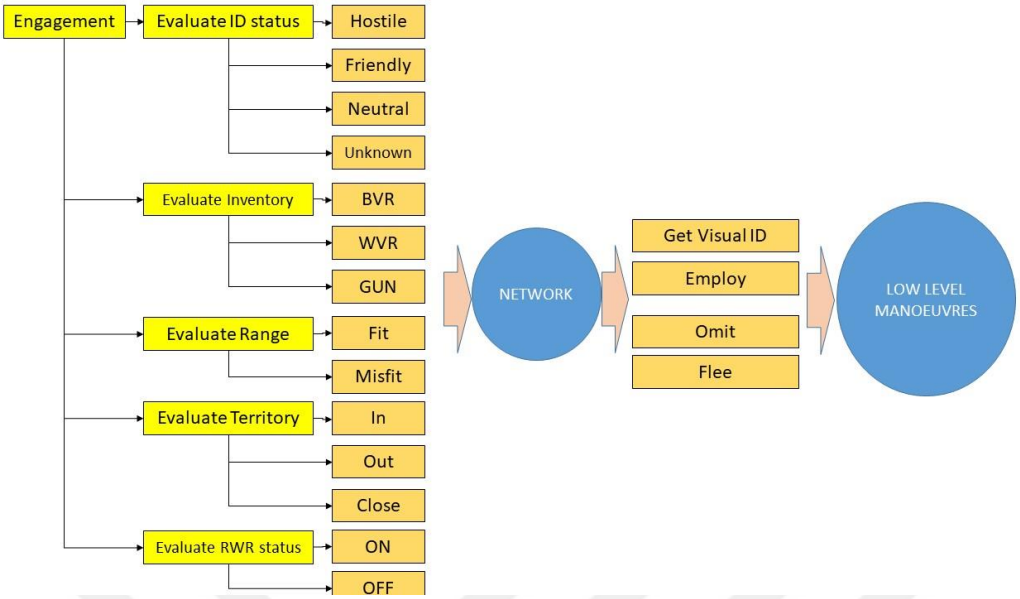


Figure 11. A brief summary of major engagement flow and decisions.

As described in the previous paragraphs, pilot's engagement decisions are hierarchical, prior to the manoeuvring decisions, higher level evaluations and decisions need to be accomplished. Ranges, territory intruding status, RWR and inventory status need all to be evaluated, target's validity need to be assigned, and a final higher-level decision need to be provided. Figure 8 represents a multiple-stage and hierarchical network abstraction for such a decision. The network architecture, represented in Figure 11, is implemented in NENGO using Neural Engineering Framework as the main task of this dissertation. The following chapters will be providing the details of the NEF and the model, and evaluate the model and methods from various aspects such as noise resilience.

CHAPTER 4

4 CONTENT OF THE MODEL: METHOD AND STRUCTURE

In this modelling study, it is aimed to explore dynamic factors that shape the high-level decisions of a fighter pilot by developing a cognitive model and simulating its behaviour under realistic aerial combat scenarios. For this purpose, a cognitive pilot model that simulates high-level pilot decisions was developed by using NENGO with spiking neuron models. Given the dynamically changing, uncertain nature of aerial combat, we followed a bioinspired, neurodynamical approach to model a pilot agent that may exhibit the required level of cognitive adaptability and flexibility involved in this decision-making context. The following sections describe the basic components of the pilot model and outline their working principles and assumptions. This is followed by a description of the aerial combat scenarios used to evaluate the pilot model. The evaluation was based on the views of highly experienced fighter pilots. The study concludes with a discussion of the findings and their possible implications for the cognitive fighter pilot model development efforts.

During an aerial mission, the "pilot in control" evaluates current inputs or knowledge in a partially observable environment, decides on an action under high uncertainty conditions, and performs the selected action. When inputs to the situation or the overall system state change during the action, the pilot's decision may also need to be re-evaluated and changed. Therefore, a pilot model may include higher functionalities in comparison with an autopilot in which the intended attitude values are manually selected by a real human agent. For a summary of older control theory-based models, interested readers may refer to Kaygusuz and Çakır (2015). In Figure 12, a basic representation of decision hierarchy is given.

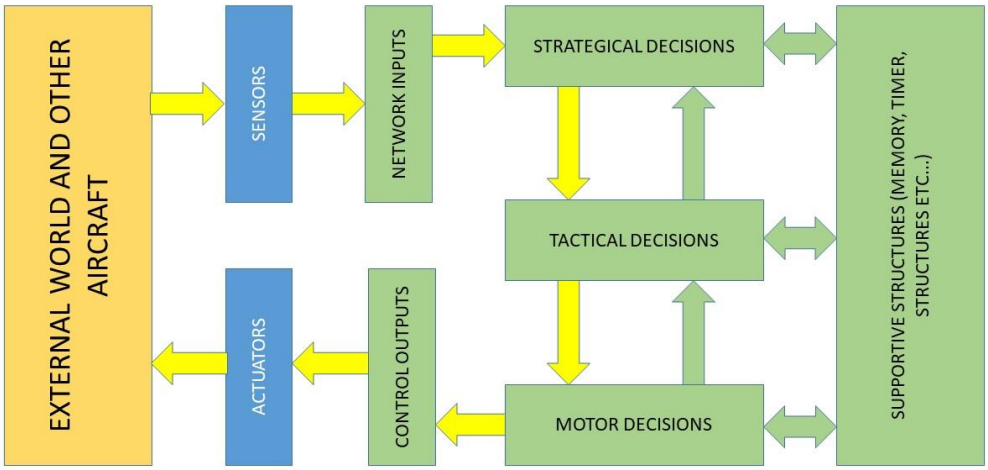


Figure 12. A Complete Aerial Mission Pilot Model Architecture.

As it can be seen in Figure 12, a pilot model shall first be able to decide on its next high-level action, i.e. shall be able to decide whether to attack, flee, chase, or just ignore in the face of new data. Therefore, the highest level of all cognitive faculties shall be the decision layer. Since there will be various other low-level decisions in the action planning, this level is named the strategic decisions layer. It shall be noted that, the Figure 12 is very parallel to the Hutchins' Information Flow Model given in Figure 2. Also neglecting the buffers of ACT-R architecture in Figure 3, similarities are also observable between Figure 12 and Figure 3. It can be deduced that, whatever the modelling approach is, current understanding of human cognition is almost as given as in mentioned figures.

Stewart and Eliasmith (2011) built one of the first large-scale spiking neuron-based cognitive models by using 150.000 Leaky Integrate and Fire (LIF) neuron models emulating the tuning properties of cells from specific brain regions. The modules of the model are designed with biological inspiration and the represented cortical areas are thought along 18 expert rules for this specific task. The model includes parts namely the cortex, thalamus, and basal ganglia, corresponding to the original function of a human agent. Therefore, there exist mature examples of spiking neuron based embodied models to perform complex cognitive tasks. In this study, a very similar approach is employed for a very specific task. For the model, a set of possible high-level decisions for a pilot to evaluate during an engagement in aerial combat are developed using NENGO.

During the preparation period, 7 different test pilots of Turkish Aerospace Industries contributed to the dataset development work. First the pilots provided a detail description of the aerial engagement and manoeuvers. In the second part, a dataset is developed in form of scenarios. Total number of scenarios is 250. Pilots background varied from 3000 hours to 5500 hours of aerial fighter aircraft, military aircraft and test flights. 3 of them are certified test pilots from international test pilot schools (ITPS). The pilots described the rules of engagement, conducted tabulating the training dataset. In the Appendix 1 of this study, a transcription with a pilot from one of the scenarios is given.

13 different data types are encoded in the model as inputs, i.e. transformed into values carried by spiking characteristics of neurons as inputs to the system. These inputs can visually be observed in a graphical user interface in NENGO-GUI or can be generated

using lambda calculus functions that oscillate automatically. For random or ordered behaviour sine functions are also used as lambda functions. 16 ensembles or neuron sets are used for computational modeling. The total number of neurons in this ensemble sums up to 2400. Increasing the number of neurons in each ensemble makes the model more resilient to the effects of noise. Despite being left out of scope in this manuscript, complex spiking neuron models' characteristics under noise is another important topic for future research of complex biologically inspired cognitive modeling. The neuron number in each ensemble creates a trade-off between noise resiliency and processing speed. The current number of neurons is 2400 for the overall model, which brings a maximum model speed of ~25% of real-time. Together with the influence of backup processes in the running environment, the speed may further decrease to 13%.

Each scenario is formed of four lower scenarios. The very first of lower scenarios is the inventory. The inventory of the ownship can be containing BVR missiles up to 4, WVR missiles up to 2 and gun ammo. Gun ammo existence can be a Go-noGo, therefore it is evaluated only as 1 or 0. Depending on the content of the BVR inventory the approach of the pilot would strongly vary. Because BVR employment is made from a relatively safer distance and is somehow providing a safety advantage. In case of the BVR lack, WVR would be a relatively positive option for the pilot in comparison with the gun. Since the inventory of the bandit is not known, in case of ownship requires to close on the WVR or gun ranges, opponents' inventory may be a load of higher range weapons leading to a catastrophic result for the ownship. Therefore, in case of visual cue is not existing, a pilot always assumes opponents inventory is in its most loaded state.

The second lower scenario is about the identification. The RADAR trace or visual cue observed may be identified or not. If not, this means an unknown RADAR trace which needs to be tracked and identified. It can be a friendly unit, a hostile unit i.e. a bandit and a commercial aircraft, namely a neutral unit.

As third, the pilots cues of Situational Awareness (SA) needs to be evaluated. i.e. the RWR is a very important tool for the SA. The situations evaluation by the pilot regarding the RWR status. In case of bandit is locked to ownship leading to an RWR alert, it is a strong excitory input for an engagement decision. The second important cue is the range status. The range input from the RADAR, needs to be compliant with the inventory. BVR missiles are employed in BVR ranges, and if the separation between the aircraft is about a BVR range, WVRs will be useless.

Finally, the map and TSD (Tactical Situation Display) are other very useful artefacts of SA. The bandit may be locked to ownship, but also may be still flying in its own nation's borders. The level of alertness is also varying in such case. In most air patrol missions, enemy aircraft further far from the border can be neglected to decrease the focus zone of the mission.

Lastly, emotional or affective state of the pilot is also changing the understanding or assessment of the pilot on the situation. Notice, there is not mature findings on the mechanisms of such affective inputs. Therefore, in this study, only affective state is applied to each subnet as an excitory input easing the threshold passing and creating increased spike rates.

As a further study, Datasets will be published separately after this dissertation after developing a guideline for the reader on the usage and understanding. Such a scenario set can be helpful in the development of the aerial robotics and CGF applications.

In addition to these 16 ensembles mentioned in the first paragraph, four Semantic Pointers (SP) are used to decode the decisions into verbal communications. These four SPs are also leading to a heavy load on the processor. The verbal outputs of the SPs can be seen in the

lower part of Figure 3. A pilot needs to evaluate the aircraft weapons inventory, the position and identification status of the bandit or the target aircraft, and finally the range. Additionally, the emotional state of the evaluator will also affect the degree of aggression or the passivity of the decisions.

The model is formed of four subnetworks, each of which is formed by the merging of an inhibitory and an excitatory network. In each subnet, various states compete to rule the output of the subnet and provide an increasing contribution to the output.

The effectivity or functionality of each subnetwork is obtained using weights in the connections representing the synaptic properties of cortical units. A weight matrix formed of 46 different weight values is used to build up a "meaningful functionality" in the overall model. The weights need to be coded individually into the model, which is due to a lack of training methods for spiking neural networks. In a model working on discrete time, a probing method and a systematic weight update method can be used to train complex networks. Such an effort will possibly require high processing power and time, but could provide a solution to the problem of large network training in NEF. Figure 5 shows a part of the weight matrix used in this study.

Each neural evaluation is modelled partially as an interactive activation and competition (IAC) model. Similar to IAC, each input is modelled as a fuzzy variable competing with alternative options. In other words, the potential identifications such as "hostile", "neutral" and "friend" are fuzzy inputs competing with each other for the decisions. The fuzziness of the input is encoded in the strength of the input, i.e. as the hostility option increases in the input observation, its numerical encoding value to the system is also increased. When manual manipulation of an input is required, a slider representing the input is used due to the capabilities of NENGO. The tool also provides the owner the capability to generate random or periodical inputs. Some oscillating fuzzy inputs are also usable when required. Different than the original IAC models, here the ideas do not support inhibitory inputs to competing rivals. This is an aspect of the model open to development.

On the other side, each input set is evaluated with one inhibitory and one excitatory network and finally as the sum of both networks. This idea is very common in the dynamical field theory where inhibitory and excitatory kernels are used together.

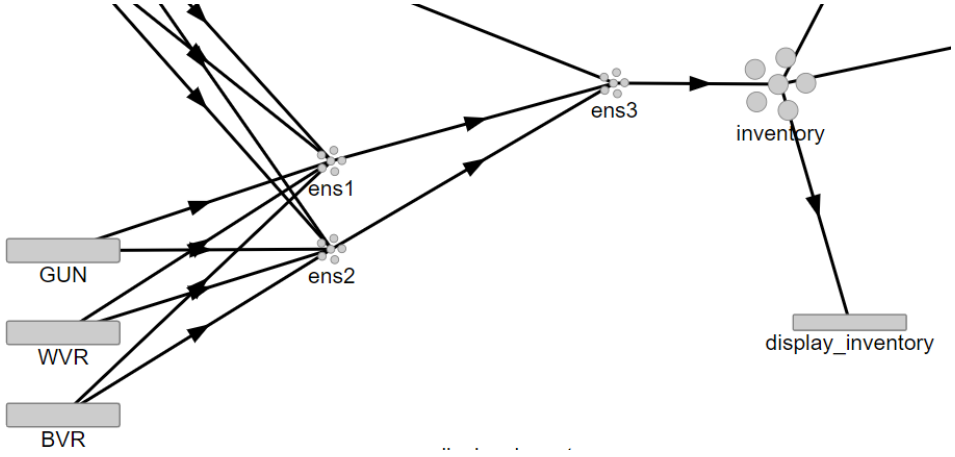


Figure 13. Usage of inhibitory and excitatory networks together.

Finally, Figure 13 shows the SP belonging to this network with the ensembles. Notice multiple SPs can be used side by side to represent the arousal of an idea or decision. With

further study, meaningful linguistic outputs can be created from such networks decoding multiple SPs in a preplanned order.

The model hierarchically contains two levels; the first or the lower level to evaluate lesser decisions and the second or higher level to evaluate the overall decision. The low-level networks are of a total number of 3, each formed of two subnetworks (one inhibitive and one excitatory), supplying spikes to the decision network. The output of all networks is decoded with a Semantic Pointer. In Figure 14, an overall scheme of the model is given.

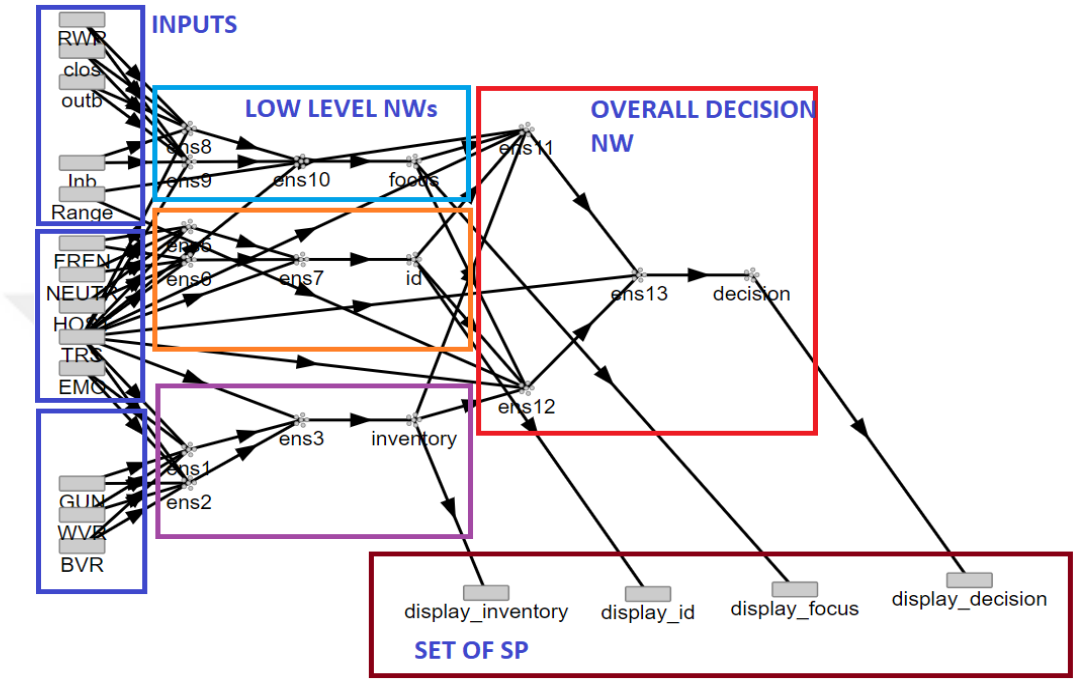


Figure 14. Overall model structure.

The first or low level is formed by the evaluation of the following aspects of the engagement:

The inventory evaluation: inventory of a fighter may include long or medium-range weapons (namely Beyond Visual Range-BVR missiles such as AIM120B), short-range weapons (namely Within Visual Range-WVR missiles such as AIM9L), and gun systems (such M61 Vulcan). Regarding the status of the inventory, the pilot decision will change. WVR manoeuvres and ranges are different and shorter than the BVR and the same principle applies to WVR-gun competition and BVR-gun as well. Additionally, the affective situation or the emotional state of the pilot is very effective in such decisions. In many cases when ammo is about to be over or only one missile is loaded into the store, choosing not to engage may be the correct decision, while aggressive affective states may lead to attacking. The inputs and the output to this network are:

- BVR, inventory status of BVR missiles,
- WVR, inventory status of BVR missiles,
- Gun, inventory status of Gun ammunition,
- TRS, Affective aggressively excitatory control,
- INV, the spiking output of the network,

- Inventory Decision, the SP of the INV value,

The graphical organization of the inventory evaluation network is presented in Figure 15.

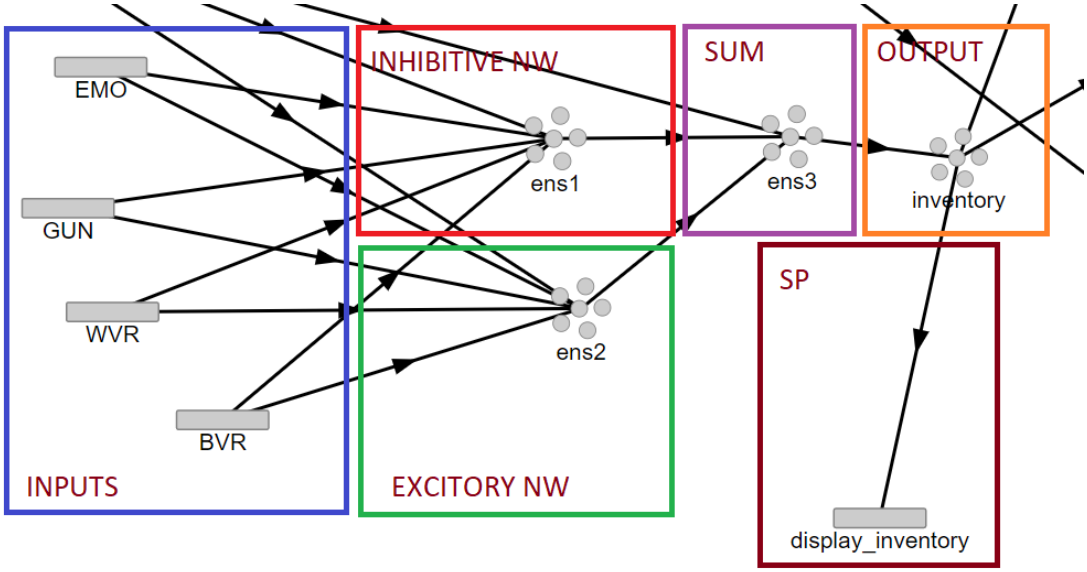


Figure 15. Inventory evaluation network.

The identification evaluation as shown in figure 16: a fighter pilot uses the identification system to analyse the object of interest for validity as a target. The object of interest in the RADAR or tactical displays may be a neutral, hostile, or friendly unit. This issue strongly affects the engagement decision but is not the sole effector. The network uses the following inputs and generates the output:

- Three identification system outputs (HOST for Hostile, FREN for Friendly, and NEUTR for Neutral) are competing in this network. There are two subnets, one being inhibitory and one being excitatory are competing. The outputs of both subnets are summed up and amplified in the id ensemble. Figure 8 depicts the structure of the NW.
- Id: The decision output of the NW about the identification for the object of interest.
- Display Id: The SP of the NW output.

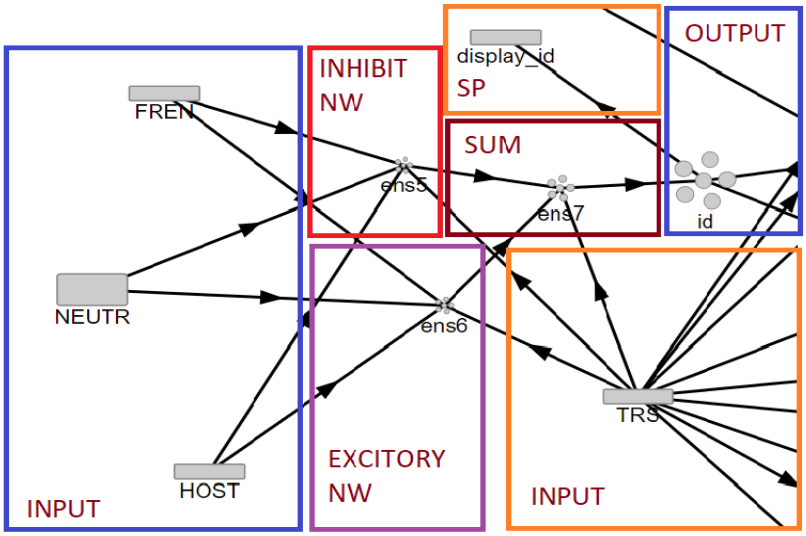


Figure 16. The Identification evaluation network.

The range evaluation network is the most complex subnet in the model with six inputs. The inputs and output of this network are:

- RWR: Radar warning receiver status. The RWR apparatus is a very important tool supplying data to the pilot in the partially observable environment of an aerial gunfight. For example, an object recognized by the RADAR out of the nation's borders is normally not a valid target. But if such an invalid target close to the borders locks down its fire control RADAR to our aircraft, this information may be the clue of a missile firing or preparation for it. Therefore, the validity status of the target may change. If the validity status did not change, such a RADAR lock may require the target to be chased for hostile activity.
- Position Evaluation Inputs: The object of interest may be inside our borders meaning that it is an intruder or a bandit if it is hostile, may be out of our border and interest; and may be close to our borders leading to being of interest if a lock on occurs. So competing inputs are INB, OUTB, and CLO.
- Finally, similar to the previous two subnets, this NW also uses an affective state input.
- In Figure 17, the structure of this network including its two subnets is depicted.
- The Range in fighter vocabulary means the slant distance from the ownship RADAR to the object of interest as a slant distance output of the acquisitions sensors (mostly the RADAR). But even if we name this NW as the Range Evaluation, the range value is not used directly in this NW, but is used in the higher decision with the output of the inventory evaluation network, because the range criteria changes due to the inventory status.

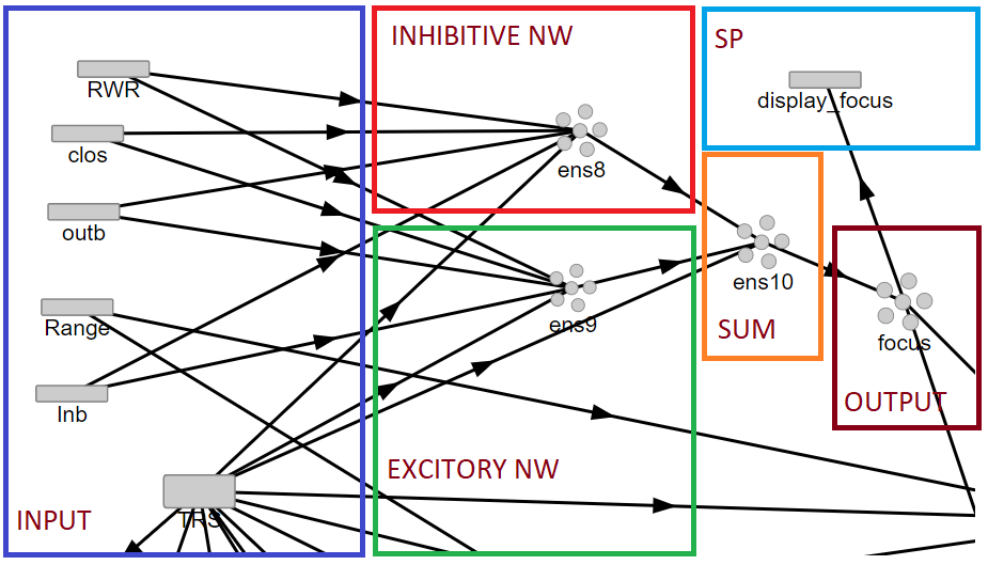


Figure 17. Range Evaluation Network.

The outputs of the subnets are evaluated in the high-level decision network, which is again formed with an inhibitory and an excitatory subnet, using all three outputs of the lower NWs and the Range value, and output one spike encoded signal and one SP. The set of four SPs forms a semantic output for further models to be connected. Please see Figure 18 for the SP set. In this snapshot of the model running, the object of interest is not a valid target probably out of the border, but needs to be chased since it locked its RADAR to our ownship.



Figure 18. Set of 4 SPs from NENGO-GUI..

The effectivity or functionality of each subnetwork is obtained using weights in the connections representing the synaptic properties of cortical units. A weight matrix formed of 46 different weight values is used to build up a “meaningful functionality” in the overall model. Weights need to be coded individually into the model. NEF normally focuses on the training of the neuron tuning functions and larger neural network training capability is limited. On the other side, there are studies employing deep learning toolkits such as Tensor Flow and Keras for deep learning integration into the NEF. In a model working on discrete time, a probing method and a systematic weight update method can be used to

train complex networks. In that regard, NENGO provides open-source capabilities for utilizing PES (Prescribed Error Sensitivity), RLS (Recursive least-squares), BCM (Bienenstock-Cooper-Munroe), and Hebbian or Vector Oja rules. In this study, smaller networks are trained using BCM with a learning rate of 1. Larger networks of multiple layers cannot be properly trained with such a method due to the increasing amount of error. But well-trained smaller networks can be unified to build up the overall functionality. Such a study may require high processing power and a long time, but such a method provides a solution to the weakness of large network training issue in NEF. Posterior to the training, obtained weights need to be manually inserted into the larger network in our current approach.

The model's eligibility is highly dependent on its capability of emulating human pilot behaviour on decisions. Therefore, to be able to evaluate the performance, some complex scenarios are developed and run on NENGO. The outputs are recorded and reported to pilots as checklists. Below is a summary of the evaluation steps:

- 12 scenarios are run where all engagements are one-on-one. i.e. there is only a single bandit.
- %50 of the scenarios are peacetime scenarios and the remaining part is passing on tense times.
- In 33% of the scenarios the target of interest (TOI) was out of national borders and in 67%, it was clearly intruding the national borders. In one of the intruding scenarios, the location of the bandit was vague or obscure to increment the difficulty of the scenarios.
- In 58% of the scenarios, the TOI was clearly hostile, in 8% TOI was a friendly unit, in 8% TOI was neutral, and finally, in 25% the TOI's identity status is unknown or unclear.
- The model's structure worked in two stages.
- In the first level, the validity of the target is evaluated by the model.
- In the second level, the decision is made by the model.

In Table 1 a summary of the scenarios is given.

Table 1. Summary of Scenarios.

#	**Input**
1	Target identification is unknown
	Target is out of our borders
	The event occurs in peacetime
	Target is expected to be non-friendly since being in abroad
	No RWR warning
	Range is not evaluated as or non-BVR range
	Inventory contains BVR
2	Target identification is hostile
	Target is inside our borders
	The event occurs in peacetime
	Target identified clearly as hostile

#	**Input**
	No RWR warning
	Range is not evaluated as or non-BVR range
	Inventory contains BVR
3	Target identification is hostile
	Target is just crossing the border
	The event occurs in peacetime
	Target is NOT identified clearly as hostile
	There is RWR warning, target has locked to ownship
	Range is not evaluated as or non-BVR range
	Inventory contains BVR
4	Target identification is hostile
	Target is inside our borders
	The event occurs in tense time
	Target identified clearly as hostile
	There is RWR warning, target has locked to ownship. Danger
	Range is not evaluated as or non-BVR range
	Inventory contains BVR
5	Target identification is hostile
	Target is inside our borders
	The event occurs in tense time
	Target identified clearly as hostile
	There is no RWR warning, target is just passing by
	Range is evaluated as a proper BVR range
	Inventory contains BVR
6	Target identification is hostile
	Target is inside our borders
	The event occurs in tense time
	Target identified clearly as hostile
	There is no RWR warning, target is just passing by
	Range is evaluated as only gun range, a very close distance
	Inventory contains only gun
7	Target identification is known as not a commercial flight
	Target is inside our borders
	Peacetime
	Target identified as hostile but not sure
	There is an RWR warning
	Range is evaluated as BVR range
	Inventory contains BVR
8	Target identification is properly a friendly unit
	Target is outside our borders
	The event occurs in peacetime
	Target identified as hostile but not sure
	There is an RWR warning
	Range is evaluated as BVR range

#	**Input**
	Inventory contains BVR
9	Target identification is properly hostile
	Target is clearly inside our borders
	The event occurs in tense time
	Target identified as hostile with no doubt
	There is an RWR warning
	Range is evaluated as BVR range
	Inventory contains BVR
10	Target identification is unknown
	Target is clearly outside our borders
	The event occurs in peacetime
	Target is not identified properly
	There is no RWR warning, target is not lock on
	Range is evaluated as BVR range
	Inventory contains only gun
11	Target identification is hostile
	Target is clearly inside our borders
	The event occurs in tense time
	Target is identified properly as hostile
	There is an RWR warning
	Range is evaluated as BVR range
	Inventory contains BVR
12	Target identification is neutral
	Target is outside our borders
	The event occurs in tense time
	Target is identified properly a commercial flight
	There is no RWR warning
	Range is evaluated as BVR range
	Inventory contains BVR

In a real aircraft, the pilot reads the information provided by each sensor from a display set. In this study, for the sake of simplicity, low-level visual perception is completely neglected and it is assumed that the pilot directly observes what is provided by the sensors such as RADAR and RWR (Radar Warning Receiver). In Figure 19, a graphical representation of the inputs is given.

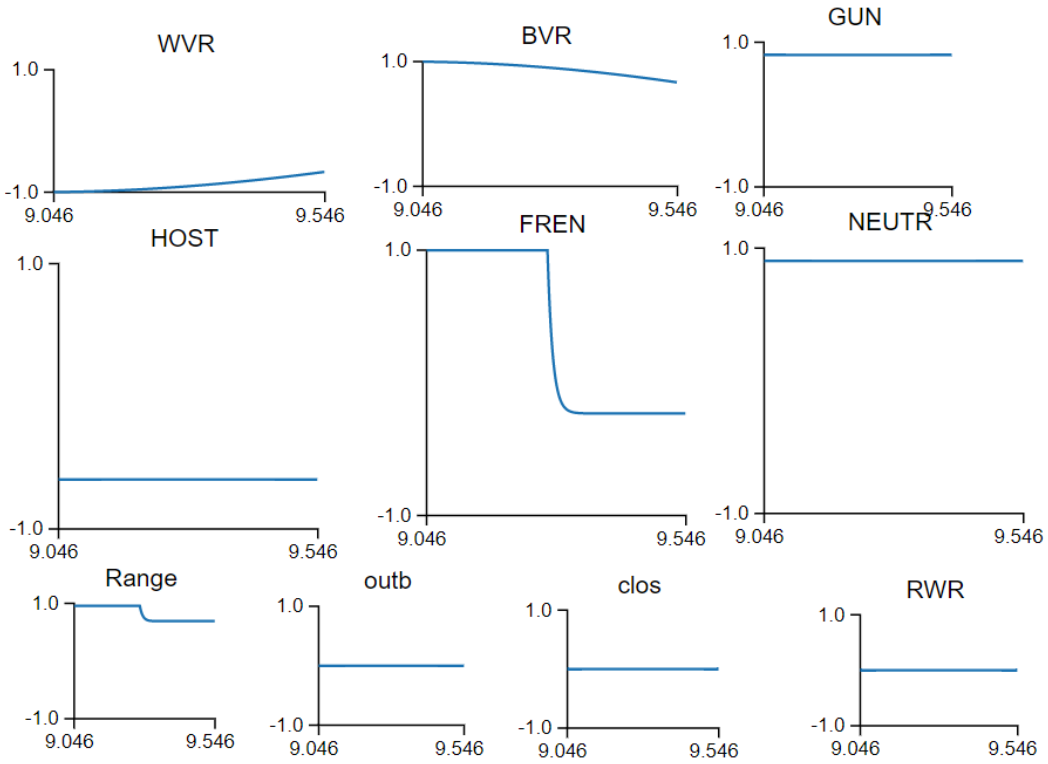


Figure 19. A set of inputs for an example scenario.

The outputs of the model are evaluated by two highly experienced fighter pilots with aerial combat experience, one having 4500 hours of flying experience, and the second with 5500 hours. Both pilots read the scenario reports and provided their own scenario outputs. Pilots also commented on the model's output. For the model evaluation, the decisions of the human pilots and the model are cross-checked.

CHAPTER 5

5 PILOTS' EVALUATION, RESULTS, DISCUSSION AND CONCLUSION

In this chapter results of the study will be provided under 3 main paragraphs. The first paragraph is about the noise resilience of the NEF and SNNs. The second one is about the comparison of SNNs with symbolic graph methods. And final paragraph will summarize the outputs of the model, the evaluation of the pilots on the outputs and the model, the discussion and conclusions.

5.1 Effects Of Noise in The Functionality Of Spiking Neuron Networks

Effects of noise has been an important domain of research during the history of the signal processing. Since signals in the nature are subject to noise, being ensembles of or individually signal processors, neuron models and models built using those, are also subject to different noise types inherently. The robustness and stochastic behaviour of the connectionist models is presently under discussion. A good study specific to spiking neuron networks can be found in Olin-Ammentorp et al. (2021). While Olin-Ammentorp et al.'s study is focused on hardware implementations of spiking neurons, especially in ReRAMs (Resistive Memories), and understanding how biological systems tolerates the synaptic weights variances, still it provides a very good understanding of spiking networks noise effects. On the other hand, since in the NENGO network provided in our study, the synaptic weights are constant, a concept of synaptic noise or synaptic weight variance is not applicable to our network, and so Olin-Ammentrop's findings are not applicable to

ours.[6] The noise application in our network is applied via addition of white noise on the ensembles itself in form of signal variance.

An important point shall be underlined. Even SNN, Neural Field or ACT-R/SOAR, whatever the application or framework is, if the implementation is performed in a soft environment, i.e. on a PC or server employing an operating system and software implemented solvers, in fact; behind the scene is a solver running on a symbolic environment. Therefore, all our spikes or neuron models are in fact just abstractions and so unreal. Such a dilemma can be solved via developing hardware capable of pretending neuron or field behaviour independent from the soft environment provided by the operating systems. Such applications can also cancel or expands the limitations caused by the limited processing power with the strength provided by the dedicated hardware. This type of studies is existing in the literature and may provide the next step of cognitive robotics and artificial intelligence studies (Sandamirskaya (2014), Hunsberger and Eliasmith (2016)).

In NENGO, different types of noise can be applied to single neurons or ensembles of neurons. The variety of the applied distortion can be of the type of filtered noise (specific to a frequency interval, i.e. applied to only a selection set of encoding neurons), white noise (applied to all frequencies, i.e. to all neurons in an ensemble), white signal (specific to a frequency interval with a cut-off frequency, i.e. bandlimited) or can be time limited (specific to only an interval being smaller then overall model working duration) while noise amplitudes can also be arranged (NENGO.ai, 2018). In between all types, only white noise is natural for a healthy network due to the nature of the noise. Therefore, in this study, we will demonstrate the applications of only noise in form on white noise. On the other side, while the proof is left out of scope of this report, it is simply deducible that limited noise types may result in similar results.

While the definition may vary depending on the context, in signal processing domain; the white noise refers to a random signal which is applied with equal intensity in all frequencies resulting in a constant power spectral density. Therefore, for a proper application the average power of white noise is 0 for all frequencies, but in a limited time interval each frequency contains an individual amplitude (Carter and Mancini, 2018). In different disciplines it may refer to slightly different signals, e.g. in music, it is mostly used as the background noise suppressing low amplitude audio signals in the environment. In out context, white noise is random, aperiodic and formed of equal intensity signals in all frequencies. So, when it is applied to a neuron ensemble, each neuron is subject to its individual random noise.

The selected framework of this study, NENGO, allows the user to define “processes” and apply to neurons or ensembles (NENGO.ai, Processes, 2018). In below example, a white noise is defined as a process and applied to an ensemble “b” which contains only a single neuron. On the Figure 9 below, the neuron b spiking voltage is shown with the legend “noisy”. For comparison, a neuron “a” is excited with the same input as “b” and shown as “deterministic” in the legend of Figure 9. The process definition, both the noisy and deterministic ensemble definitions and legend definitions for the plot are highlighted in below code sample. This specific example is given in code, since it contains only individual neurons but not a network.

```
process = NENGO.processes.WhiteNoise(dist=NENGO.dists.Gaussian(0, 0.01), seed=1)

with NENGO.Network() as model:
```

[6] For further information on neuromorphic architectures implementation on hardware, using NENGO, the interested reader may refer to DeWolf, Jaworski and Eliasmith (2020).

```
    ens_args = dict(encoders=[[1]], intercepts=[0.01], max_rates=[100])
    a = NENGO.Ensemble(1, 1, **ens_args)
    b = NENGO.Ensemble(1, 1, noise=process, **ens_args)
    a_voltage = NENGO.Probe(a.neurons, "voltage")
    b_voltage = NENGO.Probe(b.neurons, "voltage")

with NENGO.Simulator(model) as sim:
    sim.run(0.15)

plt.figure()
plt.plot(sim.trange(), sim.data[a_voltage], label="deterministic")
plt.plot(sim.trange(), sim.data[b_voltage], label="noisy")
plt.xlabel("time [s]")
plt.ylabel("voltage")
plt.legend(loc=4)
```

In Figure 20, it is observable that the neuron output voltage without noise (represented with a blue line) approaches its firing threshold being very close but unable to reach to spike threshold. The addition of the white noise causes the neuron voltage (shown with a green line) to raise above the threshold, resulting in two spikes, while after each one the voltage sharply drops to zero (NENGO.ai, Processes, 2018). Similar sample codes and tutorials can be found in NENGO development community environment (github.com/NENGO, 2022). The Figure 20 is a representation of that an increase in signal pitch with noise increasing the probability of an unexpected spike.

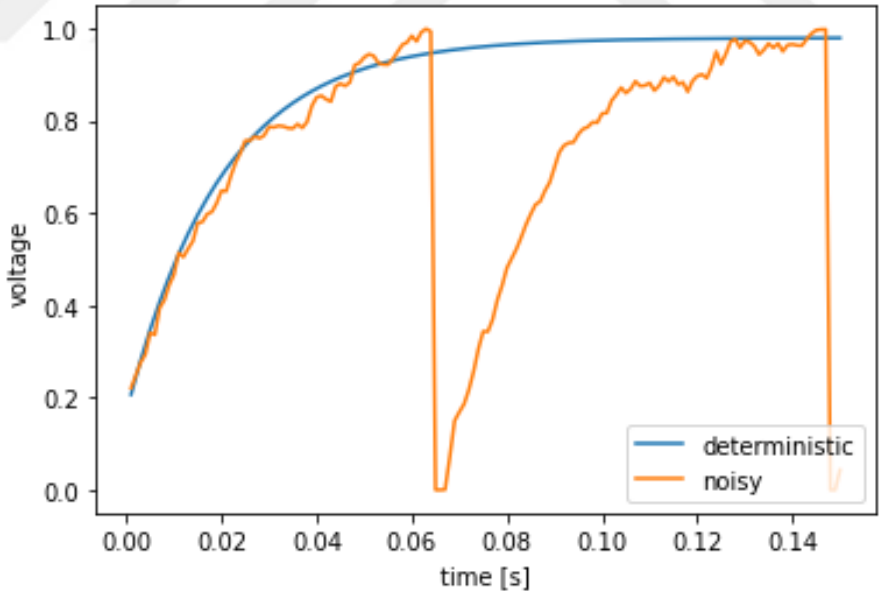


Figure 20. The output voltage difference of two spiking neurons where one is subjected to white noise (Carter and Mancini, 2018).

Above example shows that, noise can create differences in the output of single neurons in undesired or preplanned manners. On the other side, examples of use of individual neurons exists in cognitive behaviour modelling, most models consist of larger assemblies of

neurons, since it is the network which performs the task, not the neuron itself, the effect of single neurons may be negligible is various cases.

One or more ensembles of neurons in NENGO may represent some cortical functionality with different coding methods. In brief, an ensemble can represent an input, a function or an output. Such functionality be a motor, a sensory processing, a decisive one or a relatively smaller part of sub-ensemble of such functionality. In any case, regarding the required accuracy level, the functionalities can be grouped in two main classes. The first one which does not require a numerical accuracy can be very robust towards the noise. For example, determination of the azimuth of an audio source using the input from to ears is a very good example of such functionality. The output of such functionality is acceptable in tolerance of some degrees. In below model, a part extracted from the pilot model under development is used. It contains three ensembles each containing 50 neurons, and those ensembles are connected to form an interactive activation and competition network to evaluate the status of the aircraft weapon inventory. Three types of weapons (gun, beyond visual range missiles (BVR) and within visual range missiles (WVR)) are listed in the inventory and employment selection of the weapon competes between these three ensembles or nodes. Such a network does not require a strong accuracy since a numerical output is not expected. The result is much more preferred as the arousal of an idea of weapon selection. Each node is fed from an external stimulus representing the acquired sensory output and connected to other nodes to inhibit. Therefore, each node is one to one and bilaterally connected to other ones.

In our specific case, the gun node is subjected to white noise of different amplitudes. On previous paragraph, the "b" neuron was subject to white noise individually. This time, the "gun" node is formed of 50 spiking neurons and each of these neurons are subject to white noise individually.

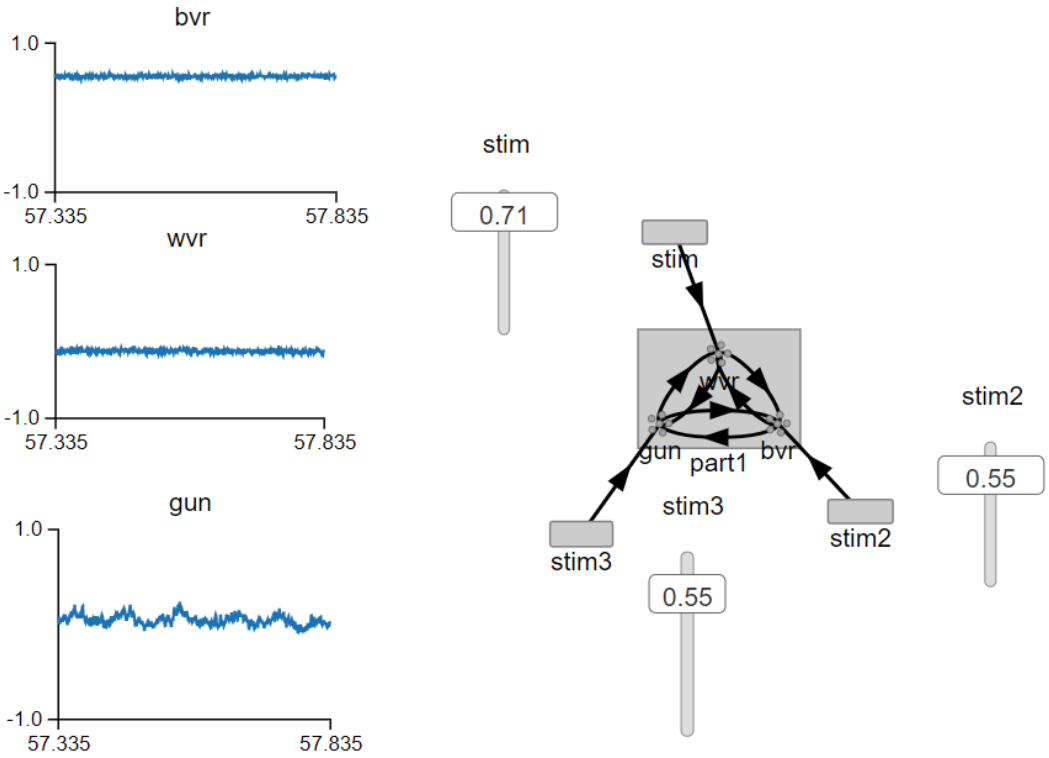


Figure 21. IAC model for inventory evaluation.

Figure 21 represents a visual description of the NENGO model and decoded voltage output graph for three nodes. The noise effect in gun node is clearly observable in the graph. The code block given below summarize the structure of the model and parameters and important aspects are highlighted.

```
import NENGO
import NENGO.spa as spa
import matplotlib.pyplot as plt
import numpy as np
process = NENGO.processes.WhiteNoise(dist=NENGO.dists.Gaussian(0, 0.1), seed=1)
model = NENGO.Network()
with model:
   stim = NENGO.Node([0])
   stim2 = NENGO.Node([0])
   stim3 = NENGO.Node([0])
   part1= NENGO.Network()
   with part1:
      gun = NENGO.Ensemble(n_neurons=50, dimensions=1, noise=process)
      wvr = NENGO.Ensemble(n_neurons=50, dimensions=1)
      bvr = NENGO.Ensemble(n_neurons=50, dimensions=1)
      wsl=0
      wls=-1.5
      wlg=-1
      wgl=-0.1
      wsg=-0.8
      wgs=0
      X= -0.8
      NENGO.Connection(wvr, bvr, transform=wsl)
      NENGO.Connection(bvr, wvr, transform=wls)
      NENGO.Connection(gun, wvr, transform=X)
      NENGO.Connection(gun, bvr, transform=wgl)
      NENGO.Connection(bvr, gun, transform=wlg)
      NENGO.Connection(wvr, gun, transform=wgs)
   NENGO.Connection(stim2, bvr)
   NENGO.Connection(stim, wvr)
   NENGO.Connection(stim3, gun)
```

The white noise is applied to gun node, but the inhibitory weight of the synaptic transformation between the gun and the WVR node is adjusted to some coefficient using the variable X. In figures 22, 23 and 24; outputs' comparisons of different synaptic weights applied to connection between gun and WVR are shown. In these examples, to provide proper visual observability, a very high amount of noise amplitude is applied. Therefore, the figures may not reflect biologically observable levels of noise. The figures 22 and 23 contain a situation where nodes are not externally excited. Oppositely, Figure 24 includes external stimulus on both gun and WVR nodes and a very strong inhibitory synaptic weight.

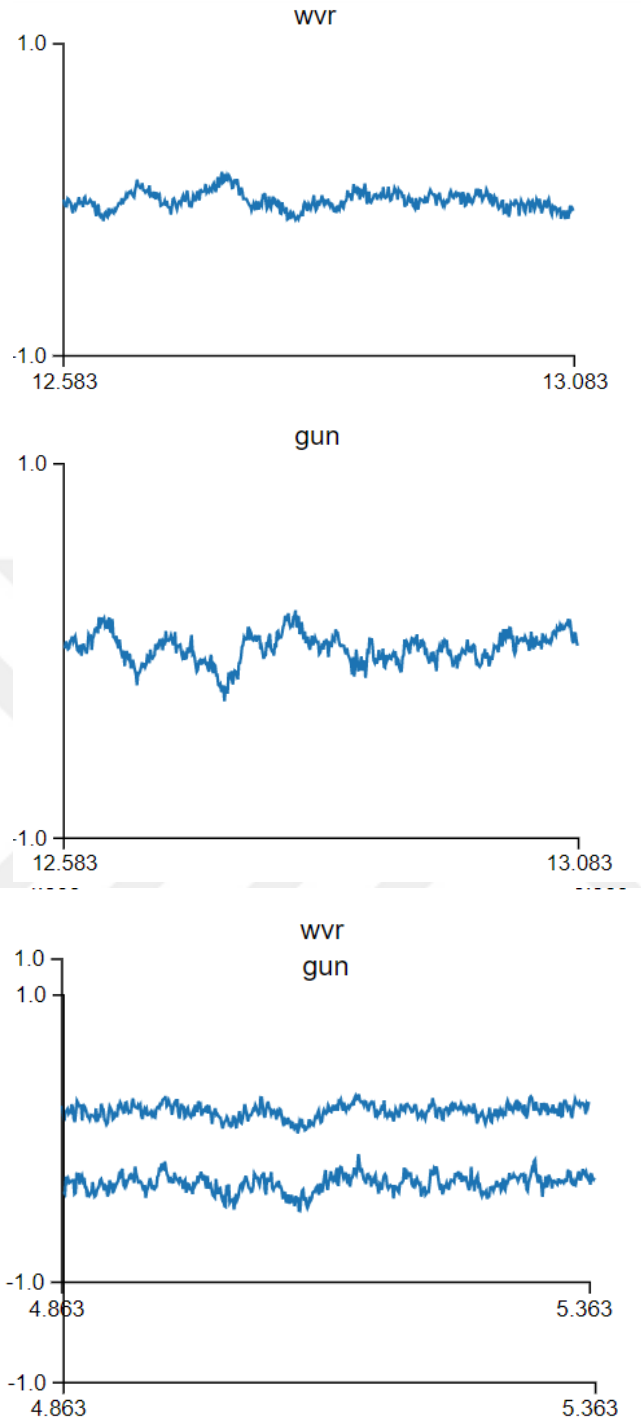


Figure 22. Noise applied to gun, Synaptic weight X = -0.8.

For figure 22, Both nodes are excited by only self. Not external stimulus. On the right hand side, clearly the parallelism and ratio (due to X) between the outputs of both nodes are shown.

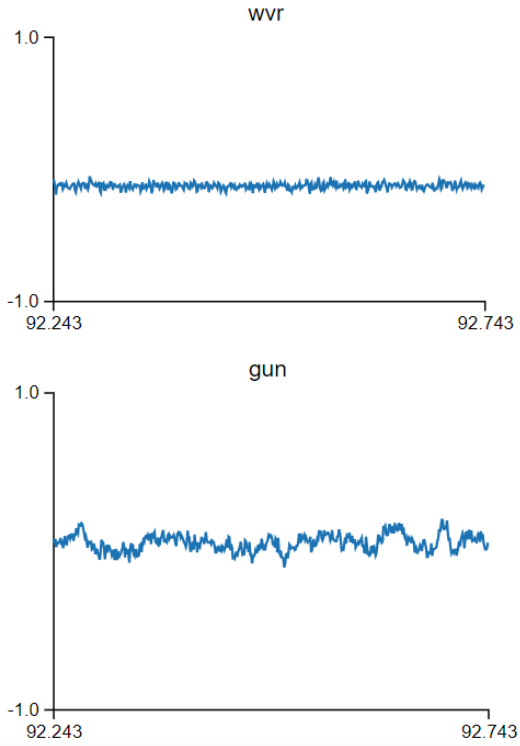


Figure 23. Noise applied to gun, Synaptic weight X = 0. Both nodes are excited by only self. No external stimulus.

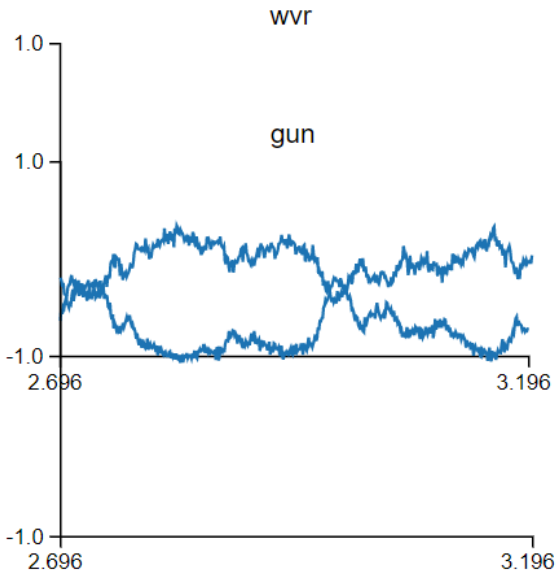


Figure 24. Noise applied to gun, a heavy inhibitory Synaptic weight X = -2.

For Figure 24, External stimulus exists. Notice WVR being strongly inhibited. But noise is clearly increased by the synaptic functionality.

Above three figures show that, the synaptic weight is not only transferring the output due to the external stimulus but also the output due to the noise also. If the function F represents the synaptic transformation, I the input and n is the noise, then output O is;

$$O = F(I) + F(n)$$ Eq (10).

While the existence of multiple neurons and different frequencies, the network does not benefit from the constant power spectral density of white noise, since in any time t, there is a different noise in each neuron and sum of all encoded neural noise leads to an observable fluctuation in the decoded output. Even the noise may not be such strong as in the above examples and synaptic weights are not required to be in amplifying nature, still the outputs are subject to direct transfers from inputs' and nodes' noises.

In this Paragraph, a further complex function including learning and numerical computation will be checked for noise effects. In this specific network a Prescribed Error Sensitivity learning (a specific type of Supervised Learning) mechanism which is built-in implemented in NENGO is used to learn squaring of inputs. The model is given in Figure 25. Each node represents an ensemble of 100 neurons and total number of neurons in the model is 300. The model originally is taken from NENGO community (NENGO.ai, 2018) and is modified for additional white noise. The learning rule PES continuously modifies the synaptic weights between ensembles of neurons to minimize an external error signal until a stopping signal is produced (NENGO.ai, Examples and Tutorials, 2018). The rule implementation is clearly observable in the Figure 25. In the relevant code block below, important points are highlighted.

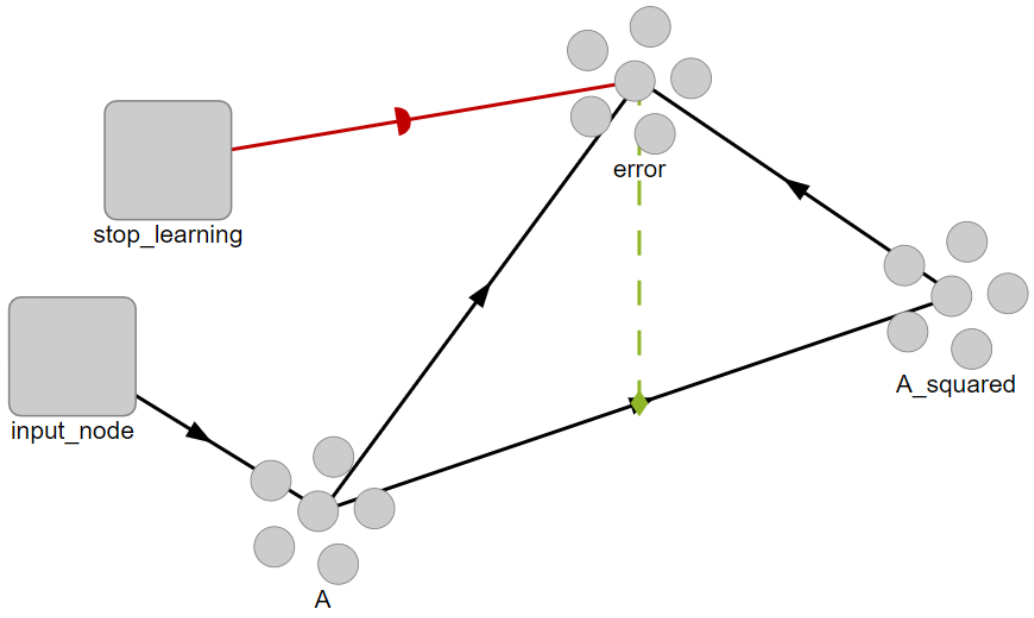


Figure 25. PES learning model for squaring function implementation.

The input node A is subjected to noise, in Figure 26 results for both heavy and light noise applications are shown.

```
import NENGO
import numpy as np
process = NENGO.processes.WhiteNoise(dist=NENGO.dists.Gaussian(0, 1), seed=1)
model = NENGO.Network()
with model:
   # Create the ensemble to represent the input, the input squared (learned),
   # and the error
   A = NENGO.Ensemble(100, dimensions=1, noise=process)
   A_squared = NENGO.Ensemble(100, dimensions=1)
   error = NENGO.Ensemble(100, dimensions=1)
   # Connect A and A_squared with a communication channel
   conn = NENGO.Connection(A, A_squared)
      # Apply the PES learning rule to conn
   conn.learning_rule_type = NENGO.PES(learning_rate=3e-4)
      # Provide an error signal to the learning rule
   NENGO.Connection(error, conn.learning_rule)
         # Compute the error signal (error = actual - target)
   NENGO.Connection(A_squared, error)
         # Subtract the target (this would normally come from some external system)
   NENGO.Connection(A, error, function=lambda x: x**2, transform=-1)
with model:
   # Create an input node that steps between -1 and 1
   input_node = NENGO.Node(output=lambda t: int(6 * t / 5) / 3.0 % 2 - 1)
   # Connect the input node to ensemble A
   NENGO.Connection(input_node, A)
   # Shut off learning by inhibiting the error population
   stop_learning = NENGO.Node(output=lambda t: t >= 15)
   NENGO.Connection(stop_learning, error.neurons, transform=-20 *
np.ones((error.n_neurons, 1)))
   with model:
   input_node_probe = NENGO.Probe(input_node)
   A_probe = NENGO.Probe(A, synapse=0.01)
   A_squared_probe = NENGO.Probe(A_squared, synapse=0.01)
   error_probe = NENGO.Probe(error, synapse=0.01)
   learn_probe = NENGO.Probe(stop_learning, synapse=None)
```

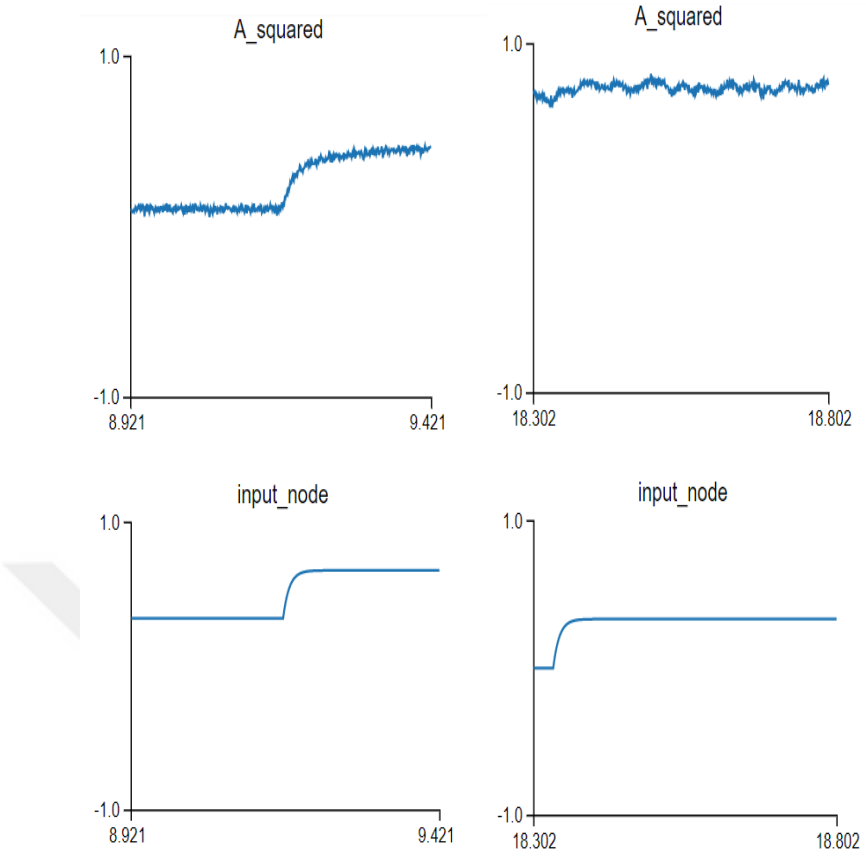


Figure 26. Squaring function learning under noise is represented. Left hand side graphic shows the case where there is no noise in the input A..

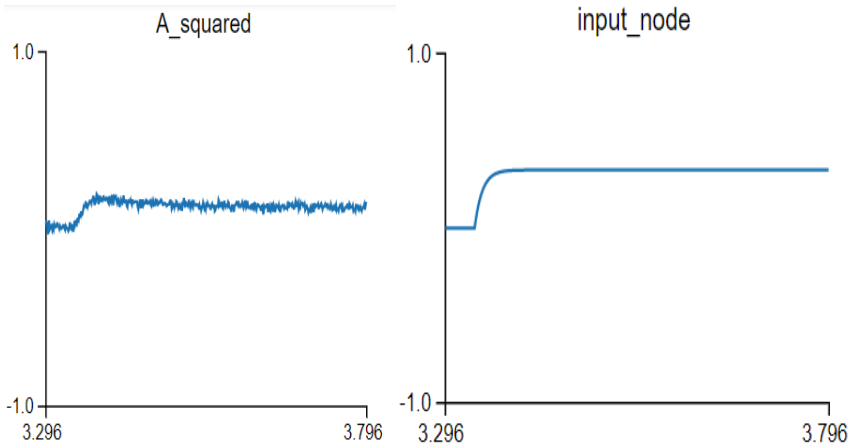


Figure 27. Applied Gaussian noise amplitude in peak is 10 times higher than the given in previous example.

The effect of noise on complex functionality is clearly observable in Figure 15 and 16. All graphs in the Figure show the decoded output of the nodes A square and A before the stop learning halting occurs. Therefore, they all represent the same system state. On the other side, in the left hand side graph in Figure 15, the system functions properly by trying to

take the square of the input. The signal shape relation between the input and the output is visually observable. In the Figure 16, a severe input white noise is applied and output functionality is completely loss. The noise affecting the system disturbs the error computations and PES rule do not work as required or planned. In the right hand side, it is observable that functionality is disturbed but still exist under noise when the noise peak amplitude is 10 times lower than the given in Figure 16. While both noisy application contains constant power spectral density and average noise of zero, the difference in the temporary amplitude values of noise disturbs the functionality in different levels, where higher noise means higher destruction in the functionality of the overall model. The destruction mechanism is left out of scope of this study, but it can simply be deduced that the root cause is the noise added which is the only difference between the Figure 26 left hand side and The Figure 27.

Above examples are also focused on how noise is causing defects on the functionality of the network. On the other hand, noise exists in the nature and may not be solely and totally harmful. Olin-Amentorp et al. (2021) shows also it can be useful on handling and tolerating the synaptic weight variances (which are also existing in the nature).

> *"However, when noise is introduced to the neuron, its output is no longer strictly dependent on its weights, and its behavior can more slowly diverge from its previous state as its synaptic weights are perturbed. From this, we predict that spiking networks using noisy neurons can be more resilient to synaptic inaccuracy." (Olin-Amentorp et al., 2021)*

5.2 Symbolic Graph Methods and SNNs

In chapter 2 of this thesis, non-connectionist types of pilot models such as control theory and symbolic ones are also summarized with some connectionist methods. On the other side, connectionism is not the only limiting or defining factor for the biological plausibility or robotic applicability of a network model. In fact, Aytekin (2022) showed that all functions implemented via a neural network can also be implemented lossless, i.e. with no degradation or performance losses, via decision trees. Clearly Decision Trees (DT) are also networks, even strong machine learning and artificial intelligence methods. Oppositely, DTs and derivations are symbolic are non-connectionist and can also be implemented embodied in robotics. The method selected for that comparison is to implement a symbolic network (a directed graph method, Steging, (2018)) with the same method for the high level pilot decisions and perform a qualitative analysis. As most applied examples of symbolic graph methods, decision trees, decision forests and random forests can be observed.

A decision tree is a machine learning model used for classification and regression tasks. Decision trees are one of the most powerful and popular approaches in data science (Rokach, 2016). Decision Forests aim to improve the predictive performance of decision trees by training multiple trees and using their predictive power in conjunction. Research on decision trees and decision forests started more than 50 years ago and decision forest approaches are performing quite competitively with contemporary algorithms. One of the most prominent approach of decision forests is random forests algorithm which has solid mathematical background and introduced nearly 20 years ago by famous statistician Breiman (2001). While boosting methods are also proven to be effective in many researches.

Combining decision trees into decision forests have been proposed by many researchers in the field and one successful implementation of decision forest idea is random forests algorithm. Random Forests are an effective tool in machine learning. Random Forests are proposed by Breiman in 2001 as an improvement for his earlier Classification and Regression Trees (CART-1984) research. The framework of Random Forests consists of strong individual predictors (decision trees). Their combination and correlation enables Random Forests to predict. The forest is randomized by randomly selecting features to split each node. This increases the performance and Random Forests reach the performance level of boosting algorithms. The random feature selection, hence the name Random Forest, as a result makes the forest more robust to noise.

Performance of a decision forest depends on the performance of individual trees in the decision forest and correlation between them. Boosting and arcing algorithms have the ability to reduce bias as well as variance when applied to decision trees. Random Forests give results competitive with boosting and adaptive bagging, yet do not progressively change the training set like boosting and arcing algorithms. Breiman concluded that Random Forests' accuracy indicates that they act to reduce bias. It is shown in literature that random inputs and random features produce good results in classification tasks. Random Forests are also applicable to regression tasks (Breiman, 2001).

Random forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. The generalization error for forests converges a.s. (almost surely) to a limit as the number of trees in the forest becomes large. Note that almost surely convergence is the strongest type of convergence in probability theory (Stark et al. 2002). Also because of the Strong Law of Large Numbers, Random Forests do not overfit the training data. The almost surely convergence and the robustness to overfitting are main strengths of Random Forests. On the other side a decision tree is defined as a weak learner which means that s decision trees are only slightly correlated with the true classification and is only slightly better than random guessing. The term boosting refers to a family of algorithms that are able to convert weak learners to strong learners (Zhi-Hua, 2012). Boosting methods also shine in decision forests approach (Breiman, 1996).

While a decision tree has many advantages, such as comprehensibility, it still suffers from several drawbacks—instability, for instance. One way to realize the full potential of decision trees is to build a decision forest. A decision forest, as its name implies, consists of several decision trees in which their predictions are combined into a final prediction. By building a forest, the mistakes of a single decision tree are compensated for by other decision trees in the forest.

The idea of generating a predictive model that combines several models has been investigated since the seventies when Tukey (1977) introduced an ensemble of two linear regression models. Tukey suggested fitting the first linear regression model to the original data and the second linear model to the residuals.

It has been repeatedly shown that ensemble learning improves the predictive performance of a single model. Ensemble learning works particularly well when decision trees are used as the base models. Combining the ideas of ensemble learning and decision tree learning formed the notion of decision forests. Decision forests are frequently used in many real-world applications in various domains such as medicine, engineering, and information retrieval. A decision forest illustration representing the basic idea is given in Figure 28.

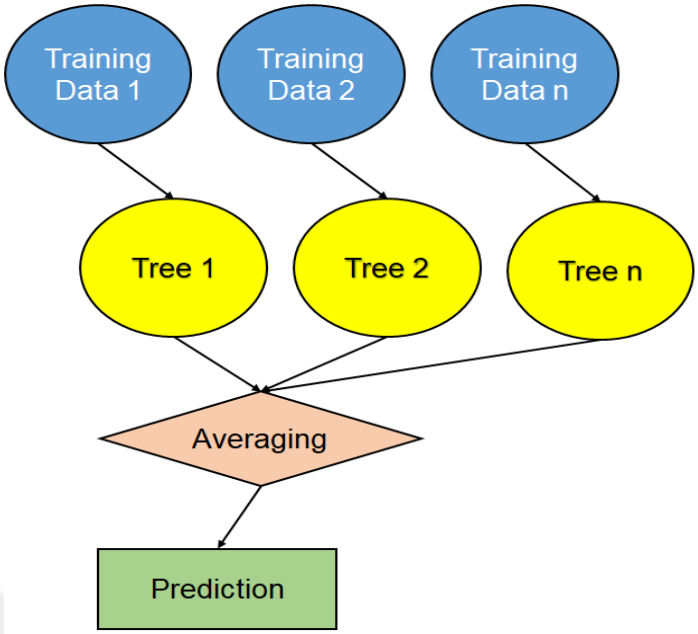


Figure 28. Decision Forest Illustration

Random Forests create a decision forest from ensemble of decision trees by random sampling, hence the name. Random Forests are flexible and easy to use machine learning algorithms. Random Forests perform good generally even without hyper-parameter tuning (Tran and Hieu, 2019). Note that there are very few hyper parameters to tune. These parameters are limited to high level parameters such as maximum depth, quality of split criterion and such. Random Forests are widely used due to being simple to understand and use and the fact that the mathematical base was laid strongly by Breiman (2001). It is shown explicitly in literature that Random Forests can be used both for classification and regression tasks.

Random Forests are also handy algorithms, because their default hyperparameters often produce a good prediction result. The number of hyperparameters is also not that high and they are straightforward to understand. One of the big problems in machine learning is overfitting, but most of the time this will not happen that easy to a random forest classifier. That is because if there are enough trees in the forest, the classifier will not overfit the model. The main limitation of Random Forest is that a large number of trees can make the algorithm slow and ineffective for real-time predictions. In general, these algorithms are fast to train, but quite slow to create predictions once they are trained. A more accurate prediction requires more trees, which results in a slower model. In most real-world applications the random forest algorithm is fast enough, but there can certainly be situations where run-time performance is important and other approaches would be preferred. And of course Random Forest is a predictive modelling tool and not a descriptive tool.

Accuracy – Random Forests is competitive with the best known machine learning methods.

Instability – if the data is changed, the individual trees may change but the forest is relatively stable because it is a combination of many trees.

To understand the Random Forest algorithm, one first shall look and understand the decision tree algorithm. Decision tree algorithm was first proposed by Breiman et al. (1984) proposing a learning mechanism based on simple decision trees. Decision trees can be used both for classification and regression tasks. However, decision trees have a major drawback, they are prone to instability in real applications. This problem arises when a single tree cannot comprehend a real problem alone. Decision trees are also prone to overfitting and overfitting decreases the reliability of the algorithm further for real world problems.

Decision Trees are considered to be one of the most popular approaches for representing classifiers. Researchers from various disciplines such as statistics, machine learning, pattern recognition, and Data Mining have dealt with the issue of growing a decision tree from available data. Most of the current methods for constructing decision tree classifiers are in a top-down manner (Rokach and Maimon, 2005).

Each node has a left branch and right branch. The tree grows by adding more nodes branching from ancestor nodes. The right-sized trees can be achieved by pruning and use of a test sample or cross-validation (Breiman et al. 1984). Pruning is the operation of removing a subtree from the tree to optimize the size of the tree. According to Breiman et al. (1984) the tree complexity has a crucial effect on its accuracy. The tree complexity is explicitly controlled by the stopping criteria used and the pruning method employed. The pruning can be based on different measures such as information gain, GINI index and so on. Information gain is an impurity-based criterion that uses the entropy measure as the impurity measure.

Although decision trees are considered powerful for many applications they have substantial deficiencies in many areas that makes them not applicable for non-ideal (real world) scenarios. For example, the greedy characteristic of decision trees leads to the disadvantages such as its over–sensitivity to the training set, to irrelevant attributes and to noise (Salzberg, 1994).

To improve over these weaknesses researchers have proposed many different approaches and methods. One of them is to use more than one model and have them vote on the outcome. Many researchers have concluded that this principle bagging improves the performance by reducing variance and decreasing effects of overfitting. This type of learning is called ensemble learning and methods proposed are called ensemble methods.

Boosting is an alternative approach to bagging algorithms where each prediction alters the weights of the dataset and thus the model learns from not only the dataset but also from the previously trained samples' performances. The dynamic training increases the performance of the decision forest. Gradient boosting is a special boosting method were errors are minimized by gradient descent algorithm. And finally Extreme Gradient Boosting (XGBoost) is an optimization technique further improving the boosting and hence the decision forest performance.

Decision Forests provide accurate fast and powerful prediction as shown by many researchers. Decision Forests are built upon decision trees and the concept of bagging and maintain all benefits of them except surrogate split concept. Decision Forests greatly decrease between-tree correlation and reduce instability as proved by Breiman in his original work. Modern implementations of Decision Forests enable parallelization thus improving the training time. Decision Forests approach improves the generalization ability of a single decision tree which is defined as a weak learner by combining the output of several decision trees to form a strong learner. Main aspects of decision forests are growing, thinning and combining. These steps are important for the performance of a decision forest. Currently research areas in decision forests include but not limited to

developing distributed implementation for addressing the big data challenge; developing decision forest methods for various learning tasks or settings other than the general purpose regular classification and regression; developing static and dynamic methods for decision forest thinning.

Random Forests do not overfit due to the Law of Large Numbers so they are effective tools in prediction. Also including the right kind of randomness makes Random Forests accurate for classification and regression tasks. Bagging and arcing algorithms improves over decision trees just like Random Forests in terms of reducing bias as well as variance. However, these methods progressively change the training set. The Random Forests algorithm on the other hand does not change the training set progressively yet give results competitive with bagging and adaptive boosting.

Boosting is another powerful method for combining multiple weak learners to form a committee (or jury) and vote on their prediction to form a combined learner which forms a strong learner and hence performs always better than single predictors. The main difference between boosting and bagging methods is that the base predictors are trained in sequence and each base classifier is trained using a weighted form of the dataset. The weighted form of the dataset is formed by the weighting coefficients associated with each data point depends on the performance of the previous predictors. Once the sequential training is completed the results are weighted through majority voting just like bagging algorithms.

To be able to model the high level decisions with a symbolic graph method, a partial training dataset extracted from the original set, containing the inventories, missions, positions, attitudes and velocities of two aircrafts is employed. Two different methods are implemented. In the first method, a regular decision tree is built to see the performance on data set, using WEKA. Figure 19 gives a representation of the DT built using the data set.

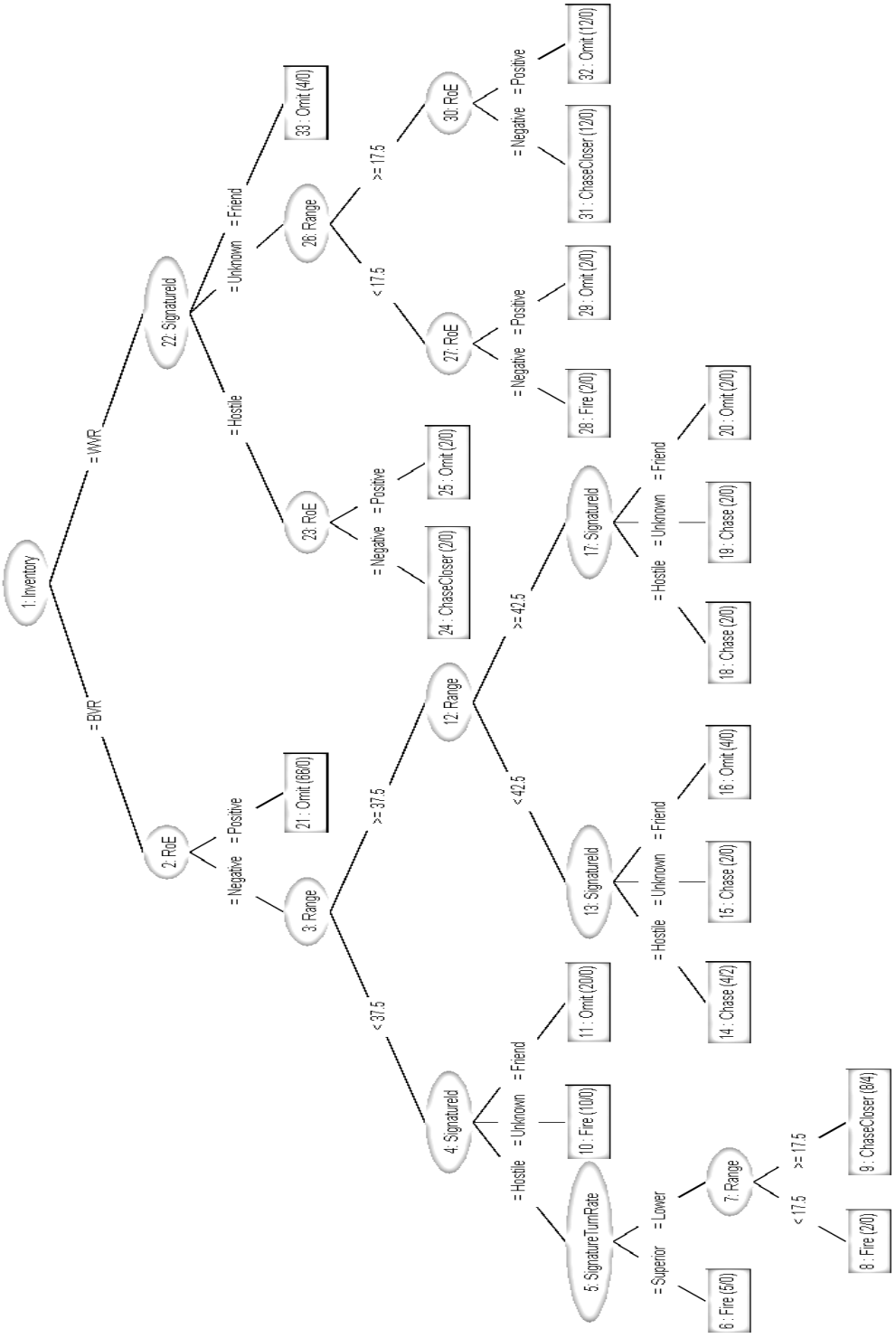


Figure 29. The Decision Tree Model for Pilot's High Level Decisions

In the second run, a random decision forest, which contains 500 random trees, is built again using the same tool. The decision tree could classify only 86.5 % of the available data. The random forest performance is found around 92%. While the performance of the decision forest is just ~7% bigger than the decision tree, a major difference on the content

of the incorrect decisions is observed. The Decision Forest obtained a negative result slightly below 8% but none of the wrong decisions were harmful. On the other side, the decision tree method gave two firing decisions in positions where in fact the target should be omitted. These two cases are clearly "*friendly fire*" situations. As expected *friendly fire* situations are one of the scariest scenarios that can happen in aerial military missions. Additionally, in five cases where the target should be chased, the decision tree method gave firing decisions earlier than expected. Finally, one of the targets that should be fired is omitted. Therefore, some of the DT method decisions were strongly wrong in comparison with DF results. At the end, it is observed that the DF method can be very useful for giving high level decisions in aerial combat pilot modelling while having some negative aspects. Some notable observations are listed below, while neglecting inherent weaknesses of DTs such as instability:

- The decision trees, forests and random forests have potential to be used in aerial robotics. But such applications may require, especially for forests, extreme processing power while leading to difficulty in robotic embodiment.
- Pilot decisions are extremely complex and a training or test set may not always cover overall engagement environment. These decisions are open to the effect of the emotional state of the human agent and non-cognitive architectures are not strong enough to fulfil such gaps. It should be noted that according to Salzberg (1994), DT algorithms are greedy and so open to the effects of noise. A DT may increase the undesired instability in an input to leading unexpected results.
- Spiking neuron networks (SNN) are biologically plausible. A DT can easily be constructed using a SNN. Especially NENGO can easily be used in the implementation of such a network thanks to its encoding capability via tuning curves. Therefore, it may be accepted that DTs may find some space to themselves in the biologically plausible algorithms list even there is no such evidence. Still, a DT is clearly implementable in a SNN. On the other hand, the DT is shown even for a partial and relatively simpler data set, not capable of covering all data set. Yet in complex functions including lower level decisions in conjunction with high level ones, DTs would potentially fail.
- Being a merger of hundreds of DTs, forest methods of course will require -in fact linearly proportional, and 500 times more in our example- much more processing power. In the chapter 7 and 8 of this thesis, a model and its processing will be provided. IT is observed that for a simple SNN of some thousand spiking neurons, hardware became the limiting item for the real time factor. For a forest of high level pilot decisions, potentially an implementation on board will be strongly behind real time.
- DTs are symbolic. So the forests are. On the opposing, Connectionism begins with the acceptance that the semantic arises from the network itself, i.e. syntax embedded in the network, and embodiment and situatedness is brought to life via this connectionism. With this axiom, DTs and forests being graph methods are in fact not proper neural networks and need to left out of context for connectionism.
- Randomness employed in the above symbolic methods are not biologically plausible. Spiking neuron models are not intending solely on providing strong cognitive models, but also ease the understanding of the human nature. Therefore, biologically inspired methods such as DFT or SNN are strongly cognitive in comparison to the DTs and forests.

- Similar to control theory models of pilots, which are in fact not pilot models, but are just some control models imitating some partial functionality of pilot for engineering tasks, DT and forest methods used in pilot modelling may only be some engineering models for some partial functionality.

5.3 Pilots' Evaluation, Results, Discussion And Conclusion

For the validity decisions (first level), the pilots agreed with each other in 75% of the scenarios. For the scenarios where the TOI is out of border, and for the scenarios where the TOI is a friendly or a neutral unit, both pilots stated that the model is providing the correct outputs or making a decision that they fully concur. In the case of hostile targets inside the national borders (i.e., intruding hostile scenarios), according to the pilot the model's evaluation of the validity of the target was wrong. In the scenarios of intruding hostile TOI which is occurring in peacetime, the model decided the TOI is not a valid target. The pilots also disagreed with each other in these scenarios, but a possible decision is that the targets might be valid and the model outputs are wrong. The confusion in this type of scenario is mostly due to the conflict between the inputs. While the status of the TOI is intruding and hostile, the time is peace season, and Rules of Engagement (ROE) are limiting the engagement. In such cases, limited training sets, and the illogical situation which is hard to model, creates a weakness for the potential pilot model candidates. Additionally, for the cases where an intruding hostile TOI, which do not engage or RADAR lock our ownship also yielded to confusion. Pilots did not agree among themselves and with the model if the target is valid. This rare situation also created a weakness for the model. But, it should be underlined that pilots also provided inconsistent decisions and evaluations in between. Table 2 shows the outputs of the model for each scenario.

Table 2. Decisions by the model.

#	**Decision**
1	The AI decided the target is invalid and decided just to chase from distance
2	The AI decided the target is still invalid and decided to chase from engagement distance
3	The AI decided the target is still invalid and decided to chase from engagement distance
4	The AI decided the target is a valid target and decided to abort and alert other friendly units since range is invalid
5	The AI decided the target is a valid target but decided to hold a lock on target and to chase
6	The AI decided the target is an invalid target but decided to chase the target from a very close engagement distance
7	The AI decided the target is a valid target and decided to chase the target for an engagement
8	The AI decided the target is a valid target but decided to abort to distance and alert friendly units
9	The AI decided the target is a valid target and decided to lock on for engagement. Notice AI did not fire a BVR even target is in range
10	The AI decided the target is a valid target and decided to abort and alert all friendly units
11	The AI decided the target is a valid target and decided to engage while holding a BVR missile ready
12	The AI decided the target is invalid and decided to neglect

For final action decisions (second level), the pilots agreed with each other in 67% of the scenarios. In 75% of the incorrect decisions (non-agreement decisions simply notified as incorrect or wrong), the wrong decision is a result of the first level wrong decision. Since the validity of the TOI is evaluated wrong or inconsistent, the action is also evaluated directly wrong by the model. But in the last scenario, where the TOI is clearly hostile, intruding, and locked to ownship, there is a strong complexity in the evaluation of the model output. While the model decided the TOI is validly a target and shall be chased in BVR distance, one of the pilots decided to fire a BVR directly, while the other pilot preferred to act in agreement with the model. In this, very specific case, what is observed is that firing a missile is a very hard decision, even the most experienced pilots hesitate on committing such an act even if all inputs, status, and ROE are compliant with a proper fire decision. Therefore, it may be argued that the model output did not give a proper fire decision in this instance, but it also showed hesitation in parallel to its training set. We may clearly underline that the model did not provide a fire decision in any of the scenarios evaluated by the pilots. Table 3 summarizes the evaluation of the pilots of the scenarios.

Table 3. Pilot evaluation status and disagreement scenarios.

#	**Does Any of The Pilots Agree with the Model?**	**Does Pilots Agree in Between?**	**Scenario**
1	Agree	Agree	
2	Agree	*Disagreement (1)*	Intruder hostile inside borders, inventory and range is OK and bandit is NOT locked to ownship, engagement on peace time
3	Agree	*Disagreement (2)*	Ambiguous identification, bandit or neutral just close to the borders, engagement on peacetime
4	Agree	Agree	
5	Agree	Agree	
6	Agree	*Disagreement (3)*	Intruder hostile inside borders, inventory and range is OK and bandit is NOT locked to ownship, engagement on tension time
7	Agree	Agree	
8	Agree	Agree	
9	Agree	*Disagreement (4)*	Clear hostile intruder inside borders, inventory and range is OK and bandit locked to ownship, engagement on tension time
10	Agree	Agree	
11	Agree	Agree	
12	Agree	Agree	

The scenarios in which pilots disagreed with each other will be further discussed in the next section. However, it shall be noted that all scenarios where pilots disagree mean that at least one pilot disagrees with the model.

Discussion and Conclusion

All the scenarios that involve a disagreement between the model and a pilot or in between the pilots present a difficulty that requires further analysis and clarification. For the disagreement (1), there is a clearly hostile aircraft inside national borders or mission territory but it is unexpected since it is peacetime, therefore, the emotional or affective inputs of the networks are suppressed. The model decided that it is an invalid target and just needed to be chased. Pilot-A disagreed with the model and thinks that the target is valid. In disagreement (2), the target is unidentified but all other inputs reflect that it is not a target. The model decided to chase it as an invalid target. Pilot-A decided that it shall not be chased and can be neglected. In disagreement (3), a hostile intruder exists in the national borders or mission territory but it is not locked onto ownship and is returning out of borders. The model decided to chase it, but one pilot decided that it could be omitted. Finally, in disagreement (4), the target is hostile inside borders in tense time and is locked to ownship for engagement. The model decided it is a valid target but decided to abort and alert. But one of the pilots decided that the validity is true but the action should be employing a BVR missile.

As it can be seen in the model evaluation study, the SPA architecture provides compositionality and understandability to the outputs of the model, so sentences like "model decided to abort" is possible. This is a great benefit of NEF and its SPA architecture. Additionally, with usage of higher number of SPAs in the model ambiguities could be easily represented. While in this model, a lower number of SPs is preferred for simplicity, it could be done in principle. When we look at the disagreement scenarios, conflicts in peacetime or strongly dangerous cases, which are very few in the training sets, are coming forth as obviously problematic.

The study contained only 13 different inputs to define the environment and the ownship. Oppositely, real-life scenarios contain a higher number of factors in comparison to the current model. Lower-level networks should be trained with more inputs without neglecting any potential effectors. Additionally, in a real cockpit, the pilots may not be always capable of evaluating so many inputs at the same time and reaction times may vary. For a better model, such effects may be considered. Additionally, in this modeling study, some inputs such as the background of the pilot or the tracking history of the TOI is not included. Larger training sets, containing tracking history of the TOI and more inputs need to be used. Currently, the training sets are manually produced and do not contain many scenarios with firing outputs. Such a training set makes the model; "a model of peacetime pilot". Therefore, it is clearly understandable the outputs' lag on fire decisions. Gathering training sets from simulator flights may be a promising method for future model-building efforts. A larger pilot pool may be needed to understand the conflicts/disagreements among them and their evaluation of the scenario elements. Especially, in environments with complex or conflicting inputs, illogical situations, such as hostile intruder in peacetime with no focus on ownship may lead the model to illogical or wrong decisions. Therefore, the training sets need to be enlarged with unexpected scenarios. Despite all the above lacking issues, our findings suggest that the NEF and the spiking neuron-based modeling approach provide a strong infrastructure for building and testing such complex models. If subnets are prepared (using the same infrastructure) with a further focus on a stronger evaluation with a larger pilot pool, the outputs of such a model may be dramatically improved. In applications where enough processing power is provided to an embedded and deterministic multi-layer subnet, it is pretty clear that NEF-based models may provide perfectly pilot-like decisions.

For a cognitive model which does not depend on measurement or experimental data such as neuroscientific imaging, the majority of the inputs are depending on subjective testing depending on scenarios. Such data is also used for training sets. In this example, the training set was limited to the evaluations of only seven pilots. Model outputs are also evaluated by only two pilots. Further increased numbers of subjects may rise more conflicts in the evaluation given the complexities of aerial combat.

On the other hand, the affective state of the pilot is contributed to the model only with a single input which could affect the network in a completely suppressing way or vice versa. But the topic is very open to further study. Pilot or not, modeling the effects of human emotions on decisions is a difficult undertaking that requires further inquiry. The results suggest that the model hesitated in every case in giving a “fire” or “kill” decision. Since the training sets also do not contain enough scenarios creating the necessity for such cases. For more aggressive and popular fire decisions, war-type scenarios shall be added, and tension times need to be further studied.

Overall, it is observed that spiking neuron-based models and the Neural Engineering Framework approaches can effectively support modeling studies for complex human tasks and may be used in generating human-like behaviours. Such tools are very promising for aerial decision support systems where the behaviour of the opponents is needed to be estimated or anticipated in advance. Since being explainable, autonomous agents or Computer Generated Forces may have areas of applications in aerospace.

REFERENCES

Amari (1977), Amari S. I., Dynamics of pattern formation in lateral-inhibition type neural fields. Biological Cybernetics, vol. 27, p.77–87.

Anderson, (2002). Anderson J.R., ACT: A Simple theory of complex cognition. In Polk T.A. and Seifert C.M. (Eds.), Cognitive Modelling. The MIT Press.

Anderson (1970), Anderson R.O., A new approach to the specification and evaluation of flying qualities. AFFDL-TR-69-120. USA Air Force Flight Dynamics Laboratory Report.

Anderson et al. (2004); J. R. Anderson, D. Bothell, M. D. Byrn, S. Douglass, C. Lebiere and Y. Qi, An Integrated Theory of the Mind, in Psychological Review, 2004, Vol.111, No.4, pp.1036–1060.

Aytekin (2022); Ç. Aytekin, Neural Networks Are Decision Trees, arxiv.org. arXiv: 2210.05189v2 [cs.LG] 17 Oct 2022.

Backhaus et al., (2013); S. Backhaus, R. Bent, J. Bono, R. Lee, B. Tracey, D. Wolpert, D. Xie, and Y. Yıldız, Cyber-Physical Security: A Game Theory Model of Humans Interacting Over Control System, IEEE Transactions On Smart Grid, Vol.4, No.4, pp. 2320-2327.

Bartlett et al., (1998), Bartlett, P., Freund, Y., Lee, W. S., & Schapire, R. E., Boosting the margin: A new explanation for the effectiveness of voting methods. *The annals of statistics*, *26*(5), 1651-1686.

Blakelock (1991); J.H. Blakelock, "Automatic Control of Aircrafts and Missiles" (2nd ed.), John Wiley and Sons publishing, New York, ISBN 0-471-50651-6

Berk (2008), Berk R.A., Random Forests. In: Statistical Learning from a Regression Perspective. Springer Series in Statistics. Springer, New York, NY

Berkeley (2008), Berkeley I.S.N., CajunBot, A case study in Embodied Cognition, in Handbook of Cognitive Science, an Embodied Approach, Calvo P. and Gomila T. (eds.), Elsevier Press, 2008, NY USA, ISBN: 978-0-08-046616-3.

Bekolay et al., (2014) Bekolay T., Bergstra J., Hunsberger E., DeWolf T., Stewart T. C., Rasmussen D., Choo X., Voelker A.R., Eliasmith C., 2014. NENGO.: A Python tool for building large-scale functional brain models. In Frontiers in Neuroinformatics. Vol. 7. Article 48. pp.1.13.

Bicho E., Mallet P.,Schöner G. (2000). Target representation on an autonomous vehicle with low-level sensors. In The International Journal of Robotics Research, vol. 19. pp. 424–447.

Bicho, Mallet and Schöner (2010); E. Bicho, G. Mallet and G. Schöner, Target Representation on An Autonomous Vehicle with Low Level Sensors. The International Journal of Robotics Research, Vol 19, pp 424-449.

Boyd, (1995); J.R. Boyd, Essence of Winning and Loosing, The d-n-i echo: The Essence of Winning and Losing, by John R. Boyd (archive.org).

Breiman et al., (1984); Breiman, L., Friedman, J. H., Olshen, R. A.,, Stone, C. J. Classification and Regression Trees. Monterey, CA: Wadsworth and Brooks.

Breiman (1996), L. Bagging predictors. Mach Learn 24, 123–140 (1996). Bagging predictors

Breiman (1996a), Breiman L., Bagging predictors, in Machine Learning 26(2), pp. 123–140.

Breiman (1998a), Breiman L., Arcing classifiers (discussion paper). Annals of Statistics, 26, 801–824.

Breiman (1998b). Breiman L., Randomizing outputs to increase prediction accuracy. Technical Report 518, May 1, 1998, Statistics Department, UCB (in press, Machine Learning).

Breiman (1999), Breiman L., Using adaptive bagging to debias regressions, Technical Report 547, Statistics Dept. UCB.

Breiman (2000), Breiman L., Some infinity theory for predictor ensembles. Technical Report 579, Statistics Dept. UCB.

Breiman, (2001); Breiman, L. Random Forests, in Machine Learning 45, pp. 5–32 (2001).
Breiman, (1996); L. Breiman, Bias, Variance and Arcing Classifiers, University of California, Technical Report 460, http://oz.berkeley.edu/~breiman/arcall96.pdf

Breiman, (1984); Breiman, L, Friedman, J H, Olshen, R A, and Stone, C J, 1984, Classification and regression trees: Wadsworth, Inc.

Bresloff and Coombes (2000); Bressloff P.C., Coombes S. Dynamics of strongly coupled spiking neurons. In Neural Computation, vol. 19. pp.91–129.

Carter and Mancini, (2018); B. Carter, R. Mancini; Op Amps for Everyone. Elsevier. pp. 10–11. ISBN 978-0-12-811648-7.

Choo and Eliasmith (2013); X. Choo and C. Eliasmith, General Instruction Following in a Large-Scale Biologically Plausible Brain Model, 35th Annual Conference of the Cognitive Science Society pp. 322-327.

Cuijpers and Erlhagen (2008); R.H. Cuijpers and W. Erlhagen, Implementing Bayes' Rule with Neural Fields, in V. Kurkova et al. (Eds.): ICANN 2008, Part II, LNCS 5164, pp. 228–237, 2008. Springer-Verlag.

de Souza et al. (2005); L.F.R. de Souza, C.E. Beluzo, E.M. Belo, F.D. Marques. Simulation and Identification of a Flight Envelope Using Neural Network. 18th International Congress of Mechanical Engineering.

DeWolf, Jaworski and Eliasmith (2020); T. DeWolf, P. Jaworski, C. Eliasmith, NENGO And Low-Power AI Hardware For Robust, Embedded Neurorobotics, arXiv:2007.10227v2 [cs.RO] 29 Aug 2020.

Enns R., Si J. (2004). Helicopter Flight Control Using Direct Neural Programming. In Si J., Barto A., Powell W. and Wunsch D. (Eds.), Handbook of Learning and Approximate Dynamic Programming. IEEE Press and John Wiley and Sons Inc.

Eliasmith, C. (2013). How to build a brain: A neural architecture for biological cognition. Oxford University Press.

Eliasmith, C., & Anderson, C. H. (2003). Neural engineering: Computation, representation, and dynamics in neurobiological systems. MIT press.

Eliasmith et al. (2012); C. Eliasmith, T.C. Stewart, X. Choo, T. Bekolay, T. De Wolf, Y. Tang, D. Rasmussen, A Large-Scale Model of the Functioning Brain, Science Magazine 2012 Vol 338 www.sciencemag.org.

Eliasmith, Tang, and DeWolf (2013); C. Eliasmith, C. Tang, and T. DeWolf, Ch.3 Biological Cognition: Semantics, in How to Build a Brain, A Neural Architecture for Biological Cognition, C. Eliasmith (ed.), ISBN 978–0–19–979454–6, Oxford University Press.

Eliasmith, Stewart and Rasmussen (2013); C. Eliasmith, T. Stewart, and D. Rasmussen, Ch.3 Biological Cognition: Syntax, in How to Build a Brain, A Neural Architecture for Biological Cognition, C. Eliasmith (ed.), ISBN 978–0–19–979454–6, Oxford University Press.

Erlhagen and Bicho (2006), Erlhagen W., Bicho E., The dynamic neural field approach to cognitive robotics, in *Journal of Neural Engineering*, vol. 3, no: 3. pp. R36–R54.

Feldman and Ballard (1982); J.A. Feldman and D.H. Ballard, Connectionist Models and Their Properties, in Cognitive Science, vol 6, pp. 202-254.

Fodor and Pylyshyn (1988), Fodor J.A., Pylyshyn Z.W. Connectionism and cognitive architecture: A critical analysis. Retrieved June 18, 2015 from online archive website http://www.sciantaanalytics.com/sites/default/files/fodor-pylyshyn.pdf.

Fuller, (1995); R. Fuller, Ch.3 Fuzzy Neural Networks, in *Neural Fuzzy Systems*, ISBN 951-650-624-0, ISSN 0358-5654.

Ghasemi et al. (2005); Ghasemi R, Nikravesh SKY, Menhaj MB, Akbari S (2005) A real time fuzzy modeling of pursuit-evasion in an air combat. In *Adv Soft Comput* 4:171–184

Gunston, (1988); B. Gunston, Modern Air Combat: The Aircraft, Tactics and Weapons Employed in Aerial Warfare Today, Crescent 1st edition, ISBN 978-0517412657.

Hemion (2013); N.J. Hemion, Ch3. A New Cognitive Architecture Based on Embodied Simulation, in Building Blocks for Cognitive Robots; Embodied Simulation and Schemata in a Cognitive Architecture; Thesis, Blefield Technical University.

Holmes, (1994); G. Holmes, A. Donkin and I. H. Witten, "WEKA: a machine learning workbench," Proceedings of ANZIIS '94 - Australian New Zealand Intelligent Information Systems Conference, 1994, pp. 357-361, doi: 10.1109/ANZIIS.1994.396988.

Hunsberger & Eliasmith, (2016); E. Hunsberger, C. Eliasmith, Training Spiking Deep Networks for Neuromorphic Hardware, DOI - 10.13140/RG.2.2.10967.06566.

Hurzook, Trujillo and Eliasmith (2013); A. Hurzook, O. Trujillo and C. Eliasmith, Visual motion processing and perceptual decision making, Proceedings of the Annual Meeting of the Cognitive Science Society. Vol 35, pp. 2590-2595.

Hutchins (1995), Hutchins E., Cognition in the Wild, MIT Press, A Bradford Book, ISBN 0-262-08231-4.

Hutchins (2000), Hutchins E., The Cognitive consequences of patterns of information flow, in *Intellectica* Vol 1:30. pp. 53-74.

Hutchins and Klausen (1996), Hutchins E., Klausen T., Distributed cognition in an airline cockpit. In Y. Engeström and D. Middleton (Eds.), Cognition and communication at work. New York: Cambridge University Press. pp. 15-34.

Johnson and Pritchett (2002), Johnson N.E., Pritchett A.R., Generic Pilot and Flight Control Model for use in simulation studies. In AIAA Modelling and Simulation Technologies Conference and Exhibit. USA.

Johnson and Busemeyer (2010); J.G. Johnson and J.R. Busemeyer, Decision making under risk and uncertainty. Cogn Sci 2010 vol 1 pp. 736–749.

Kaneshige J., Bull J., Totah J.J. (2000). Generic Neural Flight Control and Autopilot System. AIAA-2000-4281.

Kaneshige J., Burken J. (2008). Enhancements to a Neural Adaptive Flight Control System for a Modified F-15 Aircraft. In Proceedings of the 2008 AIAA GNC Conference.

Karlı, Efe and Sever (2017); M. Karli, M.Ö. Efe, and H. Sever, Extracting Basic Fighter Maneuvers from Actual Flight Data, in Lecture Notes on Information Theory Vol. 5, No. 1, June 2017

Kaygusuz and Çakır, 2015; Y. Kaygusuz, M.P. Çakır, A Dynamic Field Theory Based Pilot Model to Control Aircraft Pitch Attitudes, in Proceedings of EuroAsianPacific Conference on Cognitive Science, EAPCOGSCI 2015.

Kaygusuz, Y. (2015). An Embodied and Extended Cognitive Dynamic Field Theory (DFT) Model for Piloting Tasks Supported with Flight Records. Middle East Technical University, Graduate School of Informatics, Department of Cognitive Science. http://etd.lib.metu.edu.tr/upload/12619297/index.pdf

Kazerounian et al., (2013); S. Kazerounian, M. Luciw, M. Richter and Y. Sandamirskaya, Autonomous Reinforcement of Behavioral Sequences in Neural Dynamics, The 2013 International Joint Conference on Neural Networks (IJCNN).

Koch (1999); C. Koch, Chapter 14 Simplified Models of Individual Neurons, in Biophysics of Computation, Information processing of single neurons, Oxford University Press, 1999, pp. 330-349, ISBN 978-0-19-510491-2.

Laird J. E., Newell A., Rosenbloom, P. S. (1987). Soar: An architecture for general intelligence. Artificial Intelligence, vol 33. pp.1–64.

Lehman, Laird and Rosenbloom, (2006); J.F. Lehman, J. Laird, P. Rosenbloom, A gentle introduction to SOAR: an architecture for human cognition, 2006 update, https://web.eecs.umich.edu/~soar/sitemaker/docs/misc/GentleIntroduction-2006.pdf.

LeCun, Bengio and Hinton, (2015); Y. LeCun, Y. Bengio and G. Hinton, Deep Learning, In Nature, Vol 521, 2015, pp 437-444.

Lomb et al, (2013), Lomp O., Zibner S. K. U., Richter M., Schöner G. A software framework for cognition, embodiment, dynamics and autonomy in robotics: Cedar. In Mladenov V., Koprinkova-Hristova P., Palm G., Villa A.E.P, Apollini B., and Kasabov N. (eds), Artificial Neural Networks and Machine Learning, ICANN 2013, p. 475-482. Springer Berlin Heidelberg.

Marr D. (1982). Vision: A computational investigation into the human representation and processing of visual information. New York: Freeman.

McClelland (1981); J.L. McClelland, Retrieving General and Specific Information from Stored Knowledge of Specifics, Proceedings of the third annual conference of the Cognitive Science Society, pp. 170-172.

McClelland, J. L., & Rumelhart, D. E. (1981). An interactive activation model of context effects in letter perception: Part 1. An account of basic findings, in *Psychological Review*, 88, 375– 407.

McRuer D.T. And Jex H.R. (1967). A Review of quasi-linear pilot models. IEEE Transactions in Human Factors in Electronics, vol. 8. pp.231-249.

Muratov, Lakhman and Burtsev, (2014); S. Muratov, K. Lakhman and M. Burtsev, Neuro-evolution of sequential behaviour in multi-goal navigation tasks, ALIFE 14: Proceedings of the Fourteenth International Conference on the Synthesis and Simulation of Living Systems.

Musavi et al. (2017); N. Musavi, D. Onural, K. Gunes, and Y. Yildiz, Unmanned Aircraft Systems Airspace Integration: A Game Theoretical Framework for Concept Evaluations, in *Journal of Guidance, Control, and Dynamics*, Vol. 40, No. 1 (2017), pp. 96-109.

NATC, (2018); Naval Air Training Command, Basic Fighter Manoeuvring (BFM) Advanced NFO T-45C/VMTS, 2018, CNATRA P-826 (06-18), Corpus Christi, Texas, USA.

NENGO.ai, (2019); The NENGO Brain Maker is a Python package for building, testing, and deploying neural networks. https://www.NENGO.ai/.

NENGO.ai, Processes, (2019); Processes and how to use them. NENGO 3.2.0.dev0 docs. https://www.NENGO.ai/NENGO/ examples/advanced/processes.html.

Github.com/NENGO, (2022); GitHub - NENGO/NENGO: A Python library for creating and simulating large-scale brain models. https://github.com/NENGO/NENGO

NENGO.ai, Tutorials and Examples, (2018); NENGO PES learning rule and error signal dimensionality - Examples & Tutorials - NENGO forum. https://forum.NENGO.ai/t/NENGO-pes-learning-rule-and-error-signal-dimensionality/1112.

Newell, A., & Simon, H. (1963). GPS, a program that simulates human thought. In E. A. Feigenbaum & J. Feldman (Eds.), Computers and thought (pp. 279–293). New York: McGraw- Hill.

Olin-Ammentorp et al (2021); W. Olin-Ammentorp, K. Beckmann, C.D. Schuman, J.S. Plank, N.C. Cady. Stochasticity and Robustness in Spiking Neural Networks, in *Neurocomputing* 419(10), pp.23-36. DOI:10.1016/j.neucom.2020.07.105

Plunkett and Juola, (1999); K. Plunkett and P. Juola, A Connectionist Model of English Past Tense and Plural Morphology, in Cognitive Science Vol 23 (4) 1999, pp. 463–49.

Pollard J.J. (1975). All digital simulation for manned flight in turbulence. AFFDL-TR-75-82. USA Air Force Flight Dynamics Laboratory Report.

Rodin and Amin, (1989), Rodin E.Y., Amin M., Maneuver Prediction In Air Combat Via Artificial Neural Networks; Computers Math. Applic. Vol. 24, No. 3, pp. 95-112, 1992 (revised).

Rokach, (2016); L. Rokach, Decision forest: Twenty years of research, in Information Fusion, Vol: 27, pp.111-125. ISSN 1566-2535, https://doi.org/10.1016/j.inffus.2015.06.005.

Rokach & Maimon (2005); Rokach L., Maimon O., Decision Trees. 10.1007/0-387-25465-X_9.

Rumelhart et al., (1986), Rumelhart, D. E., & McClelland, J. L., & the PDP Research Group, Parallel distributed processing: Explorations in the microstructure of cognition. Volume I: foundations & volume II: Psychological and biological models. Cambridge, MA: MIT Press.

Salzberg, (1994); Salzberg, S.L. C4.5: Programs for Machine Learning by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc., 1993. In *Mach Learn* 16, 235–240 (1994). https://doi.org/10.1007/BF00993309

Sandamirskaya, (2014); Dynamic neural fields as a step toward cognitive neuromorphic architectures, in Frontiers in Neuroscience,

Sandamirskaya and Storck (2014); Y. Sandamirskaya and T. Storck, Learning to look and looking to remember: a neural-dynamic embodied model for generation of saccadic gaze shifts and memory formation. International Conference for Artificial Neural Networks (ICANN).

Sandamirskaya and Schöner (2010); Y. Sandamirskaya and G. Schöner, Serial order in an acting system: a multidimensional dynamic neural fields implementation, In IEEE Conference on Development and Learning, ICDL, 2010.

Shaw, (1985); R.L. Shaw, Fighter Combat: Tactics and Maneuvering, Naval Institute Press; 7th edition, ISBN 978-0870210594.

Stark et al. (2002); Stark, H., Woods, J. W., & Stark, H., Probability and random processes with applications to signal processing, p377.

Steging, (2018); C. Steging, Explainable AI: On the Reasoning of Symbolic and Connectionist Machine Learning Techniques, University of Gronigen, Department of AI.

Stewart and Eliasmith (2011); T.C. Stewart and C. Eliasmith, Neural Cognitive Modelling: A Biologically Constrained Spiking Neuron Model of the Tower of Hanoi Task, 33rd Annual Conference of the Cognitive Science Society.

Stewart (2012); T.C. Stewart; A Technical Overview of the Neural Engineering Framework, The Newsletter of the Society for the Study of Artificial Intelligence and Simulation of Behaviour, Issue 135 (Autumn, 2012).

Taatgen N., Anderson J.R. (2009). The Past, present and the future of cognitive architectures. in Topics in Cognitive Science. Vol 2009. pp.1-12.

Tukey, (1977); J.W. Tukey, Exploratory Data Analysis, Addison-Wesley, Reading, Mass.

Turing A.M. (1950). Computing machinery and intelligence. in Mind, vol 59. pp.433–460.

Virtanen, Raivio and Hamalainen (2004); K. Virtanen, T. Raivio, and R.P. Hamalainen; Modeling Pilot's Sequential Maneuvering Decisions by a Multisateg Influence Diagram, in *Journal Of Guidance, Control And Dynamics*, Vol. 27, No. 4, July–August 2004.

Wray R.E., Jones R.M. (2006). Considering SOAR as an agent architecture. In Sun R. (Ed.), Cognition and multiagent interaction: From cognitive modelling to social simulation. Cambridge University Press.

Yıldız, Agogino and Brat (2014); Y. Yıldız, A. Agogino and G. Brat, Predicting Pilot Behavior in MediumScale Scenarios Using Game Theory and Reinforcement Learning, in *Journal Of Guidance, Control And Dynamics* Vol. 37, No. 4, pp. 1335-1342.

Zhang et al. (2018); X. Zhang , G. Liu, C. Yang and J. Wu; Research on Air Combat Maneuver Decision-Making Method Based on Reinforcement Learning; in Electronics 2018, 7, 279; doi:10.3390/electronics7110279.

Zacharias et al. (1996); Zacharias, G. L., Miao, A. X., Illgen, C., Yara, J. M., & Siouris, G. M. (1996, December). SAMPLE: Situation awareness model for pilot in-the-loop evaluation. In Proceedings of the 1st Annual Conference on Situation Awareness in the Tactical Air Environment.

Zhou Zhi-Hua (2012). Ensemble Methods: Foundations and Algorithms. Chapman and Hall CRC, 978-1-498-3003-1.

Zorzi and Umilta, (1995); M. Zorzi and C. Umilta, A computational model of the Simon effect, in *Psychological Research* (1995) 58, pp. 193-205, Springer-Verlag.

APPENDIX 1

APPENDIX 1: TRANSCRIPTION OF AN ENGAGEMENT SCENARIO WITH A PILOT

As an example, the comments of a 4500 flight hours, fighter pilot at the age of 53, with true war experience in international operations is reported in below paragraphs. The comments are about the scenario 3, where this pilot found the model's decision as wrong. In the first paragraph, the pilot commented on the scenario and how he understands it. The second paragraph is his comment on t why it may be a false decision by the model.

1. "Let me first read it. Yes. It is a peace time, normal time, CAP mission, i.e. a patrol. Than you got a warning from RADAR, you redirected to that direction, there is a border intruder. During identification, you used ownship aircraft's sensors and tools or if you have datalink it is better. Already it is a border passage, then in normal conditions, with high percentage probability, if there are no operations etc., it need to be from out of our own forces. We deduce clearly it is not own force or friend. Hostility in the ID system does not always mean always hostile, the track need to show hostile behaviour to be addressed as hostile. i.e. his movement need to be hostile. For example, how does it pass the border? Its way of pass is important. It may be an arbitrary aircraft. For example, it may be an immigrant aircraft. It may not require always to shoot it down. So hostile aircraft is needed to be questioned. I look at the behaviour. We shall first understand if it has weapons. If it has weapons, this is important. But if no information; than behaviour I check. Does he fly from low altitude and is it fast? An unexpected attack is made this way. Why shall a hostile fly from airliner altitude and slow? Another behaviour is attack arm aircraft number? A solo aircraft does never come to a surprise attack. Non-manoeuvring aircraft is mostly not hostile. Entry speed gives me the type of the opponent. Transonic and supersonic speeds are to be followed. Potentially a fighter or bomber. While targeting and sorting, you first look at the speed and then altitude. Low altitude flight shows that the intruder has a capability of its low altitude flight. If it flies low, it means it has terrain awareness, additionally requires terrain clutter. Another method is to make a RADAR lock on to the opponent. If it tries to break it, then it means that it has the capability to detect a RADAR lock. So it is a military aircraft. And has potential to be truly hostile. If it locks to us, than

it is truly military, since we know that it has a fire control RADAR. Most hostile behaviour is to keep a RADAR lock on our aircraft. International rules or courtesy requires a time limit on RADAR lock. If it keeps the lock continuous (RWR warning runs long), then it is truly dangerous behaviour. Since long RADAR lock may excite the other aircrafts countermeasures, it is also a waste of resources. It happens especially in multinational missions in NATO bodies, since some identification problems may lead to countermeasures dispensing. In case of my own sensors and tools do not allow me to obtain the behaviour of the opponent, I mostly contact to the ground based RADAR stations or airborne C2s. Therefore, pilot has many inputs, RADAR locks, RWR warnings, if hostile ID is followed with long time hostile behaviour, then he decides on action, an action requires an order, either identification is followed by similarity in behaviour. i.e. if the opponent locked long, we lock long, if it manoeuvres, we manoeuvre. So his actions are either opposed equally or twice. If hostile behaviour is enduring, and ammo load is identified, then a radio warning is given before employing a missile. In radio, in international English language, a warning is given, the opponent is ordered to quit the national borders immediately and it declared clearly if he does not quit, he will be intercepted. If a hostile employed missile is observed in sensors, first action is avoidance and countermeasures dispensing. After a successful avoidance, counter employment is performed. After warnings, operations centre may always order to shot. When there is no hostile behaviour, then proper identification and contact is required to address correct behaviour."

2. "Yes, invalid decision is given. The opponent is hostile but unclear. So it needs to be identified. But there is an ambiguity. There is RADAR lock, so it may be needed valid regarding the duration of the lock. RWR may mean hostility. So my decision would be validity of the target. But it is not an easy case."