Using Learning from Demonstration to Enable Automated Flight Control Comparable with Experienced Human Pilots

Haitham Ahmed Omer Baomar

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy

University College London

Department of Computer Science

Supervised by Dr. Peter J. Bentley

2020

DECLARATION

I, Haitham Baomar, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

ABSTRACT

Modern autopilots fall under the domain of Control Theory which utilizes Proportional Integral Derivative (PID) controllers that can provide relatively simple autonomous control of an aircraft such as maintaining a certain trajectory. However, PID controllers cannot cope with uncertainties due to their non-adaptive nature. In addition, modern autopilots of airliners contributed to several air catastrophes due to their robustness issues. Therefore, the aviation industry is seeking solutions that would enhance safety. A potential solution to achieve this is to develop intelligent autopilots that can learn how to pilot aircraft in a manner comparable with experienced human pilots. This work proposes the Intelligent Autopilot System (IAS) which provides a comprehensive level of autonomy and intelligent control to the aviation industry. The IAS learns piloting skills by observing experienced teachers while they provide demonstrations in simulation. A robust Learning from Demonstration approach is proposed which uses human pilots to demonstrate the task to be learned in a flight simulator while training datasets are captured. The datasets are then used by Artificial Neural Networks (ANNs) to generate control models automatically. The control models imitate the skills of the experienced pilots when performing the different piloting tasks while handling flight uncertainties such as severe weather conditions and emergency situations. Experiments show that the IAS performs learned skills and tasks with high accuracy even after being presented with limited examples which are suitable for the proposed approach that relies on many single-hidden-layer ANNs instead of one or few large deep ANNs which produce a black-box that cannot be explained to the aviation regulators. The results demonstrate that the IAS is capable of imitating low-level sub-cognitive skills such as rapid and continuous stabilization attempts in stormy weather conditions, and high-level strategic skills such as the sequence of sub-tasks necessary to takeoff, land, and handle emergencies.

IMPACT STATEMENT

The Intelligent Autopilot System (IAS) is proposed, which is a fully autonomous autopilot powered by Artificial Neural Networks capable of providing unprecedented abilities to control and fly various types of aircraft. By using Artificial Neural Networks and the Learning from Demonstration concept, the IAS introduces the possibility to transfer human intelligence, intuitions, training, and skills that experienced pilots have to a system. The IAS requires few examples and small training datasets which speeds up development and enhances performance. The proposed approach eliminates the "black-box" issue and allows the possibility to undergo strict verification and validation for aviation certification purposes. The IAS has proved its ability to autonomously fly in simulation by executing complete flight cycles from takeoff to landing, and handling extreme conditions that can go beyond the capabilities of modern autopilots and experienced human pilots as well according to a recent evaluation by Oman Air. Introducing the IAS to the aerospace industry can enhance safety through its ability to provide intelligent control solutions that can handle emergency situations and extreme weather conditions, and lower costs by reducing the number of pilots in cockpits, or minimizing ground control costs associated with Unmanned Aircraft Systems for various applications including civilian and military. From academia to the aerospace industry, we have been collaborating to develop, train, and evaluate the IAS by working closely with experienced pilots of airliners to receive their inputs and training demonstrations for the IAS, and by signing agreements with some of the leaders of the aerospace industry to investigate the possibility of adopting the proposed approach of the IAS. Multiple research papers have been published describing the development journey of the IAS which in addition, has been highlighted in various international academic and industry-based conferences and events, and featured in multiple news articles in some of the most popular magazines around the world such as the Economist, BBC Focus, WIRED, Plane & Pilot Magazine, and Flight Safety Australia.

ACKNOWLEDGMENTS

I would like to thank my creator for giving me the strength and endurance to carry on this work. I would like to thank my family and friends for their support, and I would like to thank my brilliant supervisor Dr. Peter J. Bentley for providing me with exceptional guidance and support even before starting this endeavour. I would like to thank all the amazing people of the Department of Computer Science at my beloved university UCL for their help and support throughout this PhD.

I would also like to thank Captain Khalid Al Hashmi and Oman Air for providing me with their valuable time, facilities, and support which allowed for the fruitful collaboration mentioned in this work.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Flying an aircraft is a safety-critical job. It requires comprehensive education, training, and practice to prepare pilots for this career. This ranges from relatively short courses when intending to acquire a licence suitable for flying light airplanes powered by propeller-based engines, to long and extensive courses when intending to acquire a license suitable for flying large airliners powered by jet engines.

Pilots of passenger aircraft learn to perform piloting tasks that are required during the different phases of the flight, which comprise the flight cycle [1]. Performing a complete flight cycle starts with a ground-run on the runway to gain speed, rotate after a certain airspeed is achieved, climb, cruise while navigating between waypoints, descend, prepare for final approach while intercepting the landing runway path line, touchdown, flare, and lower airspeed before coming to a full stop [1]. Pilots are trained to perform landing under difficult weather conditions such as strong crosswind, and abort landing by executing a go-around if needed [1]. In addition, Human pilots learn to handle flight uncertainties or emergency situations such as severe weather conditions or system failure [1]. For example, pilots are exposed to scenarios of forced or emergency landing, where gliding is performed, which is the reliance on the aerodynamics of the aircraft to glide for a given distance while altitude is lost gradually when the aircraft has lost thrust due to full engine failure in relatively high altitudes [1].

For much of the time, human pilots are not in control of the aircraft. They perform takeoff manually, then engage the Automatic Flight Control System (AFCS/Autopilot) shortly after takeoff [2]. During final approach before landing, the human pilots disengage the AFCS to land manually unless visibility is too poor which requires an automatic landing via the Autoland feature of the AFCS [2]. Automatic Flight Control Systems or autopilots are used to control the trajectory of the aircraft by manipulating the control surfaces of the aircraft such as the elevators, ailerons, and rudder in three different axis which are pitch, roll, and yaw [2]. Autopilots are highly limited, capable of performing minimal piloting tasks in non-emergency conditions [2]. Autopilots are not capable of handling flight emergencies such as engine failure, fire, performing a Rejected Takeoff, or a forced (emergency) landing [2]. Although modern autopilots can maintain or hold a desired heading, speed, altitude, and even perform auto-land, they cannot handle complete flight cycles automatically, and they must be engaged and operated manually by the human pilots to constantly change and update the desired parameters, and they cannot handle severe weather conditions, such as strong crosswind components

combined with wind shear, gust, and turbulence [2]. Strong turbulence, for example, can cause the autopilot to disengage or even attempt an undesired action which could jeopardise flight safety [2]. The limitations of autopilots require constant monitoring of the system and the flight status by the flight crew, which could be stressful especially during long flights [3]. On the other hand, trying to anticipate everything that could go wrong with a flight, and incorporating that into the set of rules or control models "hardcoded" in an AFCS is infeasible [3]. There have been reports either discussing the limitations of current autopilots such as the inability to handle severe weather conditions, or blaming autopilots for several aviation catastrophes [3]. One such example was Air France flight AF447 on June 1st, 2009 where the aircraft entered a severe turbulence zone forcing it to climb steeply and stall. Shortly after that, the autopilot disengaged causing the aircraft to lose altitude dramatically. Unfortunately, it was too late for the flight crew to rectify the situation [4] [5]. Although the investigation concluded that the incident was a result of a series of events that was initiated by the autopilot disengaging itself after the aircraft's Pitot tubes froze, and placed the blame on the pilots' lack of situational awareness and their failure to initiate the appropriate procedure, Paul Salmon et al [6] argue that blaming the pilots is inappropriate, instead, the lack of designing flight control systems that are equipped with adequate situational awareness capabilities when a sensor fails for example is the main problem.

One solution to overcome the limitations of conventional AFCS is to introduce intelligent cockpit autonomy capabilities. This can be achieved by developing intelligent autopilots that use a combination of Apprenticeship Learning and Machine Learning to learn how to handle the different piloting tasks including dealing with uncertainties and emergencies.

Apprenticeship Learning or Learning from Demonstration is defined as the task of learning from an expert by observing demonstrations of the task to be learned given by the expert [7]. This type of knowledge acquisition depends generally on a real-time or hands-on practical transfer of knowledge and skills from expert trainers to trainees. An example is the use of simulators which are designed to simulate the real task a human operator is expected to experience. Learning from demonstrations can be effective in the sense that learners observe the actual performance of the task to be learned which significantly helps to capture the set of actions and rules required to perform the task successfully.

In a similar way, Machine Learning learns models from training data that allow a system to perform a given task autonomously. For instance, Artificial Neural Networks (ANNs) which

are loosely inspired by the neurons and synapsis in biological brains, can be used to generate models from training data [7]. Similar to how humans learn how to perform a given task by observing demonstrations by teachers, Machine Learning methods can use the demonstrations as training data containing inputs which represent information about the current state, and outputs which represent actions that can be taken given the state [8]. To achieve this, a mapping between each moment of the state (input), and the appropriate action (output) is generated and used as training data for the Machine Learning method [8]. After training, the system can be presented as a closed-loop system which continuously takes inputs, and produces the appropriate outputs over time [9]. The result can be viewed as a control system which is continuously performing a given control task (producing actions based on state information).

However, there are challenges and limitations when applying the concept of teaching a control system how to perform given tasks by utilizing the demonstrations of these given tasks (training data). For example, not all control problems involve basic linear actions [10]. In fact, some control problems require sets of complex tasks that could represent rapid and dynamic continuous actions such as trying to correct the path of a vehicle in difficult conditions, and other more general tasks that could be parts of a strategic goal such as the successful travel of the vehicle from point A to point B autonomously [10].

Therefore, to successfully tackle a given control problem using Machine Learning by learning from demonstrations, an appropriate learning method is required to ensure the generation of control models capable of handling the set of tasks required for the given control problem. In addition, the generated models should be able to accurately mimic the performance of the teacher (represented by the demonstrations) to ensure the generation of autonomous behaviour comparable with the behaviour of the human teacher.

This work aims to address the problem of robustness and limited capabilities that modern autopilots of large jets suffer from by proposing a novel intelligent autopilot which introduces the possibility to transfer the skills of experienced human pilots to a system. The proposed approach is to train the intelligent autopilot by demonstrations representing different piloting tasks provided by human teachers.

This work aims to prove:

Learning from Demonstration enables automated flight control comparable with experienced human pilots.

In this work, Learning from Demonstration means the use of aircraft-piloting demonstrations as training data. The demonstrations are provided by human teachers in a professional flight simulator. Initial research and development made use of demonstrations provided by the author of this work in order to avoid delays. The final stages of this work then made use of professional demonstrations provided by an experienced pilot who is Captain Khalid Al Hashmi, Senior Manager Crew Training at Oman Air. All the demonstrations were provided in the flight simulator X-Plane 10 Pro which is used by the aerospace industry due to its ability to provide professional simulation capabilities, and the possibility to integrate it with external systems[1]. The simulated aircraft models were Cirrus (light single-jet engine), Boeing B777, and Boeing B787 as it was intended to experiment using different models ranging from light aircraft to more complex and large multi-jet engine aircraft. In the training data, the inputs which represent the flight data, and the outputs which represent the piloting commands are mapped. The mapping can be handled by multiple Artificial Neural Networks (ANNs) which is a popular Machine Learning method applied to generate models representing the mapping between inputs and outputs [7]. By breaking the problem into smaller sub-problems, each piloting task can be handled by a dedicated ANN. The methodology for comparing the behaviour of the proposed intelligent autopilot with its human teacher was achieved by running multiple tests in the flight simulator to collect continuous flight data and autonomous control data representing the performance of the proposed intelligent autopilot, and comparing them with the performance of the human teacher to prove performance equivalency. The compared data include altitude, speed, climb and sink rates, and different angles such as pitch, roll, path-line, and glideslope. The statistical methods used to test the performance include Mean Absolute Error (MAE), Mean Absolute Deviation (MAD), and the Two One-Sided Test (TOST) for equivalency. The comparisons cover performing complete flight cycles that include ground-run, takeoff, climb, cruise, descend, and landing in the presence of varying weather conditions and emergency situations.

In order to provide evidence to support the hypothesis, this work is subdivided into the following objectives:

1. Proving the possibility to teach a flight control system how to perform basic flights.
2. Teaching the system how to handle emergency situations.
3. Teaching the system how to perform complete flights including navigation and landing.

[1] X-Plane for Professional Use: https://www.x-plane.com/pro/ [accessed 2018]

4. Teaching the system how to handle severe weather landing.
5. Teaching the system how to behave like experienced human pilots of airliners by learning from professional demonstrations provided by an experienced pilot unlike the demonstrations used for objectives 1 to 4 which were provided by the author of this work.

Since it is intended to prove the possibility of teaching Artificial Neural Networks (ANNs) how to behave like experienced human pilots, the ANNs should acquire the ability to perform continuous and dynamic tasks such as correcting the path of the aircraft especially in difficult conditions such as during final approach when the aircraft's speed is relatively slow which hinders manoeuvrability, and in the presence of severe weather conditions. In addition, the ANNs should acquire the ability to perform sets of tasks aimed towards achieving a strategic goal such as flying from airport A to airport B, or dealing with an emergency such as engine fire. Therefore, the reason for choosing the above objectives is the comprehensive scope they cover including continuous and dynamic tasks, and strategic sets of tasks which are essential to acquire the overall ability of flying airplanes.

This work however, does not consider related challenges such as sensory faults or deprivation which can affect the performance of such autonomous systems. In addition, faults of flight control surfaces during flights are not considered.

This thesis is structured as follows: chapter 2 is a literature review which covers the control problem, related work on utilizing Learning from Demonstration in autonomous flying, and handling autonomous flights including navigation and landing under uncertainties. Chapter 3 explains the proposed Intelligent Autopilot System (IAS), the methodologies applied for designing and testing the IAS, and the first prototype of the IAS designed to prove the possibility of transferring human pilots' skills and abilities to a flight control system, to enable it to perform basic flights autonomously. Chapter 4 explains the second prototype of the IAS designed with the objective to learn how to handle emergency situations. Chapter 5 explains the third prototype of the IAS designed with the objective to learn how to perform complete flights including navigation and landing. Chapter 6 explains the fourth prototype of the IAS designed with the objective to learn how to handle severe weather landing. Chapter 7 explains the fifth prototype of the IAS designed to behave like experienced human pilots of airliners. The five chapters which explain the prototypes include experiments, results that compare the behaviour of the human pilot with the behaviour of the IAS when applicable, as well as

comparisons between the prototypes of the IAS, and analysis. Chapter 8 provides conclusions, and chapter 9 discusses future work. The appendix of this thesis contains copies of five research papers which were submitted and published to relevant conferences and journals.

2. LITERATURE REVIEW

In this chapter, review and analysis are provided of the state-of-the-art literature of control automation in aerospace with a focus on related non-linear and robust methods, mainly Artificial Neural Networks, transfer of skills and abilities to machines, and autonomous flying in uncertain conditions.

2.1 Controllers

2.1.1 Non-Adaptive Linear Controllers

Current operational autopilots fall under the domain of Control Theory [9]. Classic and modern autopilots rely on controllers such as the Proportional Integral Derivative (PID) controller, and Finite-State automation [10]. Non-adaptive linear controllers such as Proportional Integral Differential (PID) controllers are extensively used in the aerospace domain for manned and unmanned aircrafts ranging from fixed-wing airliners to Micro Aerial Vehicles (MAV) [10]. PID controllers are used to provide adequate trajectory-tracking applied to control the aircraft's surface controllers such as rudders, ailerons, and elevators by sending control commands to their actuators [10]. PID controllers have a pre-designed gain factor or control law applied continuously to transform the vehicle from the current state to the desired state by calculating the gain which should be applied based on the proportional error, the integral error, and the differential error [10]. PID controllers are capable of providing relatively simple autonomous control of an aircraft such as maintaining a certain trajectory by controlling speed, pitch, and roll [10]. However, PID controllers cannot cope with uncertainties due to their non-adaptive nature [10]. Another limitation of PID controllers is the extensive tuning requirement of the controllers where gains are tuned to specific aerial platforms, which makes it difficult to transfer the tuned controllers to different aerial platforms [11].

2.1.2 Adaptive Non-Linear Controllers

Due to the current need to develop intelligent flight controllers for fixed-wing aircrafts that are capable of handling flight uncertainties, the classic control theory field is being extended by adding intelligent components capable of enhancing and aiding conventional controllers used by modern aircrafts [12]. Unlike non-adaptive conventional controllers which are designed based on linearized aircraft models, adaptive nonlinear controllers are expected to be the next generation of flight controllers due to their ability to handle uncertainties affecting gain performance which is one of the limitations of current conventional flight controllers [13].

To have further autonomy under normal conditions such as executing a flight path by an autopilot, a Simultaneous Localization and Mapping (SLAM) algorithm is coupled with the PID controllers [14]. In large scale aircraft, such as fixed-wing airliners, the flight control computers use several on-board sensor data such as speed and altitude, and satellite data such as Global Positioning System (GPS) data to determine localization of the aircraft and path mapping [14]. Recent research effort has been focusing on investigating and introducing different SLAM methods to achieve better locomotion. For example, Machine Learning methods such as computer vision algorithms are used to achieve autonomous navigation capabilities for small scale Unmanned Aerial Vehicles (UAVs) where GPS signals are weak or unavailable [14]. Such scenarios can be observed with indoors MAVs which apply a different set of sensors such as stereo vision, laser, infrared, or ultrasonic [15] [16] to first, generate a 2-D or 3-D map of the environment, and then, determine proportional distance values which are processed and sent to the PID controllers as current state inputs that are used to generate gain outputs [14].

Green et al [17] presented an autonomous navigation system based on a depth perception optic flow microsensor capable of Autonomous Take-off and Landing (ATOL). While auto take-off was achieved by simple increasing throttle to full power and deflecting elevators, auto landing was achieved by the following: 1. a microcontroller takes an initial reading from the optic flow sensor, 2. the reading is considered as the desired value (distance to landing pad), and 3. the controllers adjust throttles (gain) based on continuous readings of distance (error between current and desired positions) in a fashion similar to how PID controllers operate [17].

Bills et al [18] proposed a Machine Learning based technique that does not require 2-D or 3-D map generation, but rather relies on classification and computer vision algorithms for indoors MAVs. First, an on-board camera takes images of the environment ahead, then, Canny edge detector algorithm is applied to detect edges, and the probabilistic Hough transform algorithm is used to detect long lines [18]. The results are used to classify the environment as a corridor or a staircase, and depth images generated from the camera are analysed to detect obstacles (pixel's level of grey) [18]. Finally, the generated data is processed and sent as inputs to PD microcontrollers to control the MAV and navigate through the corridor or the staircase [18]. This technique which focuses on utilizing Machine Learning based computer vision

TABLE 2.1
RESEARCH PAPERS ADDRESSING MACHINE LEARNING BASED COMPUTER VISION ALGORITHMS TO ACHIEVE AUTONOMOUS COLLISION FREE NAVIGATION

Description	Reference
Fly-inspired visual sensors for autonomous micro-flyers	[19]
A 2-D vision system that sends velocity commands to a low-level controller	[20]
A stereo-vision system that remaps the world onto a cylindrical co-ordinate system which simplifies range computations and collision avoidance	[21]
Using an optic flow sensor for velocity and position estimation for indoors UAVs	[22]
A visual odometer that estimates the vehicle's ego-motion by extracting and tracking visual features, using an on-board camera	[23]
Estimating the altitude of a UAV from top-down aerial images taken from a single on-board camera	[24]
Mapping the image errors onto the actuator space via a depth-independent interaction matrix	[25]
A vision system and imitation learning techniques to train a MAV controller how to avoid obstacles	[26]
UAV Landing Using Computer Vision Techniques for Human Detection	[27]
A Computer Vision Line-Tracking Algorithm for Automatic UAV Photovoltaic Plants Monitoring Applications	[28]
Swaying displacement measurement for structural monitoring using computer vision and an unmanned aerial vehicle	[29]
Visual Object Detection For Autonomous UAV Cinematography	[30]
Real-Time SLAM Based on Image Stitching for Autonomous Navigation of UAVs in GNSS-Denied Regions	[31]
UAV path automation using visual waypoints acquired from the ground	[32]
Learning Pose Estimation for UAV Autonomous Navigation and Landing Using Visual-Inertial Sensor Data	[33]

algorithms to achieve autonomous collision free navigation have investigated in many research papers as Table 2.1 shows.

Other techniques aimed at enhancing conventional controllers were investigated such as the introduction of Genetic Algorithms as a PID gain tuning approach [34]. First, PID controllers responsible for controlling the three axis of flight (pitch, Yaw, and Roll) were tuned using Roulette selection function and a two-point crossover properties of Genetic Algorithm in Matlab [34]. Then, the optimized gain values are sent to the controllers as enhanced gain values compared with the gain values generated from the PID controllers alone [34].

In [35], an autopilot design that utilizes a combination of indirect adaptive controller with approximate feedback linearisation is proposed to tackle the challenging problem of providing fast response and minimum tracking error for missiles. The introduction of adaptation enhanced closed-loop control robustness, and approximate feedback linearisation tackled the issue of unstable zero dynamics [35]. The proposed approach can be used for nonlinear missiles with uncertain parameters, which is supported by the desirable results in simulation [35].

In [36], an active disturbance rejection control (ADRC) strategy based on fuzzy control is proposed, which is designed to improve the ability of anti-interference, meanwhile, fuzzy control is adopted to adjust the ADRC parameters online, which makes control performance better. Simulation results show that compared with conventional PID the Fuzzy-ADRC strategy can suppress the disturbances quickly and efficiently, with higher control accuracy, stronger robustness and so on [36]. In [37], a fuzzy self-tuning PID (FSPID) controller to tackle the disadvantages of conventional PID controllers in aircraft autopilots is proposed where fuzzy self-tuning PID tunes the PID parameters to achieve the optimal performance, which based on the results in simulation, the proposed controller can adaptively improve the system response by on-line setting of PID parameters. Other methods were used to enhance the pitch control performance such as Linear Quadratic Regulator (LQR) [38], and Fuzzy Logic Controllers (FLC) [39].

A large amount of research effort aimed towards the development of such intelligent flight controllers adopted Artificial Neural Networks (ANNs) as the method of choice [7]. In this context, ANNs have been applied to enhance conventional controllers by adding adaptation capacities in three ways: 1. Aiding autopilot adaptive controllers, 2. Fault tolerance, and 3. Prediction of aerodynamic. The efforts have been focusing on enhancing stability and navigation [7].

2.1.2.1 Artificial Neural Networks

Artificial Neural Networks (ANNs) are popular learning methods due to their ability to handle highly dynamic real-time large volumes of data [7]. They are a highly-interconnected system capable of processing data through their dynamic state response to external inputs [7]. Although Artificial Neural Networks are sometimes referred to as slow learners, as soon as the learning model is generated, ANNs are very fast classification and regression techniques that are suitable for applications running in dynamic and high-speed environments [40] such as electrical circuits management and analysis [41], and high frequency trading [42]. ANNs are

also used in robotics applications due to their capability of handling large amounts of real-time noisy sensor data [43] [44]. The latter resemble the Intelligent Autopilot System (IAS) which should be able to receive real-time flight status data from multiple sensors, process the data, and apply the appropriate control commands actions given the current flight state.

2.1.2.2 Artificial Neural Networks -Based Adaptive Non-Linear Controllers

Pashilkar et al [45] presented a neural-aided controller which does not replace the conventional controllers used by the aircraft's autopilot, but aids them by enhancing the fault tolerant measures during windy landings. In such scenarios, the possibility to encounter a fault represented by a jammed surface controller is increased [45]. For example, an elevator, or an aileron, or both could get stuck at a certain deflection due to strong winds [45]. The neural-aided controller applies an error feedback learning approach, and a Dynamic Radial Basis Function neural network which uses online learning to enhance the outputs sent to the surface controllers by the conventional controller which is not able to cope with fault [45]. The neural-aided controller outputs act as weights which increase or decrease the values of the conventional controllers' output values to cope with the deflection caused by the jammed surface controller, and ensure a safe landing [45].

Under NASA's Intelligent Flight Control System (IFCS) program, Soares et al [46] proposed an adaptive intelligent controller called G-2 which is based on neural networks. G-2 aids the aircraft's conventional controllers by compensating for errors caused by faults [46]. The faults are represented by surface controller failures, or dynamic failures caused by modelling errors [46]. Neural networks were applied to output command augmentation signals to reimburse for the errors caused by faults [46]. Conventional controllers such as the widely used PID (Proportional Integral Differential) controllers suffer from a limitation represented by the inability to change the gains of the system -also known as proportionality constants- after the system starts operating [46]. At this point, only one solution can be applied to tackle this problem, which is to stop the system, perform gain re-tuning, and restart the system, which might not always be practical [46]. Therefore, neural networks were introduced to eliminate this problem, and to replace the fixed gains by actually learning what the gains should be, and incorporating them with the actual output of the conventional controller [46].

Artificial Neural Networks have been applied as nonlinear dynamic models to aid the controllers of small scale unmanned Aerial Vehicles (UAV) as well. For example, neural network controllers were introduced to aid the trajectories controllers of a hexacopter [47], and

a quadrotor drone [48]. The main goal is to reduce the inverse errors, and ensure a more stable flight [47] [48]. The results show the ability of neural-based controllers to enhance the robustness of conventional controllers of drones in a fashion similar to the enhancements achieved for fixed-wing aircrafts [47] [48]. Other than stability, ANNs have been applied to small scale UAVs to enhance position tracking controllers using integral of the Signum of the error (RISE) feedback along with ANNs, which proved to be an effective approach based on accuracy results [49].

The possibility to enhance the stability of unstable UAVs using ANNs has been investigated in [50]. Here, the concept of aiding conventional controllers using ANNs has been applied to maintain a steady pitch rate via ANNs with linear filters and backpropagation to approximate the control law [50]. Offline-training coupled with finite time training was applied to enhance stability, while online-training was applied to compensate for flight uncertainties caused by aerodynamic and surface control problems [50]. Results showed the ability of ANNs to enhance the capabilities of conventional controllers to maintain stability for unstable UAVs [50].

The instability problem found in many aerial vehicles can be clearly seen in helicopters due to their extremely non-linear aerodynamic characteristics [51]. For such platforms, a different set of control theory-based controllers are usually used to overcome the aerodynamic issues such as Direct Inverse Controllers (DIC) [51]. To enhance the performance of the DIC, an ANN was applied to investigate the possibility to adapt to variations in the dynamic of the system [51]. The ANN was capable of learning the dynamics of a drone helicopter performing a hover-hold [51]. The main advantage was the ANNs ability to cancel out unnecessary dynamics data distorting the hover-hold control command as per the results [51].

A different unstable platform can be observed in experimental Hypersonic Flight Vehicles (HFV) [52]. ANNs have been applied to the problem of trajectory control and navigation for HFVs as well. Xu et al [52] proposed neuro-controllers to tackle the HFVs problems of motion equations high complexity, and the lack of knowledge of the aerodynamic parameters of HFVs. The neuro-controllers were designed to mimic nonlinear adaptive and dynamic controllers via the back-steeping technique capable of controlling the aircraft's altitude and velocity from an initial flight state to a desired trim. The results showed the proposed approach's ability to provide better control for HFVs [52].

Fighter jets are also known as unstable aerial platforms [53]. Savran et al [53] proposed yet another adaptive and dynamic nonlinear controllers based on ANNs to aid the conventional controllers operating on-board F-16 fighter jets. The proposed control system was designed to handle the compensation issue of uncertainties caused by system failures [53]. The work was done by: 1. modelling the dynamic behaviour of the nonlinear model of the F-16 fighter jet, and 2. designing ANN based PID controllers capable of enhancing the system's gain attempts through online-training and backpropagation [53]. Results showed the proposed systems ability to yield good results for fault tolerance [53].

In [54], an interesting approach is proposed to develop a control technique for an unusual aerial fuselage which is a spherical UAV. To tackle this challenging problem, a nonlinear control theory combined with Adaptive Neural-Networks Disturbance Observer (NN-DOB) is proposed to control the attitude and altitude of the UAV in the presence of model uncertainties and external disturbance [54]. Experiments in simulation showed that the proposed approach estimated the uncertainties effectively without prior knowledge [54].

The research effort aimed towards enhancing navigation using ANNs have gained interest in the missile navigation context. Rajagopalan et al [55] proposed the substitution of conventional Proportional Navigation (PN) guidance controllers with ANN based controllers due to PN controllers' limitations hovering mainly around the need for faster and more accurate calculations and approximations [55]. Results showed a faster throughput is possible when using ANN controllers instead of PN controllers due to the ability of ANNs to process data in parallel [55].

Using Artificial Neural Networks as intelligent controllers was expanded in a very interesting research conducted by DeMarse et al [56]. In their work, a living neuronal network extracted from a rat's brain, and cultured on a microelectrode array was trained to provide simple aid to a simulated aircraft's conventional controllers [56]. The living network was used as a matrix of weights that has the ability to measure and manipulate feedback control to stabilize a simulated aircraft [56]. The errors generated by the gains of the aircraft's controllers were compensated by measuring the proportional feedback using the calculated synaptic weights between neurons within the rat cortical network [56].

The same concept of enhancing the capabilities of conventional controllers by adding adaptive capabilities to handle nonlinearities using Artificial Neural Networks has been investigated and applied in various research papers as Table 2.2 shows.

2.1.2.3 Multiple Controllers & Sensors

The break-down approach of problems into smaller sub-components was applied early in the work of Rodney Brooks [71] where a multi-layered hierarchy representing a decomposition of a complex behaviour into sub-behaviours is proposed. Each sub-behaviour represents an agent responsible for a particular behaviour competence [71]. For example, the lowest or the zeroth layer of a robot could be dedicated for the purpose of avoiding obstacles [71]. After that, the first layer could be dedicated for the purpose of wandering around the environment in which the robot is located [71]. Then, the second layer could be dedicated for the purpose of exploring

TABLE 2.2
RESEARCH PAPERS ADDRESSING ENHANCED CONTROLLERS BY ADDING ADAPTIVE CAPABILITIES TO HANDLE NONLINEARITIES USING ARTIFICIAL NEURAL NETWORKS

Description	Reference
Verification and validation techniques for non-adaptive ANN based controllers	[57]
Neural Network Identification for modelling the nonlinear dynamics of a miniature helicopter	[58]
Incorporating a neural network with real-time learning capability in a flight control architecture	[59]
An adaptive controller design method based on neural network for reconfigurable flight control systems in the presence of variations in aerodynamic coefficients	[60]
UAV parameter estimation using Iterative Bi-Section Shooting, Artificial Neural Network, and "Hybrid ANN_IBSS"	[61]
Adaptive UAV autopilot design using two-stage dynamic inversion, and feedback dynamic inversions based on a command augmentation system	[62]
Single Neural Adaptive PID Control for Small UAV Micro-Turbojet Engine	[63]
Neural-Networks Control for Hover to High-Speed-Level-Flight Transition of Ducted Fan UAV With Provable Stability	[64]
Universal Adaptive Neural Network Predictive Algorithm for Remotely Piloted Unmanned Combat Aerial Vehicle in Wireless Sensor Network	[65]
Adaptive Neural Motion Control of a Quadrotor UAV	[66]
Adaptive Neural Fault-Tolerant Control for the Yaw Control of UAV Helicopters with Input Saturation and Full-State Constraints	[67]
Neural Network-based Adaptive Backstepping Controller for UAV Quadrotor system	[68]
Adaptive tracking control of an unmanned aerial system based on a dynamic neural-fuzzy disturbance estimator	[69]
Neural PD Controller for an Unmanned Aerial Vehicle Trained with Extended Kalman Filter	[70]

the overall environment [71]. All these layers of sub-behaviours work together simultaneously to achieve the strategic goal which could be the successful travel from the starting point to the end without colliding with obstacles [71]. The layers receive inputs as data provided by the sensors of the robot, then, work in parallel to generate output data used as control command for the actuators [71].

The problem of coordinating multiple sensor-motor architectures found in complex robotic systems is challenging due to the simultaneous and dynamic operation of these motors while insuring rapid and adaptive behaviour, and due to the need to properly handle the fusion of data from disparate sources [72]. In nature, animals manage this problem by the large number of neural circuits in the animals' brains [72]. For example, neural circuits which are responsible for motion are connected to the muscles (motor systems), and operate simultaneously and dynamically while handling changes in the environment [72]. This has inspired the field of complex robotics to develop multiple neural-based controllers and integrate them together to tackle larger problems such as long-endurance locomotion under uncertainties [72]. In [72], the problem of coordinating multiple sensor-motor architectures is addressed in the context of walking by developing a neural circuit which generates multiple gaits adaptively, and coordinates the process of walking with different behavioural-based processes in a hexapod robot. The results showed the ability of biology-inspired system to detect and stabilize multiple instability scenarios, and to determine what needs to be controlled at each moment which allows the system to handle changes in the environment [72].

Multiple Artificial Neural Networks were applied to the problem of detecting roads visually. In [73], different inputs are fed into multiple ANNs to handle multiple segments of the image. The proposed approach allows the system to detect and classify multiple factors of the environment ahead which leads to an enhanced performance compared with other computer-vision solutions [73].

In [74], Multiple ANNs were applied to tackle the limitations problem of traffic light control systems that are based on conventional mathematical methods. In simulation, the results showed that the approach of using multiple ANNs to address this problem presented an improvement in performance compared with other methods [74].

Another proposed system inspired by biology is presented in [75], which is designed to handle the challenging problem of gesture recognition. The system shares similarities with the human visual system by developing multiple spiking ANNs [75]. The outputs of the spiking

ANNs are used to generate a fusion of multiple data from different segments of the gesture [75]. The results proved the system's ability to handle dynamic visual recognition with the presence of complex backgrounds [75].

The approach of segmenting or breaking down the problem, and using multiple ANNs to handle the generated multiple segment showed the potential to enhance the properties of ANNs as explained in [76]. A large ANN is split into parallel circuits that resemble the circuits of the human retina [76]. During training, the Backpropagation algorithm runs in each circuit separately. This approach does not only decrease training time, but it also enhances generalization [76].

Having multiple sensors, control surfaces, low-level control components, high-level control, etc., require an approach that can take all these different components and tasks into consideration, and provide an architecture that can utilise and operate all these components in harmony to achieve the overall goal. In [77], an open-source rational agent architecture is proposed, which aims to provide autonomous decision-making and reconfigurability to suit different platforms. The proposed approach provides a hybrid architecture to separate the high-level autonomous decision-making from continuous or low-level control components, which makes decision strategy verifiable, and allows a modular approach to the architecture [77]. This approach introduces a high level of flexibility where different components can be added, removed, or replaced to suit the different requirements [77].

2.1.3 Analysis of the Controllers Literature

Analysing the literature and the state-of-the-art of both non-adaptive linear controllers and adaptive non-linear controllers shows the superiority of the latter when handling the non-linear conditions present in the environment and the different models of multiple flying platforms. Non-linear adaptive controllers are also capable of handling uncertainties such as adapting to a surface controller fault [14-17], [34].

Introducing Machine Learning based approaches, aided controllers by adding an extra level of adaptivity to handle more complex nonlinearities present in highly unstable aerial platforms such as fighter jets and Hypersonic Flight Vehicles (HFV) [52]. Artificial Neural Networks (ANNs) seem to be a very popular choice due to their ability to handle highly dynamic real-time large volumes of data [41], [42].

Literature suggests that conventional controllers such as Proportional Integral Derivative (PID) controllers even when enhanced are not capable of learning a high-level task comprising of a sequence of multiple sub-tasks [10]. For example, learning a full take-off task which is comprised of a sequence of sub-tasks (release brakes, increase throttle and set to maximum power, wait until a certain speed is achieved then deflect elevators, wait until a certain altitude is achieved then retract gear, etc.) is beyond the capacities of these conventional controllers, which means, performing a fully autonomous flight by relying just on conventional controllers is not feasible [10]. On the other hand, manually designing and developing all the necessary controllers to handle the complete spectrum of flight scenarios and uncertainties ranging from normal to emergency situations might not be the ideal method due to feasibility limitations such as the difficulty in covering all possible eventualities [10].

Replacing PID controllers with Artificial Neural Networks, or enhancing these controllers by adding an ANN layer, proved to tackle the robustness issue of PID controllers [46-70]. In addition, breaking down the problem by using multiple ANNs instead of one large ANN delivered superior results [71] [72]. The latter approach which is inspired by biological neural circuits in the brain, each dedicated to a certain task, such as controlling a specific part of the biological body, proved to be an excellent approach for problems that require a fusion of sensory data, and the control and manipulation of multiple components, such as the problem of autonomous flying [71] [72]. By following this approach, the results showed the ability of such multiple ANNs-based systems to provide higher levels of accuracy since each ANN is carefully designed and trained to handle a specific segment, rather than the whole problem [73] [74].

2.2 Learning from Demonstration

Learning a model represented by a mapping between states in an environment and actions or behaviours is one of the main goals in the field of robotic systems. Learning this mapping or policy gives robotic systems the ability to perform actions or behaviours autonomously given the current state of the environment. Developing a policy manually is usually a challenging task especially if there are more than one policy required to allow the system to perform a set of tasks without human intervention [78]. Therefore, Machine Learning approaches have been investigated and applied to aid the process of policy learning [78]. An interesting approach known as Learning from Demonstration (LFD) has been applied where robotic systems learn policies that govern successful performance of tasks by first, observing a human expert while

performing the tasks to be learned, and then, performing the same tasks autonomously after generating learning models [78]. The demonstrations can be defined as sequences of pairs or mappings between the environment's states and actions or behaviours that can be logged as training datasets [78]. After that, Machine Learning algorithms can be applied to generate behaviour models or policies from the demonstration datasets [78].

Learning from Demonstration has been applied by following different Machine Learning approaches. Authors gave LFD different names based on the followed Machine Learning approach such as Behavioural Cloning [79], and Apprenticeship Learning [8]. Learning by Imitation is split into two main parts each with its own objectives: 1. learning a policy or a low-level task which could represent a direct mapping between states and relative actions, and 2. learning a reward function or a high-level task which could represent a specific goal to be achieved [8] [79].

In the aviation context, Iiguni et al [80] presented a Learning from Demonstration approach for autonomous flight control that relies on Nearest Neighbours (kNN) or (mNN). This approach uses a k-d tree structure method to search for the current state of the environment in a priori to locate the previously recorded nearest neighbours to the current state [80]. The nearest neighbour to the current state can then be used to estimate the appropriate actions recommended for the current state, and build a Linear Local Model (LLM) [80]. This step is constantly repeated by discarding the generated model and forming a new one based on the change of states [80]. More recently, Matsumoto et al [81] proposed a similar learning approach that depends on Learning from Demonstration (LFD) to capture the human pilot's skills and apply them in an autonomous Unmanned Aerial System (UAS) to achieve the same level of safety observed in civil aviation [81].

2.2.1 Behavioral Cloning

Sammut et al [82] presented an early attempt to develop an autopilot that can learn by imitation. A Decision Tree learning algorithm was used to capture the set of rules or high-level tasks required to fly an aircraft in a flight simulator [82]. The rules were transformed into a collection of If-Statements that govern the control commands sent by the autopilot [82]. The autopilot was able to perform a sequence of piloting tasks as follows: take-off and climb to a certain altitude, level out the aircraft and cruise for a certain distance, perform a series of waypoint checks, descend and land [82]. This set of tasks was performed by three human pilots in a flight simulator [82]. While the human pilots demonstrated the tasks, the states of the flying

environment were coupled with events or actions performed by the demonstrators, and logged. The log contained about 90,000 events comprised of 1,000 events for each pilot flying 30 times [82]. The reason behind collecting a large number of events from each demonstrator was to present the learning algorithm with sufficient data to deduce the rules to be learned [82]. Another reason was due to the method followed to collect data, which made it difficult for the learning algorithm to deduce the rules [82]. For example, during demonstrations, the human pilots followed a visual approach to navigate from one checkpoint to another, where a certain landmark for instance was used to indicate that the current checkpoint was met, and it was time to follow a different heading to navigate to the next checkpoint [82]. In addition to that, and to avoid having to gather more flight examples, a data analysis process using C4.5 had to be done to assist the learning algorithm to deduce the rules, especially that each pilot or demonstrator flow the aircraft differently, which introduced variations in the collected logs or datasets [82]. Even though C4.5 was never developed to learn reactive strategies, the authors favoured it because they were familiar with it, and because of its reliability [82]. Another limitation of the C4.5 faced by the authors, was the need to have discrete values as class labels instead of the aircraft's surface controllers (ailerons, elevators) continuous values [82]. This limitation called for an additional processing step which broke down the action settings into sub-ranges that can be labelled using discrete values [82]. After generating the control or piloting rules from the decision trees, they were transformed into blocks of If-Statements written in the programming language C [82]. The If-Statements continuously checked for certain conditions, and applied appropriate actions [82]. For example, just before take-off, an If-Statement checked in a closed loop fashion if the speed exceeded a certain value before deflecting the elevators in order for the aircraft to leave the ground [82].

In a different research paper, Sammut [83] elaborated on the main challenge faced while working on the autopilot [83] mentioned above. The main challenge was the inability of the Decision Tree based induction program to deduce complex relationships present in the highly dynamic and noisy continuous data [83]. This highly dynamic data contain complex relationships between different states and actions which represent the sub-cognitive or low-level and rapid actions performed by skilled human demonstrators given rapid changes in the state or the environment [83].

Behavioural Cloning has been applied in other contexts as well. Sheh et al [84] applied Behavioural Cloning to mimic a traverse obstacle avoidance task performed by an expert.

Through Decision Tree learning, their model captures the decision-making process performed after observing high dimensional sensory inputs such as Depth images [84].

2.2.2 Apprenticeship Learning

Recent research effort proposed a strategy aimed towards capturing and learning human user models through the concept of Apprenticeship Learning using Inverse Reinforcement Learning, either by considering a Markov decision process proposed by Pieter Abbeel and Andrew Y. Ng [8], or by considering Gradient methods proposed by Neu et al [85]. These methods in general do not depend on receiving a Reward Function in advance, which is how classic Reinforcement Learning works, instead, the proposed approach attempts to find a reward function by observing how a human expert demonstrates the task to be learned by the system [8] [85]. Apprenticeship Learning eliminates the need to explicitly present the rewards, policies, and strategies manually, which could be difficult or unfeasible tasks [8] [85]. This has opened the window to apply Apprenticeship Learning to a wide range of contexts ranging from navigation, to video games, and music [8] [85]. Asta et al [86] proposed an interesting approach for vehicle routing using a Hyper-heuristic method. Their approach follows a Supervised Learning method, which generates classifiers that capture various actions an expert performs while the search process is taking place [86]. After that, the classifiers are used to generate a Hyper-heuristic learning model, which is potentially capable of generalizing the actions of the expert hyper-heuristic while solving unseen instances [86]. Messer et al [87] also applied Apprenticeship Learning through Inverse Reinforcement Learning to discover a reward function in a musical context. While experts generated melodies, their behaviour was used by the learning agent to discover a reward function [87]. The reward function was used to generate new melodies [87].

Abbeel et al [88] applied Apprenticeship learning to a difficult dynamic control system to teach controllers of a helicopter how to perform acrobatic manoeuvres autonomously. Applying Apprenticeship Learning proved to be an efficient learning technique to capture the expert demonstrator's skills [88]. To learn how to perform a specific manoeuvre, a multiple number of demonstrations by an expert were captured [88]. The goal was to consider observations as noisy attempts from the expert to perform the desired manoeuvre successfully [88]. An Expectation–maximization algorithm that implements a modified Kalman Smoother along with a Dynamic Programming algorithm to deduce the desired trajectory model of the manoeuvre were applied [88]. The main reported challenge was the difficulty to capture high-

level dynamic models present in complex manoeuvres which consist of multiple reward functions or sub-tasks rather than just one [88].

To tackle the Inverse Reinforcement Learning problem of assuming that the demonstrator is trying to maximize a single reward function, Michini et al [89] proposed a different approach where the demonstrator is assumed to be trying to maximize a set or sequence of sub-tasks representing multiple reward functions rather than a single one [89]. A Bayesian nonparametric reward-learning framework was proposed which infers multiple reward functions or goals chained together from a single demonstration containing all of the goals [89]. The proposed approach handles both continuous and discrete values by applying Gaussian process reward representations [89]. This approach presented results showing the advantage of assuming that the demonstrator is trying to maximize multiple reward functions rather than just one [89].

2.2.3 Analysis of the Learning from Demonstration Literature

Analysing the literature and the state-of-the-art of Learning from Demonstration (LFD) leads to the conclusion that there are two main schools of thought. One school tends to use the term Behavioural Cloning [82] to describe learning by imitation or LFD. This school focuses one using supervised learning methods, mainly Decision Tree/Random Forest learning to capture the human expert's model of high-level/strategic tasks [82-84]. The trees model the rules and thresholds that are constantly checked to control the transfer from one action to another within a sequence. Sammut et al [82] were able to capture the general rules of take-off, climb, cruise, and landing, but reported challenges when trying to capture dynamic and real-time continuous actions. This school in general reported the same challenges and difficulties when trying to capture the human expert's model of low-level tasks or actions that represent the rapid and continuous sub-cognitive actions performed by a human expert in a highly dynamic environment [82].

On the other hand, the other school of thought uses the term Apprenticeship Learning [8] to describe learning by imitation. This school focuses on using a modified form of Reinforcement Learning called Inverse Reinforcement Learning [85-88]. While the concept of Reinforcement Learning allows a given system to explore an environment and exploit the available knowledge to generate policies and maximize reward functions, Inverse Reinforcement Learning allows the system to observe an expert who is believed to be trying to maximize a reward function. Unlike Reinforcement Learning where the reward is not explicitly given, the Inverse Reinforcement Learning concept makes the reward available and is represented as the expert's

actions towards the successful completion of a task, in a fashion similar to supervised learning. In a notable contrast, Abbeel et al [89] succeeded in capturing low-level highly dynamic models representing an expert helicopter pilot performing complex manoeuvres. However, they reported challenges when trying to capture a main model that explains the high-level tasks or actions/strategies the human pilot is performing to manage the smooth transition from one manoeuvre to another or from one set of related rapid sub-tasks to another [89]. In other words, they reported the difficulty of capturing a sequence of sub-tasks or reward functions that makeup a complete general or high-level task rather than just a single reward function.

Literature suggests that supervised learning methods were relatively straightforward to apply in the context of LFD [82-84]. Even when Reinforcement Learning was the method of choice, it had to be modified [85-88] in a fashion that rather resembles supervised learning in "spirit". This was done by presenting the rewards that are not explicitly given in traditional Reinforcement Learning problems in a way that is similar to presenting training datasets where the labels or desired outputs are known, hence the term Inverse Reinforcement Learning [8].

Other than being popular classification techniques, Decision Tree/Random Forest learning algorithms represent efficient tools for inferring how the learning model was achieved due to their relatively easy to visualize and understand tree structure [82]. The structure allows the inference or understanding of how the decisions are made from the root across branches to the terminal leaves [82]. However, they are probably not the appropriate method for handling noisy and dynamic numerical values [83]. Literature suggests that the challenges reported when trying to handle noisy and dynamic numerical values are due to this nature of Decision Tree/Random Forest learning algorithms [83]. On the other hand, using Inverse Reinforcement Learning along with a suitable optimization technique proved to be able to handle noisy and dynamic numerical values, however, capturing the higher set of strategic tasks was challenging [8]. In [90], a comparison between a Random Forest Behavioural Cloning learning approach and Inverse Reinforcement Leaning approach was conducted. In their work, the Random Forest Behavioural Cloning learning approach performed better than the Reinforcement Leaning approach in the task of predicting Ground Delay Programs at airports, although both techniques struggled to predict when a delay is started and when it is cancelled [90]. Literature suggests that the better performance of Behavioural Cloning is due to its capability to capture high-level tasks such as managing airports' Ground Delay Programs that are not necessarily as dynamic as traversing or flying for example [90]. This comparison suggests a strong evidence to support the argument of Random Forest for capturing high-level models versus Inverse Reinforcement

Learning for capturing low-level models. The airports' Ground Delay Program problem is more likely to be a strategic (high-level) problem rather than a highly dynamic low-level problem, hence the better performance of Random Forest learning compared with Inverse Reinforcement Learning. In addition, The work of Bloem et al [90] suggests that Behavioural Cloning is suitable for high-level tasks, while Apprenticeship Learning is suitable for low-level tasks.

Extended analysis leads to the conclusion that in general, breaking down the problem of learning in the context of Learning by Imitation, or looking at the tasks to be captured as collections of strongly related sub-tasks enhances the learning model. This was observed in the work of Michini et al. [89]

The key concept summarized from the analysis of the literature, is the importance of initially analysing the given LFD problem to identify whether it is a low-level tasks/actions problem, a high-level tasks/actions problem, or in most cases, a combination of both. It is important to consider the appropriate learning method for both low-level and the high-level learning parts of the problem.

2.3 Autonomous Flying

The following section discusses recent research effort related to the scope of this work, which covers general aircraft control, handling emergency situations, navigation, landing, and flying under severe weather conditions, autonomously.

2.3.1 General Aircraft Control

To autonomously control an aircraft, the autopilot must manipulate the control surfaces and other control interfaces to perform the general tasks of piloting such as using the ailerons for banking or turning, the elevators for pitch maintenance, the elevators' trim for altitude maintenance, the throttle for speed maintenance, flaps for drag control, etc.

Controlling the aircraft's roll by using the ailerons is in the heart of navigation and path interception. In [91], an autopilot system that uses sliding mode control (SMC) method is proposed. The results show enhanced performance using MATLAB/Simulink environment [91]. A variant of the sliding technique used in [91] is used in [92] as well. The proposed SMC algorithm-based on nonlinear sliding surface is derived using the kinematic equations for bank-to-turn vehicles [92]. In addition, utilizing the rudder ensures the interception of the runway's centreline during takeoff and landing in the presence of crosswind. In [93], a grid method for computing the value function and optimal feedback strategies for the control and disturbance

is used to optimize the control of the rudder by handling nonlinear and linearized model of the aircraft on the ground. During final approach, maintaining a desired glideslope ensures safe and soft landings. In [94], controllers that modify the reference model associated with aircraft pitch angle are proposed. The control of the pitch angle and longitudinal velocity is performed by a neural network adaptive control system, based on the dynamic inversion concept [94]. In [95], a network model optimization algorithm based on onboard flight recorder data is suggested. For altitude control, a non-minimum phase (NMP) dynamic control systems is proposed in [96] where an invert closed loop system performed better than conventional Linear Quadratic Gaussian (LQG) when holding a given altitude. In [97], Artificial Neural Network's direct inverse control (DIC-ANN) with the PID control system is proposed where the linearization simplified the solving process for such mathematical based model, omitting the nonlinear and the coupling terms is unsuitable for the dynamics of the multirotor vehicle. Applying intelligent control methods to aircraft speed control is investigated in [98] where a speed command controller is enhanced by applying a command filter as well as an additional feed forward command. For flaps control, [99] proposes a dynamic flaps controller that continuously adjusts the flaps settings based on speed to achieve optimal flight dynamics throughout the flight. In addition, the controllability of a flap-controlled system is analysed based on nonlinear controllability theory [99].

2.3.2 Emergency Situations

Overcoming emergency situations is covered in recent research effort, mainly by investigating control systems that are fault-tolerant. When examining fault-tolerant systems, a fault is "an unpermitted deviation of at least one characteristic property of the system from the acceptable, usual, standard condition." [100], while failure "is a permanent interruption of a system's ability to perform a required function under specified operating conditions." [100]. To handle faults and failures, recent research effort has been focusing on designing Fault Detection and Diagnosis (FDD) systems that can either stream information to ground crew members especially in the case of UAVs, or feed fault-tolerant systems that are capable of handling such system faults [100]. The first type of such systems are known as the Passive Fault-tolerant Controllers which can handle moderate faults such as parameters deviations by using a robust feedback controller [96]. However, if the faults are beyond the capabilities of such controllers, another type of fault-tolerant systems becomes a necessity [100]. This type is known as an Active fault-tolerant control system which includes a separate FDD system that adds and extended and enhanced level of fault-tolerance capabilities [100]. An example of a

successful intervention of ground crew members that were alerted of the failure by the FDD system, is the emergency landing of a U.S. Airforce Global Hawk UAV [100]. The ground crew were able to execute a successful gliding descent remotely [100].

In [101], a Fault Detection and Diagnosis system is proposed for UAVs. The system continuously monitors the flight status and the systems' conditions, and detects faults or undesired behaviour, which are sent to the ground crew to manually intervene [101].

In case of emergency situations such as system failure conditions, mainly engine failure or fire, flight instruments failure, or control surface damage or failure, continuing to fly becomes either impossible or can introduce a serious threat to the safety of the flight, therefore, a forced or emergency landing on a suitable surface such as a flat field becomes a must especially if it is not possible to return safely to the runway [102]. In [102], an emergency landing control is proposed for an Unmanned Aerial Vehicle by segmenting the emergency landing period into four sub-levels known as slipping guiding, straight line down, exponential pulling up, and shallow sliding [102]. Each level uses different control strategies aimed at insuring the safe execution of the complete emergency landing [102]. For example, during the exponential pulling up level, the system maintains a certain pitch without causing the UAV to stall [102]. Using a simulator, the proposed approach showed its ability to handle emergency landing [102].

2.3.3 Navigation

Autonomous navigation is the ability of the travelling vehicle to estimate the state of its trajectory automatically [103]. In autonomous aerial systems, such as UAVs or cruise missiles, it is common to estimate the state of trajectory by fusing data from multiple navigation systems such as the Inertial Navigation System (INS) and the Global Navigation Satellite System (GNSS) such as the Global Positioning System (GPS) [103]. It is also possible to fuse additional data from different types of systems such as vision-based navigation systems [103]. The latter can be used either as an additional layer to enhance accuracy, or to be used on its own in situations where GPS signals are either unavailable, or at the risk of being jammed [103]. Such alternative systems are also used to tackle the problem of low-cost INS which can worsen if the travelling system is small and lightweight such as small UAVs which are affected more by the accumulating drifting errors over time [103].

In [104], an image matching system which uses aerial images acquired during flights in addition to aerial georeferenced images, is proposed to estimate the position of a UAV. The

proposed image matching system applies image-edge detection algorithms to the acquired images, and the posterior automatic image registration to estimate the location of the UAV [104]. An Artificial Neural Network (ANN) with an optimal architecture set by the Multiple Particle Collision Algorithm (MPCA) is used to detect the edges, while the automatic image registration is acquired through a cross-correlation process [104].

Different navigation and path planning approaches are being investigated as well. In [105], an algorithm based on inspection path planning is proposed, which is tailored inherently for structural inspection. The proposed algorithm is designed to computes full coverage and collision–free paths depending on a model of the UAV's nonholonomic constraints [105]. The inspection path is computed using a mesh–model representation of the desired area of traverse, by continuously attempting to calculate configurations of viewpoint which presents the desired coverage of the area, while applying the Lin-Kernighan heuristic [106] to achieve the best path while taking into consideration the motion constraints of the UAV [107]. A resampling of the viewpoint technique applies randomized sampling, which allows the designed algorithm to achieve continuous enhancements of the path cost without affecting the desired area to be covered [107]. In addition, navigation with a collision avoidance capability is achieved by applying Boundary Value Solver and a motion planner [107] for the used UAV model [107].

In [108], an autonomous navigation approach which does not rely on known artificial land-based synergic waypoints for UAVs, is proposed. The approach utilizes Speed Up Robust Features (SURF) which uses a Kalman filter that is based on multi-rate indirect and feedback error correction [108]. The latter is designed to integrate the measured data of the optical flow between two frames with the measured data of a Strap Down Inertial Navigation System (SINS), in addition to the readings of the altimeter, the laser range finder, and the electronic compass [108].

Given the current ability of the modern Global Positioning System (GPS) to provide superior location accuracy, recent research effort is investigating autonomous navigation systems that rely on GPS alone, such as the work presented in [109], where a cost-efficient cruise control system is designed for a GT-500 recreational aircraft using affordable and off-the-shelve components such as an Arduino system. GPS works by using receivers that receive signals sent from GPS satellites to determine the location through an approach known as Trilateration [110]. The latter is commonly mistaken for another process known as Triangulation which measures angles, while Trilateration measures distances [110]. GPS

measures longitude, latitude, and altitude in a three-dimensional world, where four satellites are needed to provide an accurate location [110]. Therefore, any given GPS receiver is required to be in line of sight of four or more GPS satellites always [110]. The four GPS satellites provide four different distances to a given object or location which is positioned on a sphere x distance from satellite 1, y distance from satellite 2, z distance from satellite 3, and w distance from satellite 4 [110]. The four distances are used to accurately determine the location [110]. GPS satellites use L-band radio waves which has an operating frequency that ranges between 1 and 2 GHz in the radio spectrum [110]. L-band frequencies have a wavelength that ranges between 15 and 30 cm [110]. The wavelengths of the L-band frequencies are able to penetrate all types of weather conditions including thick clouds due to their high bandwidth which is excellent for the purpose of modulating code, and introduces the advantage of making it easier for antennas to receive signals while needing low directionality due to their wide beam width [110].

For aviation, the newest L5 frequency is used, which is 1,176.45 MHz with a wavelength of 25.48 cm [111]. This L5 band "is protected worldwide for aeronautical radio-navigation use, and will support aviation safety-of-life applications [111]. The addition of L5 will make GPS a more robust radio-navigation service for many aviation applications, as well as all ground-based users (maritime, railways, surface, shipping, agriculture, recreation, etc.)" [111]. GPS gives the desired level of aviation safety and efficiency during flights by offering an accurate 3D position determination during all phases of the flight [111]. This allows pilots to fly any desired path or route without having to rely on ground-based waypoints, especially while flying over data-sparse areas like oceans [111]. GPS is also able to give accurate locations when approaching landing runways which greatly enhances both safety and operational benefits as well [111]. One of the recent additions to the Global Positioning System is the application of the Wide Area Augmentation System (WAAS), which when coupled with the recent introduction of the L5 frequencies, GPS errors which are caused by the ionosphere are greatly minimized which adds to the reliability of the Global Positioning System [111].

In [112], a GPS based generic trajectory prediction and smoothing algorithm is proposed. The algorithm is deigned to be able to handle both accurate frequency legs, and inaccurate legs that are present in old flight procedures, that have not been updated using advanced Flight Management Systems (FMS) [112]. Handling inaccurate legs is done by obtaining the exact coordinates of the start and end points of each leg via computations of the performance and the geodetic measurements, while smooth transitions are obtained between adjacent legs using

Radio Frequency (RF) legs [112]. The estimation of the desired trajectory is calculated using numerical integration of the different states of the aircraft given the flight path [96].

2.3.4 Landing

Pilots operating Remotely Piloted Aircraft Systems (RPAS) or UAVs do not get to feel the aircrafts they are flying as on-board pilots do [113]. Feeling the forces of the surrounding environment such as the wind, and the aircraft itself, such as getting a feel of how the engines are behaving, the vibrations, motions, and so on, is not possible for ground pilots [113]. The lack of this on-board sensing affects the situational awareness which is a crucial factor that pilots depend on especially during the most difficult flight phases such as landing [113]. Therefore, most UAV accidents happen during landing [113]. In addition, performing an optimum landing all the time is important for maintenance cost reduction, and durability preservation [113]. So, investigating the possibilities of developing autonomous landing systems (Auto Land) for UAVs has been a significant challenge, and is being covered in recent research effort [113].

In [113], a landing sequence algorithm is proposed, which can either be initiated by the ground pilot, or automatically during emergency situations such as the loss of connection between the UAV and the ground command and control station. The proposed landing system utilizes the Global Positioning System (GPS) along with geometry to orient the UAV to a desirable point in space from which it can initiate the descend process [113]. The algorithm works by plotting multiple slopes via MATLAB, and are considered as potential descend paths that the UAV can follow, in a fashion like creating a virtual inverted cone, where the circular base of the cone can act as a potential point of descend, and the taper surface can be considered as the glide path [113]. First, a comparison between the UAV's position, and the starting point of each generated slope is performed [113]. Then, the closest starting point is designated as the chosen point of descend, and the UAV is autonomously flown to that point, and finally, the landing process is initiated [113]. The proposed system does not rely on ground-based support systems such as Instrument Landing Systems (ILS) [113].

A landing system inspired by the Global Positioning System (GPS) is proposed in [114]. The system is intended for autonomous UASs landing on aircraft carriers when GPS signals are denied [114]. This design of the system incorporates several radio transmitters which operate like GPS satellite transmitters, and are positioned on the deck of the aircraft carrier [114]. The transmitters, and the receivers that are integrated with the aircraft, are equipped with

antennas that are specially designed to provide extra directional information to be gained during each transmission [114]. The proposed system can calculate the Pseudo-Range (PR) between the transmitter and the receiver, and in addition, calculate the Angle of Attack (AoA), and the Angle of Transmission (AoT) as well [114]. The autonomous and accurate interception of the landing runway line requires the integration of the readings generated from the Inertial Measurement Unit (IMU) of the aircraft, and the Radio Frequency (RF) sensors [114]. The proposed system can utilize two different navigation filters, an Unscented Kalman Filter (UKF), and a Nonlinear Maximum Likelihood Estimator (NLMLE) [114].

Other than GPS guided landing, different solutions are being investigated. In [115], an autonomous landing system for fixed-wing Unmanned Aerial Systems (UAS) that depends on ground instruments is proposed. The system utilizes a passive Ultra Wide Band (UWB) positioning network which listens to the UWB signals that are continuously sent from the UAS [115]. The UWB positioning network requires minimal UWB communications, and does not interfere with the Radio Frequency (RF) signals that are used by different systems in the airport [115]. The proposed system calculates the position of the UAS through the UWB signals that are sent from the aircraft, then, the runway alignment information is sent back to the aircraft via a protected aviation communication channel, in a fashion that is similar to how current Instrument Landing Systems (ILS) work [115].

To achieve higher levels of accuracy required for landing on significantly small, or moving landing runways such as aircraft carriers, some recent research effort is focusing on fusing multiple guidance systems, such as the work presented in [116]. The proposed system works by fusing readings from multiple systems or sensors including GPS, the aircraft's INS, the aircraft carrier's INS, and a vision-based navigation system mounted on the aircraft [116]. The system computes the aircraft-ship relative position, while the acceleration and velocity of the ship are sent to the aircraft via a dedicated data-link [116]. The aircraft-ship relative position, and the relative velocity are added to the state vector, and the relative position information retrieved from GPS, along with the airborne INS, the carrier's INS, and the vision-based navigation system are utilized to build the vector via a Kalman filter. Finally, the relative position information having the same period as the one generated from the INS is calculated [116].

Vison-based landing systems are being considered to serve as either sole landing systems or to support other landing systems [117][118][119]. In [119], a landing approach which utilizes

a combination of Strap Down Inertial Navigation System (SINS), and a vision-based system is proposed for autonomous landing. The system works by calculating the trajectory during the glide phase and the flare phase, while the navigation output is computed using the SINS [119]. Next, during the glide phase, a certain object is positioned on the landing runway, which is captured by the UAV's camera, and used to estimate the interception accuracy of the runway line [119]. The drift errors of the SINS are corrected using a Kalman filter [119]. During the flare phase, the edges of the runway and the centre line are detected through the vision system, which are used to estimate the desired position of the UAV [119].

In [120], A comprehensive Autoland design for a representative model of a twin-engine commercial aircraft is proposed where a cascaded control structure is selected which resembles integrator chains. The classical loop shaping is used to design the individual control loops where the emphasis is on providing a complete and comprehensive qualitative design strategy [120]. In [121], a control system architecture with strong disturbance rejection characteristics for Unmanned Aircraft is presented where the primary objective is to accurately land a fixed-wing aircraft under adverse weather conditions. A synergistic controller architecture is presented, where the aim is to design a structure capable of executing one of three landing techniques, or combination thereof, by simply activating various controllers at different stages of the landing phase [121]. An acceleration-based controller architecture is used for the inner-loop controllers to reject disturbances at the acceleration level before they manifest as deviations in inertial position and velocity [121]. [122] proposes an autonomous approach and landing navigation method whose accuracy is comparable with Inertial/Differential GPS (DGPS) integration. The method integrates inertial data, forward-looking infrared (FLIR) images, and runway geographic information to estimate kinetics states of aircraft during approach and landing [122]. An existing method is enhanced to robustly detect runway, accurately extract three vertexes of runway contour from FLIR images and synthesize the virtual runway features by runway geo-information and aircraft's pose parameters [122]. Then, real and synthetic runway features are used to create vision cues and integrate them with inertial data in square-root unscented Kalman filter to estimate the motion errors [122]. [123] proposes an improved multi-group swarm-based optimization method that can not only optimize the parameters of the lateral flight control system, but also find diversity solutions of the underlying optimization problem. During the optimizing process, several swarm groups are generated to search potential areas for the optimal solution [123]. These groups exchange information with

each other during the searching process and focus on their different but continuous spaces [123].

2.3.5 Severe Weather Navigation & Landing

The effect of wind disturbance affecting the performance of autonomous navigation systems or UAVs is being simulated and analysed in recent research effort to investigate possible solutions to this problem which significantly affects small fixed-wing UAVs due to their relatively light weight, and lower thrust capability [124] [125]. Wind disturbance causes the UAV to drift from the desired course, and when added to the accumulated errors of the navigation systems, maintaining a desired flight path or course becomes a significant challenge [124] [125].

In [126], the physical properties of the Vehicle Dynamic Model (VDM) are used to study the effects of wind on navigation systems in addition to the control inputs within the algorithm of the navigation filter. The latter is used to investigate the possibility of increasing the accuracy as well as the reliability of the autonomous navigation system [126]. The proposed approach utilizes solutions to the VDM equations to present an estimation of the UAV's position, its velocity, and attitude, which are updated within the navigation filter based on the available observations, while the navigation filter provides an estimation of wind velocity and the parameters of the dynamic model [126]. Monte Carlo based simulations via real three-dimensional wind velocity data are used to investigate the possibility of enhancing the performance of the autonomous navigation system in the presence of dynamic fluctuations in wind velocity [126].

In [127], the authors proposed an approach to tackle strong wind effects during flights. The approach estimates wind effects that are steady and strong in nature, and delivers a manoeuvring strategy to tackle such conditions [127]. Wind effects estimation utilizes the Global Positioning System (GPS) velocity via a new filter design that does not require airspeed readings [127]. Then, the calculated estimation of the wind effects is used to force the UAV to enter a crabbing manoeuvre in a direction that is perpendicular to the direction of wind [127]. This technique allows the UAV to avoid significant drifting from the desired flight path [127]. The proposed algorithm applies the vector relation between the wind velocity, air, and ground to calculate an estimation of the steady wind effects such as the crosswind component, and the airspeed using a recursive least square algorithm which exploits the geometric relation between the inertial velocity and the effects of wind [127]. In addition, a basic guidance approach is

included to handle the challenge of maintaining a low speed flight under strong wind conditions [127].

Landing in the presence of crosswind conditions requires a final approach that is carefully flown to ensure the desired position of the aircraft on the glide path, and the appropriate interception of the line that represents the centreline of the runway [128]. This must be handled before the initiation of the flare stage which is the slight lift of the aircraft's nose just before touchdown to decrease the sink-rate, and land the aircraft on the main landing-gear first [128]. The final stage is what is known as the Decrab stage which is the cancellation of the crabbing orientation, and aligning the aircraft's fuselage with the centreline of the runway [128]. To tackle crosswind during an approach, two methods are used, the first method is known as Crabbing where a certain degree of drift or crab is induced to change the orientation of the aircraft's nose heading towards the direction of the wind [128]. The second method is known as Wing-down, in which a steady sideslip is induced to tackle the drift caused by the crosswind [128]. In practice, it is common to combine both methods, following degrees which could vary during the approach phase [129]. For the Boeing 777 of which a simulated model is used in this work, the maximum crosswind components are 45 knots for a dry runway, and 40 knots for a wet runway [128].

The three stages performed during crosswind landing which are the crab stage, the flare stage, and the decrab stage require a set of high skills to be possessed by the human pilot [130]. These skills are being analysed in multiple research effort to investigate the possibility of designing autonomous landing systems that are equipped with synthetic versions of these skills [130]. Artificial Neural Networks (ANNs) were used for the purpose of estimating a mapping relationship between the given situation, and the human pilot inputs [130] [131]. In addition, the possibility of using conventional Control Theory fault tolerance techniques, that are used for Proportional Integral Derivative (PID) controllers to tackle the crosswind landing challenge, is being investigated [132]. For example, applying Integral Windup handling methods which are used in situations where a large change happens, and a significant error accumulation causes overshooting, which is similar to the overshooting or drifts from the desired path line caused by crosswind [132].

2.3.6 The Intelligent Autopilot System (IAS)

In a report [133] prepared for NASA by Honeywell Aerospace and Defence, the Intelligent Autopilot System (IAS) described in this thesis is briefly evaluated with the emphasis on the

problem-breakdown approach of the IAS where multiple and independent small components designed to handle specific tasks are managed by a high-level component, which is in line with the recommendations of the report. However, [133] mistakenly claims that the IAS uses Inverse Reinforcement Learning where capturing a sequence of sub-tasks or reward functions that makeup a high-level task becomes quite challenging [7]; in fact, the IAS uses Supervised Learning by applying fully connected single-hidden-layer Artificial Neural Networks (ANNs), which is a method that can undergo Verification and Validation (V&V) given the absence of a black-box. [133] emphasizes the need for assuring that the intelligent control system must not behave unexpectedly and must have a certain level of situational awareness where the behaviour is altered to handle an emergency for example. Although the IAS is a proof-of-concept designed to prove the possibility of introducing intelligent autonomy to the cockpit, not a fully developed mature autopilot, attention was given to the assurance points by making sure the training datasets contain specific patterns that guarantee the elimination of unexpected behaviour. In addition, the IAS is capable of detecting several unusual conditions such as emergency situations where the behaviour is altered to cope with the situation.

2.3.7 Analysis of the Literature of Autonomous Flying

In addition of having limited capabilities, modern autopilots can contribute to catastrophes since they can only operate under certain conditions that fit their design and programming, otherwise, they cede control to the pilots, and with the lack of proper situational awareness and reaction, the result could be fatal [4]. Therefore, introducing intelligent autonomy to the aviation industry through developing intelligent control techniques that fit into an overall flight management system capable of making the highest level of decisions, is expected to significantly enhance safety, and lower costs [134]. Manned aircrafts especially airliners require significant attention to enhance safety by addressing the limitations of modern autopilots and flight management systems, and the human error factor as well.

However, the current focus of the relevant and recent research effort is on Unmanned Aircraft Systems (UAS) especially small and micro drones [87-127] by introducing solutions that may not be suitable for medium to large jets, although the civil aviation sector that uses these jets is the largest with the highest risk and costs. Although the literature review provided in section (*2.2 Autonomous Flight*) presents valid solutions to the autonomous flying problem when it comes to navigation, landing, and handling severe weather, the solutions were intended for platforms that are not part of the scope of this work. Using computer vision for example proved to enhance accuracy [117] [118], however, airliners are not equipped with such sensors.

The proposed techniques that tackle the wind effect may work well with light-weight platforms, but not necessarily with heavy platforms such as airliners. Therefore, it is not possible to provide a thorough analysis of the related literature in this section. The only exception is the work presented in [130] and [131], where the crabbing technique performed by pilots during crosswind landing is analysed using Artificial Neural Networks, however, the authors did not provide an approach aimed towards applying the analysed results in autonomous systems fit for such airplanes.

It is clear that intelligent autonomy is covered in the literature in many recent research papers, however, the work is dedicated to tackling specific flight automation problems such as maintaining speed or altitude rather than proposing comprehensive cockpit autonomy solutions such as the IAS. In most papers, the authors tackle the robustness issues of PID controllers that modern autopilots rely on by applying a layer of intelligent control to those conventional controllers such as adding Artificial Neural Networks or Fuzzy Logic to the PID controllers closed loop to enhance performance and accuracy [37] [96] instead of fully replacing them with intelligent control solutions, which increases the complexity of the proposed solution rather than attempting to simplify it. Therefore, given the lack of work that studies airliners, and the significant need for such work, solutions that can be applied to multiple aircraft categories especially large jets such as airliners and cargo airplanes are proposed in this work.

3. PROTOTYPE 1 (METHODOLOGY & BASIC FLYING)

This work aims to address the problem of robustness and limited capabilities that modern autopilots of large jets suffer from by proving the hypothesis that Learning from Demonstration enables automated flight control comparable with experienced human pilots. To provide evidence to support the hypothesis, an autopilot that utilises Learning from Demonstration must be designed to prove the possibility of learning from human pilots how to perform piloting tasks, perform full flight cycles, handle emergencies and uncertainties including severe weather conditions, and behave like experienced human pilots of airliners. The proposed autopilot must undergo iterative testing in the flight simulator covering different scenarios ranging from basic piloting tasks to complete flight cycles in normal and severe conditions including emergencies in order to compare its capabilities with flights by human pilots. Based on the results of the different tests, the capabilities of the proposed autopilot should be improved incrementally to cover all the objectives starting from the ability to perform basic piloting tasks, and ending with the ability to behave like experienced human pilots of airlines.

The proposed Intelligent Autopilot System (IAS) can be viewed as an apprentice that observes the demonstration of a new task by the human teacher, then, performs the same task autonomously after training. A successful generalization of Learning from Demonstration should take into consideration the capturing of low-level models and high-level models, which can be viewed as rapid and dynamic sub-actions that occur in fractions of a second, and actions governing the whole process and how it should be performed strategically. It is important to capture and imitate both levels in order to handle flight uncertainties successfully.

The IAS is made of the following components: a flight simulator, an interface, a database, and Artificial Neural Networks. Choosing Artificial Neural Networks (ANNs) for this work is due to the analysis of the literature (2.1.3 Analysis of the Controllers Literature) which suggests that ANNs are superior methods for handling highly dynamic, and noisy data that are present in flight control environments. Conventional controllers such as Proportional Integral Derivative controllers are not used since the literature (2.1.1 Non-Adaptive Linear Controllers) mentions their inability to learn sequences of tasks and handle uncertainty, instead, the proposed system relies on ANNs as the sole controllers. Since further analysis of the literature suggests that breaking down a highly dynamic problem yields better results (2.1.2.3 Multiple Controllers & Sensors), many Artificial Neural Networks were developed instead of just one, each designed to handle specific control. In addition, the literature suggests that using

Supervised Learning is straightforward to apply and robust in the context of Learning from Demonstration compared with other approaches such as Inverse Reinforcement Learning (2.1.3 Analysis of the Learning from Demonstration). Therefore, the IAS implementation method has the following three steps:

A. Pilot data collection (labelled training sets collection suitable for Supervised Learning).
B. Training (offline Supervised Learning training using ANNs).
C. Autonomous control.

In each step, different IAS components are used. The following sections describe each step and the components used in turn.

The work in this chapter was published in the 2016 International Conference on Unmanned Aircraft Systems (ICUAS), Arlington, VA, USA (Appendix B).

3.1 Pilot Data Collection

Fig. 3.1 illustrates the IAS components used during the pilot data collection step.

3.1.1 Flight Simulator

Before the IAS can be trained or can take control, demonstration data from a pilot must be collected. This is performed using X-Plane which is an advanced flight simulator for professional applications that provides unprecedented features such as highly accurate modelling of physics by using Blade Element Theory which is an engineering process that involves breaking the aircraft down into many small elements and then finding the forces on each element many times per second [135]. These forces are then converted into accelerations which are then integrated to velocities and positions [135]. On the other hand, other flight simulators use Stability Derivatives which is a far too simplistic modelling approach [135]. In addition, X-Plane provides built-in data I/O, and the ability to simulate various scenarios such as emergencies compared with other popular flight simulators such as FlightGear and Microsoft X Flight Simulator [135].

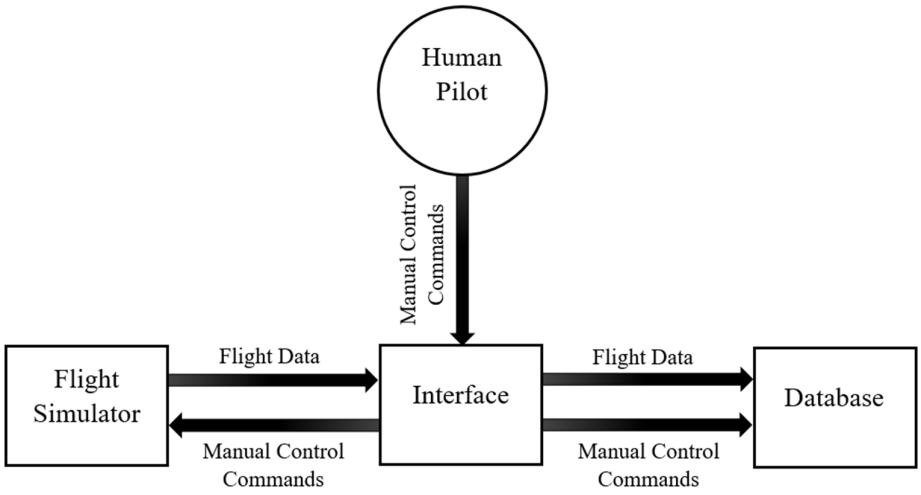

Fig. 3.1. Block diagram illustrating IAS components used during the pilot data collection step.

X-Plane is used by multiple organizations and industries such as NASA, Boeing, Cirrus, Cessna, Piper, Precession Flight Controls Incorporated, Japan Airlines, and the American Federal Aviation Administration.[2] It has been used as the simulator of choice in many research papers such as [136] [137] [138] [139] [140] [141] [142] to test the performance of the authors' contributions to the autonomous UAV field. X-Plane can communicate with external applications by sending and receiving flight status and control commands data over a network through User Datagram Protocol (UDP) packets. For this work, the simulator is set up to send and receive packets comprising desired data every 0.1 second since initial empirical testing indicated that this rate is fast enough to receive and send flight and control data, and at the same time, suitable for the hardware capabilities of the computer used for this work.

3.1.2 IAS Interface

The IAS Interface is responsible for data flow between the flight simulator and the system in both directions. In addition, the Interface contains a Graphical User Interface which provides simplified yet sufficient aircraft control buttons including throttle, brakes, gear, elevator, aileron, and rudder buttons which can be used to perform basic tasks of piloting an aircraft such as take-off in the simulator. It also displays flight data received from the simulator.

[2] X-Plane 10 Global

http://www.x-plane.com [accessed 2015]

As mentioned above (*3.1.1* Flight Simulator), X-Plane can send and receive flight data and control commands through its ports. UDP Packets received from or sent to X-Plane consist of 41 bytes (40 indexes starting from 0). For example, a UDP raw packet received from X-Plane containing the 'Throttle' variable is as follows:

68,65,84,65,64, 25,0,0,0, 9,36,185,62, 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0

Decoding this packet gives the following:

- 68,65,84,65,64: The first 4 bytes is an identifier representing "DATA" (the fifth is ignored).
- 25,0,0,0: Byte 6 is the value that refers to a specific data in X-Plane's input/output table. In this example, index 25 refers to 'Throttle Command'.
- The remaining 32 bytes represent the value of the data. It is made of eight groups of four bytes, or eight single precision floating-point numbers. In this case 9,36,185,62 gives (0.361809) after converting the four bytes to a single precision floating point, which is the value of the current engine thrust. The remaining zeros indicate that there are no further data in this particular example (single-engine aircraft).

When receiving more than one value from the simulator, the next value is added to the UDP packet after the end of the previous one. For example, a UDP raw packet containing two variables: 'Gear/Brakes' (handled in one packet) and 'Throttle' is as follows:

68,65,84,65,64, 14,0,0,0, 0,0,128,63, 0,0,128,63, 0,0,0,0, 0,0,0,0, 0,192,121,196, 0,192,121,196, 0,192,121,196, 0,192,121,196, 25,0,0,0, 149,20,16,62, 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0

Decoding this packet gives the following:

- The first variable (index 14 in X-Plane input/output table) refers to 'Gear/Brakes'. The decoded values for 0,0,128,63 are 1 (Down) for 'Gear' and also 1 (ON) for 'Brakes'.
- Since the first variable data ends at element (40), the next variable starts at element (41), which represents as before, the index of the variable in X-Plane. "DATA" is not sent again.

X-Plane has a data input/output window in its user interface where users can pick the variables that the simulator sends and receives. Each variable refers to a certain control

interface or surface such as the throttle or ailerons. A simple formula to calculate how many variables are included in a packet is as follows:

$$v = (b - 5) / 36 \qquad (3.1)$$

where v is the number of variables, b is the number of bytes, (36) is the number of bytes in each variable excluding "DATA" and the ignored fifth byte, and 5 is the first five bytes ("DATA" + the ignored 5th byte).

The same method of UDP row packets communication is applied when sending UDP packets to the simulator from an external application. The exact same structure of the UDP raw packet used when receiving from X-Plane as explained above is used for sending as well. When sending packets, the number of variables to be sent to X-Plane is known since each variable represent a specific control interface or surface, therefore, the number of bytes must be determined based on the number of variables to be sent. The latter can be achieved as follows:

$$b = 36v + 5 \qquad (3.2)$$

Data collection is started immediately before demonstration. Then, the pilot uses the Interface to perform the piloting task to be learned. The Interface collects flight data from X-Plane over the network using UDP packets, and collects the pilot's actions while performing the task, which are also sent back to the simulator as manual control commands. The Interface organizes the collected flight data received from the simulator (inputs), and the pilot's actions (outputs) into vectors of inputs and outputs, which are sent to the database every 1 second.

3.1.3 Database

An SQL Server database stores all data captured from the pilot demonstrator and X-Plane, which are received from the Interface. The database contains tables designed to store: 1. continuous flight data as inputs, and 2. pilot's actions as outputs. These tables are then used as training datasets to train the Artificial Neural Networks of IAS.

3.2 Training

3.2.1 Artificial Neural Networks

After the human pilot data collection step is completed, Artificial Neural Networks are used to generate learning models from the captured datasets through offline training. Fig. 3.2 illustrates the training step.

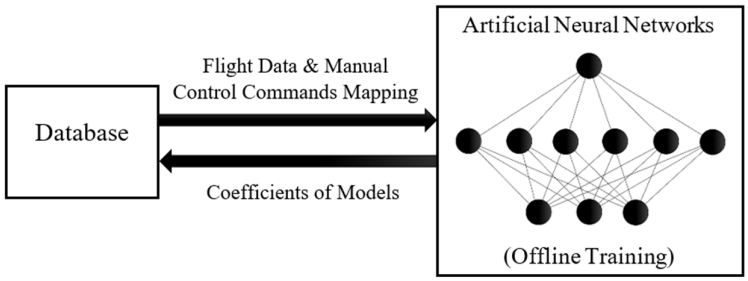

Fig. 3.2. Block diagram illustrating IAS components used during training.

The method for choosing ANN topologies in this work is based on an implication [143] which indicates that direct mapping problems requiring more than one hidden layer are rarely encountered, and compared with Deep Learning, this approach means that the system is more understandable and easier to test and verify compared with single deep solutions which are black-boxes that are difficult to interpret [143]. In addition, it is suggested that to avoid under-fitting caused by too few neurons in the hidden layer, or over-fitting caused by too many neurons, the number of hidden neurons should be less than or equal to twice the size of the input layer [144].

The design approach of the ANNs breaks down the different tasks required for flying, that take place during the multiple flight phases. The break-down approach methodology identifies each control interface or surface of the aircraft, and assigns a dedicated ANN to manipulate it. For example, to control the aircraft's pitch, the control surfaces of the aircraft that are used for this purpose are the elevators, therefore, a dedicated ANN is designed and trained to manipulate the elevators for the purpose of controlling the aircraft's pitch. Another example is designing and training a dedicated ANN for the purpose of controlling the aircraft's gear. The alternative for the proposed break-down approach is designing and training a single or few large ANNs that manipulate all the required interface and control surfaces of the aircraft, however, this would yield a large ANN that could represent a black-box which is difficult to design, train, and interpret. In addition, if a single control component requires redesigning or further enhancement, the single large ANN which controls all the other control components must be redesigned and retrained, which could affect the overall performance and hinder progress. Therefore, the proposed break-down approach allows for the ability to isolate any single control component for the purpose of maintenance or enhancement without affecting the overall system.

The ANNs (control commands) dictate the required inputs that correlate to the desired outputs. For example, the ground-run phase, where the pilot attempts to gain speed required for takeoff, is done by releasing brakes (output) and pushing to full throttle (output) while monitoring airspeed (input), and the task of keeping the aircraft on the centreline of the runway is done by using the ruder (output) based on the heading (input). For these tasks, two ANNs can be designed. The first would control the brakes and throttle based on airspeed (task 1), and the second would control the rudder (task 2) based on heading, by predicting the appropriate control commands based on the relevant flight data inputs.

In empirical experiments to evaluate the approach of normalizing the training data to values between 0 and 1 versus keeping the values unnormalized, the latter resulted in lower training Mean Squared Error compared with normalized training datasets as suggested in [145]. In addition, scaling the inputs to make them a magnitude closer to the outputs resulted in lower Mean Squared Error as well compared with unscaled inputs. An example of scaling to achieve a close magnitude of the inputs and outputs is illustrated in Table 3.1 which shows the climb rate inputs captured from the appropriate aircraft instrument in the flight simulator before and after scaling to suite the outputs of the elevators trim in this example. Since the essence of the IAS is control function approximation, following this approach which is suitable for such problems as suggested in [145] proved to yield better models with lower Mean Squared Error. Additional empirical experiments were conducted to evaluate whether applying the Sigmoid activation function (3.3) [144] for training datasets with positive values would yield better results compared with applying the Hyperbolic Tangent activation function (3.4) [144] to train ANNs on all training datasets (positive and negative). However, using (3.3) when the training datasets consist of positive values only, resulted in lower training Mean Squared Error compared with using (3.4).

Therefore, the approach of using unnormalized values, scaling, and applying (3.3) when the datasets contain only positive values, is applied uniformly throughout this work.

TABLE 3.1
SCALING THE INPUTS BY DIVIDING THEM BY 1000 TO ACHIEVE A CLOSER MAGNITUDE TO THE OUTPUTS

Unscaled Inputs	**Outputs**	**Scaled Inputs**	**Outputs**
-2810	0.566	-2.810	0.566
-2532	0.45	-2.532	0.45
-2235	0.312	-2.235	0.312
-1521	0.23	-1.521	0.23

-500	0.11	-0.500	0.11
0	0	0	0
491	-0.123	0.491	-0.123
1600	-0.219	1.600	-0.219
2190	-0.323	2.190	-0.323
2559	-0.46	2.559	-0.46
2796	-0.5	2.796	-0.5

During training, the datasets are retrieved from the database. Then, the datasets are fed to the ANNs. Next, Sigmoid (3.3) and Hyperbolic Tangent (Tanh) (3.4) functions are applied for the neuron activation step, where e is the exponential function, and x is the neuron output:

$$\Phi(x) = \frac{1}{1+ e^{-x}} \qquad (3.3)$$

$$\Phi(x) = tanh\,(x) \qquad (3.4)$$

The Sigmoid activation function (3) is used by ANN 1 since all input and output values are positive, while Tanh (3.4) is used by ANN 2, 3, and 4 since the datasets contain few negative values: pitch (ANN 2), rudder (ANN 3), roll, and aileron (ANN 4).

Next, Backpropagation is applied. Based on the activation function, (3.5) [144], or (3.6) [144] are applied to calculate the derivatives of the relevant activation function, where phi (Φ) of x is the result of the activation function:

$$\Phi'(x) = \Phi(x)(1 - \Phi(x)) \qquad (3.5)$$

$$\Phi'(x) = 1.0 - \Phi^2\,(x) \qquad (3.6)$$

Finally, coefficients of models (weights and biases) are updated using (3.7) [144], where ϵ is the learning rate, $\frac{\partial E}{\partial w_{(t)}}$ is the gradient, α is the momentum, and $\Delta w_{(t-1)}$ is the change in the previous weight:

$$\Delta w_{(t)} = -\epsilon \frac{\partial E}{\partial w_{(t)}} + \alpha \Delta w_{(t-1)} \qquad (3.7)$$

When training is completed, the learning models are generated, and the free parameters or coefficients represented by weights and biases of the models are stored in the database.

Since training is done offline, an Artificial Neural Network training software was developed for this purpose using C# and Microsoft Visual Studio 2015. No third-party libraries or resources were used for this software, and all the functionalities were developed from scratch. For ease of use and flexibility, the training software has a Graphical User Interface (GUI) as Fig. 3.3 shows, which allows the user to select several options before learning starts. The user

can select the training dataset source, enter the preferred Learning Rate (α) which is a positive scalar that dictates the size of the step or the magnitude by which the weights are updated [144], and the Momentum (μ) which accumulates an exponentially decaying moving average of past gradients and continues to move in their direction to avoid falling in a local optima [144]. In addition, the user can select the preferred activation function, select the number of epochs, the desired minimum error, the ANN topology (number of hidden neurons and hidden layers), evaluate the learning results, and compare the generated learning model against a validation set (compare the Training Mean Squared Error against the Test Mean Squared Error). The user can save the learning model coefficients in the database.

In empirical experiments to identify the optimum values of the Learning Rate (α), the Momentum (μ), and the number of epochs suitable for training the ANNs in this work given the similarity between the different training datasets in size, number of features, and number of labels, setting the Learning Rate (α) value at 0.1, the Momentum (μ) at 1, and the training epochs to a maximum of 10,000 resulted in better conversion and lower training Mean Squared Error. Therefore, the latter training settings are applied uniformly throughout this work.

3.3 Autonomous Control

Once trained, the IAS can now be used for autonomous control. Fig. 3.4 illustrates the components used during the autonomous control step.

3.3.1 IAS Interface

Here, the Interface retrieves the coefficients of the models from the database for each trained ANN, and receives flight data from the flight simulator every 0.1 second. The Interface organizes the coefficients into sets of weights and biases, and organizes data received from the simulator into sets of inputs for each ANN. The relevant coefficients, and flight data input sets are then fed to the ANNs of the IAS to produce outputs. The outputs of the ANNs are sent to the Interface which sends them to the flight simulator as autonomous control commands using UDP packets every 0.1 second.

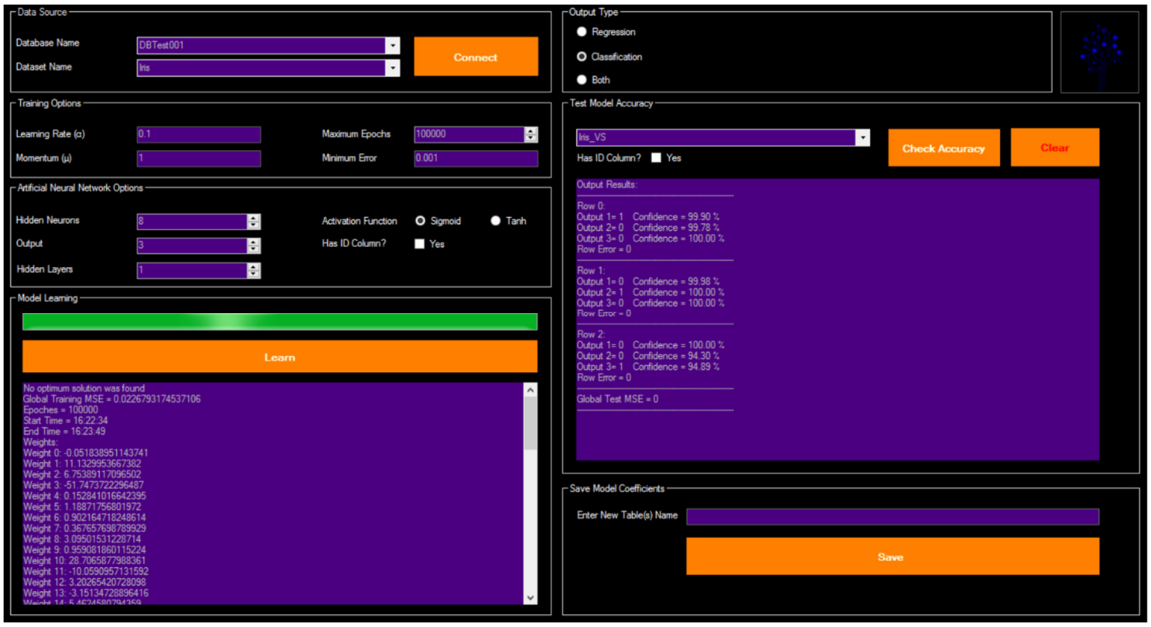

Fig. 3.3. GUI of the ANN training software.

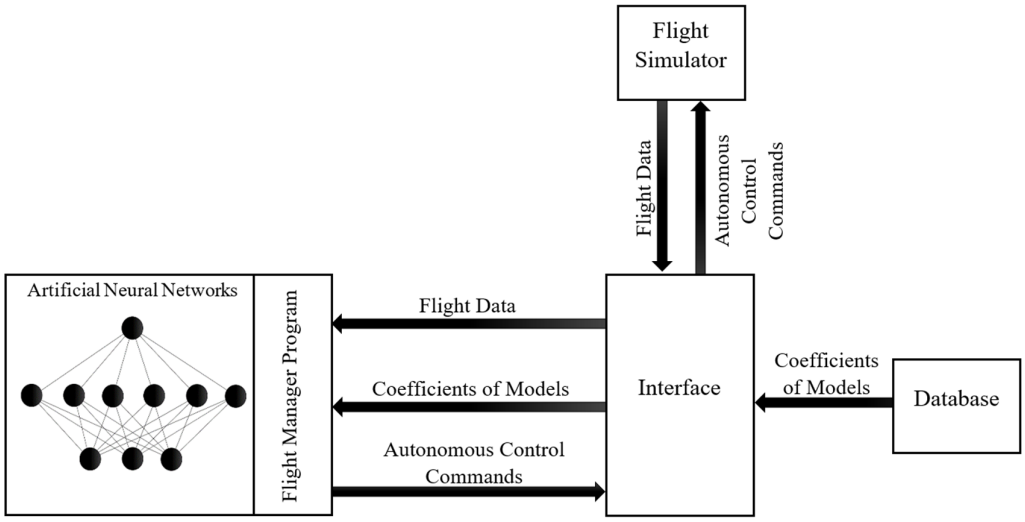

Fig. 3.4. Block diagram illustrating IAS components used during autonomous control.

3.3.2 Artificial Neural Networks

The relevant set of flight data inputs received through the Interface is used by each ANN input neurons along with the relevant coefficients to predict and output the appropriate control commands given the flight status by applying (3.3) and (3.4). The values of the output layer

are continuously sent to the Interface which sends them to the flight simulator as autonomous control commands.

3.4 Testing the Proposed Methodology

To test the proposed development, training, and testing methodology of this work, the first prototype was designed with the objective to prove the ability of Artificial Neural Networks to learn piloting tasks by generating learning models from training datasets containing demonstrations performed by a human teacher in a flight simulator. The learning models should capture low-level and high-level skills and abilities that would enable the IAS to perform basic flights under calm and severe weather conditions.

For the objective of achieving autonomous basic flying, the IAS should imitate the behaviour of the human pilot during the ground-run, takeoff, climb, and cruise phases since these flight phases are relatively less complex compared to other challenging flight phases such as final approach and landing. During each of these segregated four phases, the IAS should be able to manipulate a number of controls including brakes, throttle, rudder, ailerons, and elevators, based on a number of flight data inputs including speed, altitude, pitch, roll, and heading. To achieve this, and based on the four segregated phases, a break-down of the overall problem to four independent and small tasks, each requiring the monitoring and manipulation of different inputs and controls, is required.

Based on the approach mentioned in section (*3.2 Training*), four feedforward Artificial Neural Networks were designed. Each ANN is designed and trained to handle specific controls. ANN 1 takes speed and altitude values as inputs, and predicts throttle, gear, and brakes values. ANN 2 takes speed, altitude, and pitch values as inputs, and predicts elevators value. ANN 3 takes the roll value as input, and predicts ailerons value. ANN 4 takes the heading value as input, and predicts the rudder value. The topologies of the four ANNs are illustrated in Fig. 3.5.

3.4.1 Experiments on Prototype 1

In order to assess the effectiveness of the proposed approach, the Intelligent Autopilot System was tested in two experiments:

1. autonomous flying under calm weather

2. autonomous flying under stormy weather

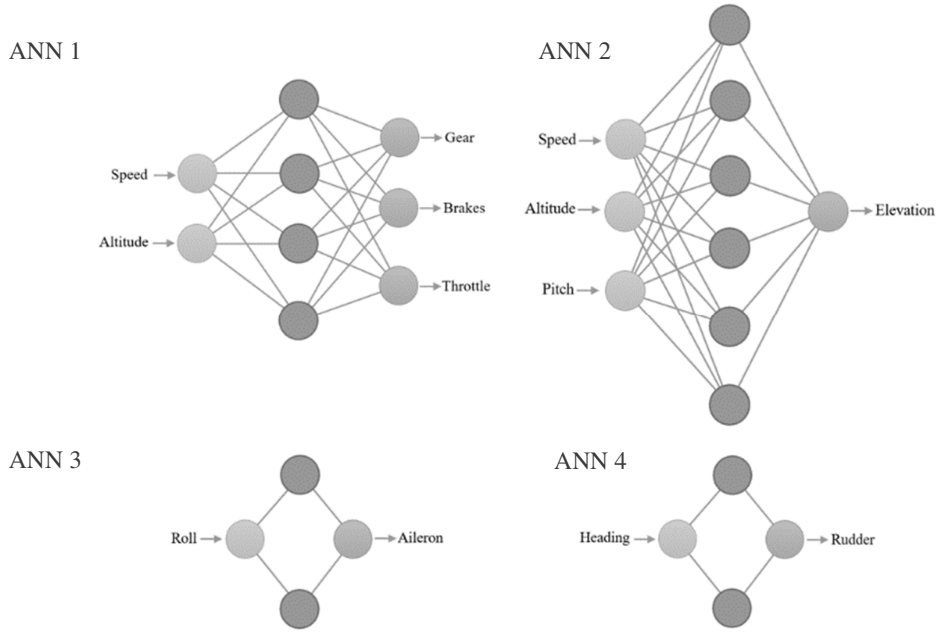

Fig. 3.5. Topology of ANN 1 trained to handle throttle, gear and brakes (top left), topology of ANN 2 trained to handle elevator control (top right), topology of ANN 3 trained to handle aileron control (bottom left), and topology of ANN 4 trained to handle rudder control (bottom right).

Each experiment is composed of 10 attempts by the IAS to fly autonomously under the given weather conditions. Fig. 3.6 illustrates a break-down of the piloting task to be learned, to four sub-tasks based on time. Each attempt lasted for 182 seconds. The human pilot who provided the demonstrations is the author. Before using a large jet (airliner) in the simulator to prove the possibility of teaching an autopilot how to fly, and since this chapter aims to prove the possibility of teaching an autopilot basic piloting tasks, using a light single-engine aircraft for this purpose is a suitable option given that such aircraft is relatively simpler to control, and responds quickly to pilot inputs. The latter is similar to how human pilot students start learning using a light single-engine aircraft. Therefore, the simulated aircraft used for the experiments in this chapter is Cirrus Vision SF50. The experiments are as follows:

3.4.1.1 Autonomous Flying under Calm Weather

The purpose of this experiment is to assess the behaviour of the IAS compared with the behaviour of the human pilot under calm weather conditions.

3.4.1.1.1 Data Collection

In this experiment, the human pilot used the IAS Interface to perform the following in the flight simulator: takeoff, gaining altitude, and maintaining a slower climb rate with a fixed

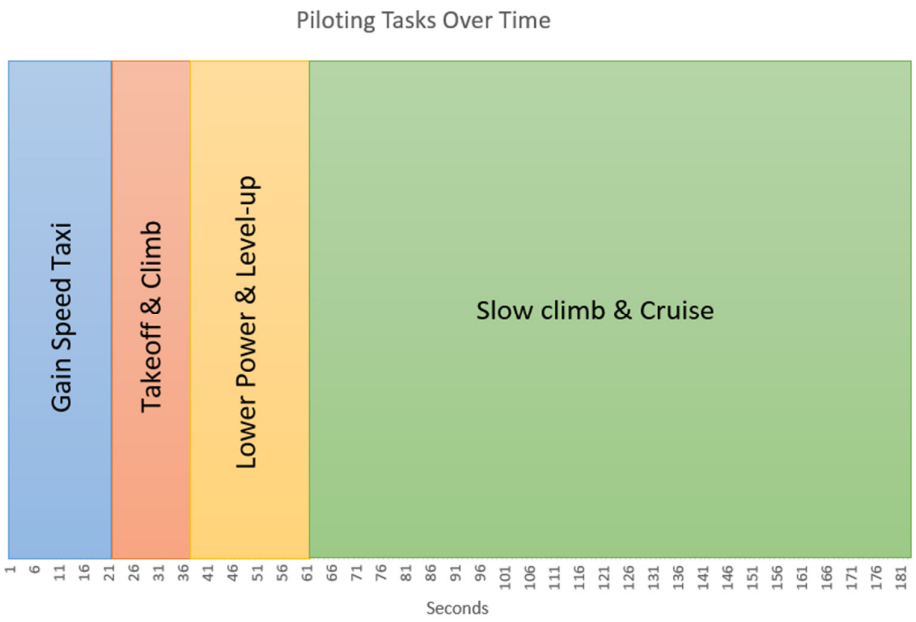

Fig. 3.6. Piloting tasks in performed in the four flight phases over time.

vector, under calm weather with null readings of wind gusts and turbulence. The performed tasks in the initial four flight phases lasted for 182 seconds as Fig. 3.6 shows. While the pilot performed the demonstration, the Interface collected speed and altitude as simulator inputs, throttle, gear, and brakes as pilot outputs, and elevator control data (speed, altitude, pitch as simulator inputs, and elevator as pilot output). Using only one demonstration in calm weather conditions, the Interface stored collected data as training datasets in the database.

An additional data collection process was initiated to capture and compare the aircraft's Automatic Flight Control (AFC)/Autopilot performance with the IAS under calm weather conditions. Due to the AFC's inability to take-off, it was engaged at an altitude of 1600 ftmsl. The AFC was set to climb to an altitude of 6000 ftmsl at a rate of 1500 ftmsl per minute.

3.4.1.1.2 Training

For this experiment, ANN 1 (throttle, gear, and brakes control), and ANN 2 (elevator control) were trained until low Mean Squared Error (MSE) values were achieved (below 0.1). Since single-hidden-layer ANNs are used for training, and since the training datasets are small given that a single demonstration was provided, training requires a short time to be completed (under 10 minutes).

3.4.1.1.3 Autonomous Control

After training the ANNs on the relevant training datasets, the aircraft was reset to the runway in the flight simulator, calm weather conditions were chosen, and the IAS was engaged. ANN 1 (throttle, gear, and brakes control), and ANN 2 (elevator control) operate simultaneously to control the aircraft autonomously. Through the Interface, they receive: 1. continuous flight data from the flight simulator as inputs, and 2. coefficients of models from the database (calm weather throttle, gear, brakes, and elevator control models) to predict and output control commands that are sent to the flight simulator. This process allows the IAS to perform learned tasks: takeoff, gaining altitude, and maintaining a slow climb rate with a fixed vector autonomously. This was repeated 10 times to assess performance consistency.

3.4.1.2 Autonomous Flying under Stormy Weather

The purpose of this experiment is to assess the behaviour of the Intelligent Autopilot compared with the behaviour of the human pilot under stormy weather conditions.

3.4.1.2.1 Data Collection

In this experiment, the human pilot used the IAS Interface to perform the following in the flight simulator: takeoff, gaining altitude, and maintaining a slower climb rate with a fixed vector, under stormy weather. The weather conditions included: wind gusts reaching up to 33 knots, wind directions flowing from all directions (0 to 360 degrees clockwise deviation from north), local turbulence up to 0.19, and rain and hail perception up to 68 mm.

While the pilot performed the demonstration, the Interface collected rudder control and aileron control data, and stored them as two training datasets in the database.

Two demonstrations were required to capture the skill needed to keep the light aircraft on the runway during strong crosswinds using rudders, and only one demonstration of roll stabilization using ailerons was presented to the system. To test the system's ability to generalize well in unseen conditions, no new throttle, gear, brakes, and elevator control data was collected under stormy weather conditions; instead, the data collected for these controls in Experiment 1 were reused. This aims to test the ability of the models generated under calm weather conditions to generalize in the unseen stormy weather conditions.

During taxi speed gain on the runway, the human pilot attempted multiple heading corrections using the rudder to stay on the runway while strong crosswinds pushed the aircraft right and left. After take-off, the human pilot constantly corrected the roll deviation by controlling the ailerons.

An additional data collection process was initiated to capture and compare the aircraft's AFC performance with the IAS under stormy weather conditions with the same settings used in experiment 1. It should be mentioned that the AFC disengaged itself multiple times while flying through the storm which made it difficult to capture a complete demonstration, especially when the strong winds affected the aircraft's stability and caused it to stall.

3.4.1.2.2 Training

For this experiment, ANN 3 (rudder control), and ANN 4 (aileron control) were trained until low Mean Squared Error (MSE) values were achieved (below 0.1). Preliminary empirical testing showed that for this task, to generate the desired behaviour, the model must be trained until the MSE value is slightly below 0.1 compared with values larger than 0.1. Since single-hidden-layer ANNs are used for training, and since the training datasets are small given that few demonstration were provided, training requires a short time to be completed (under 10 minutes).

3.4.1.2.3 Autonomous Control

After training the ANNs on the relevant training datasets, the aircraft was reset to the runway in the flight simulator, stormy weather conditions were chosen, and the IAS was engaged. ANN 1 (throttle, gear, and brakes control), ANN 2 (elevator control), ANN 3 (aileron control), and ANN 4 (rudder control) operate simultaneously to control the aircraft autonomously. Through the Interface, they receive: 1. continuous flight data from the flight simulator as inputs, and 2. coefficients of models from the database (calm weather throttle, gear, brakes, and elevator control models, and stormy weather rudder and aileron control models) to predict and output control commands that are sent to the flight simulator. This process allows the IAS to perform learned tasks: takeoff, gaining altitude, and maintaining a slow climb rate with a fixed vector autonomously, while continuously correcting the aircraft's heading and roll. This was repeated 10 times to assess performance consistency.

3.4.2 Results of Experiments on Prototype 1

The following section describes the results of the conducted tests. The 10 attempts by IAS to fly autonomously in each experiment (calm and stormy weather) were averaged and compared with the performance of the human pilot and the aircraft's AFC using the statistical methods Mean Absolute Error (MAE) and Mean Absolute Deviation (MAD). MAE is used to measure the error in consistency when performing a specific task by the IAS, while MAD is used to analyse the difference between the performance of the IAS and the human pilot or the AFC by measuring and comparing the variability in data. In addition, Behaviour Charts are

used to compare the behaviour deviation between the continuous variables representing the actions performed by the human pilot and the IAS.

3.4.2.1 Experiment 1 (Calm Weather Condition)

Two models were generated with the following MSE values as table 3.2 shows. Table 3.3 lists the accuracy assessment results by comparing the behaviour of IAS with the behaviour of the human pilot in the calm weather experiment. Table 3.4 lists the accuracy assessment results by comparing the behaviour of IAS with the behaviour of the aircraft's AFC in the calm weather experiment.

Fig. 3.7, 3.8, and 3.9 illustrate the Intelligent Autopilot's control commands compared with the human pilot. Fig. 3.10 and 3.11 illustrate altitude and speed over time comparisons between the human pilot, the Intelligent Autopilot System, and the aircraft's AFC.

TABLE 3.2
THE RESULTING MEAN SQUARED ERROR VALUES OF THE MODELS GENERATED AFTER TRAINING THE RELEVANT ANNS UNDER CALM WEATHER

ANN Model	MSE	Weather Condition
Throttle, gear, and brakes model	0.03	Calm
Elevation control model	0.09	Calm

TABLE 3.3
IAS ACCURACY ASSESSMENT RESULTS IN CALM WEATHER. MAE IS USED TO MEASURE THE ERROR BETWEEN THE OUTPUTS OF THE IAS AND THE HUMAN PILOT WHEN PERFORMING FIVE DIFFERENT TASKS. MAD GIVES THE VARIABILITY OR STATISTICAL DISPERSION FOR THE IAS AND THE HUMAN PILOT.

Command Control/Flight Status	Weather Condition	(MAE)	(MAD) - Human	(MAD) - AI
Throttle Command	Calm	0.016	0.1	0.1
Gear Command	Calm	0.016	0.3	0.3
Elevation Command	Calm	0.007	0.022	0.022
Altitude	Calm	91.694	1204.779	1251.816
Speed	Calm	6.148	23.031	24.925

TABLE 3.4
IAS ACCURACY ASSESSMENT RESULTS IN CALM WEATHER. MAE IS USED TO MEASURE THE ERROR BETWEEN THE OUTPUTS OF THE IAS AND THE AIRCRAFT'S AFC WHEN PERFORMING TWO DIFFERENT TASKS. MAD GIVES THE VARIABILITY OR STATISTICAL DISPERSION FOR THE IAS AND THE AIRCRAFT'S AFC.

Command Control/Flight Status	Weather Condition	(MAE)	(MAD) - AFC	(MAD) - AI
Altitude	Calm	200.096	814.382	833.118
Speed	Calm	17.308	1.098	4.785

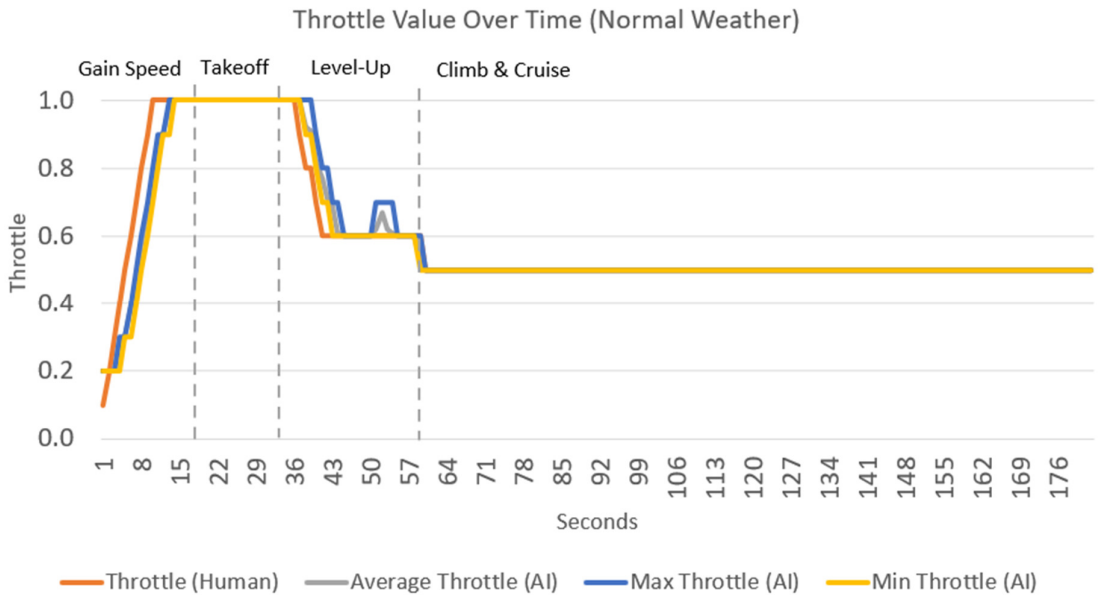

Fig. 3.7. (Exp. 1) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum throttle commands over time during the four flight phases –separated by dotted lines- as illustrated in Fig. 3.6. Throttle value 1 refers to 100% throttle.

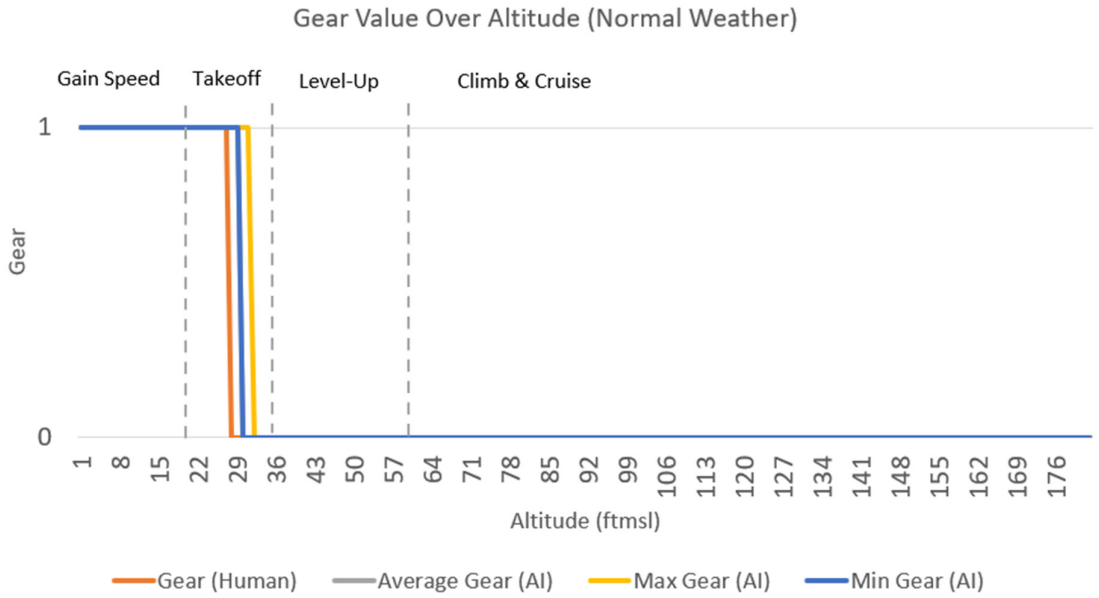

Fig. 3.8. (Exp. 1) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum gear commands over time. Gear value 1 refers to deployed gear, and 0 refers to retracted gear.

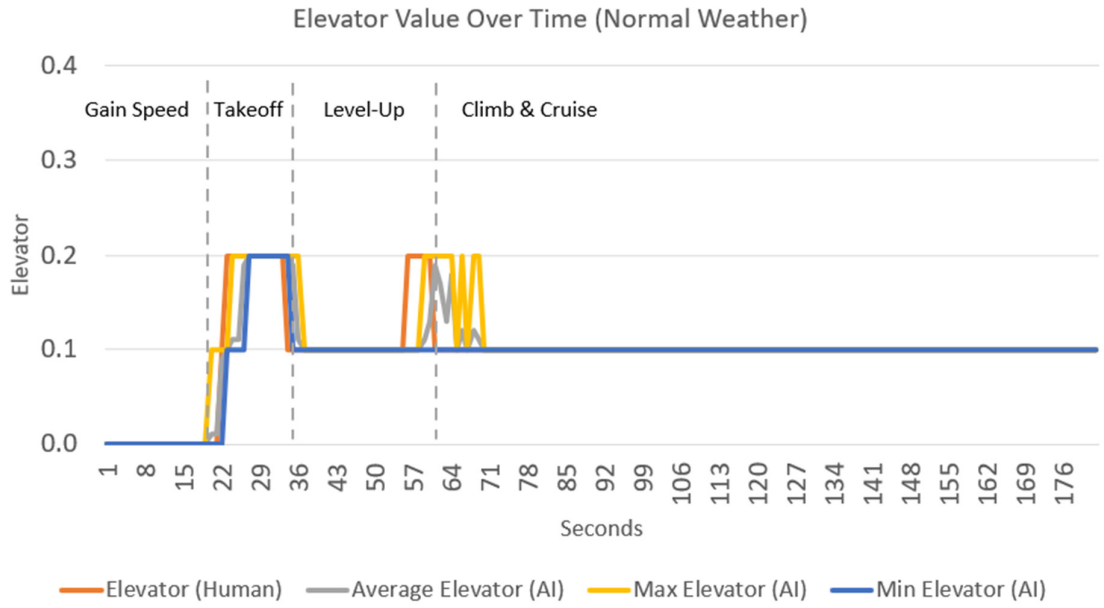

Fig. 3.9 (Exp. 1) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum elevator commands over time. An elevator value of 0.2 refers to 20 degrees deflection of the elevators.

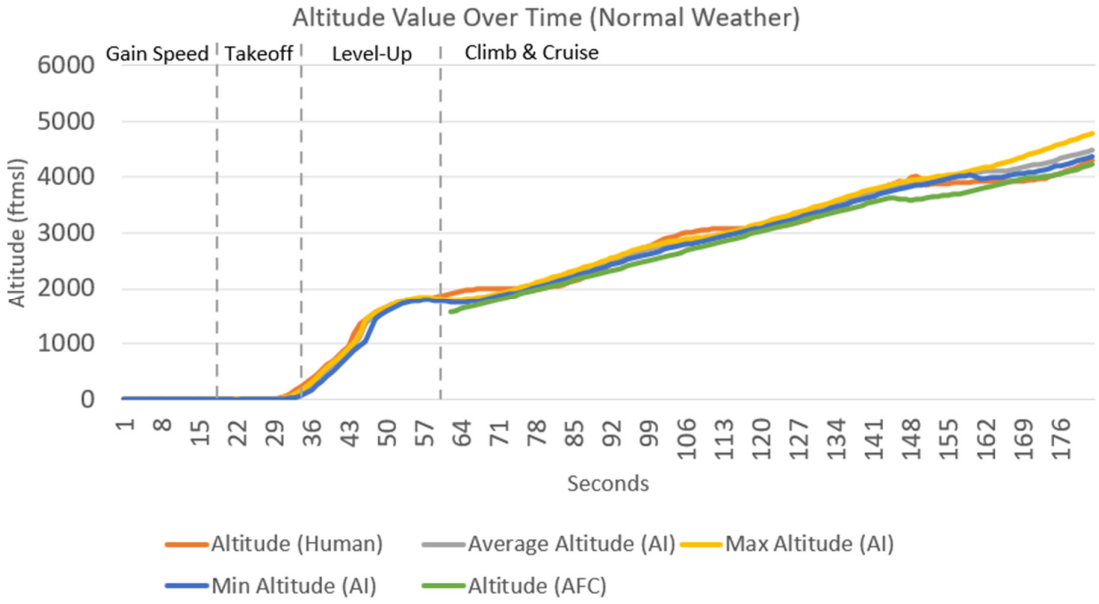

Fig. 3.10. (Exp. 1) A comparison between the human pilot, the aircraft's AFC/Autopilot, and the Intelligent Autopilot's average, maximum, and minimum altitude over time.

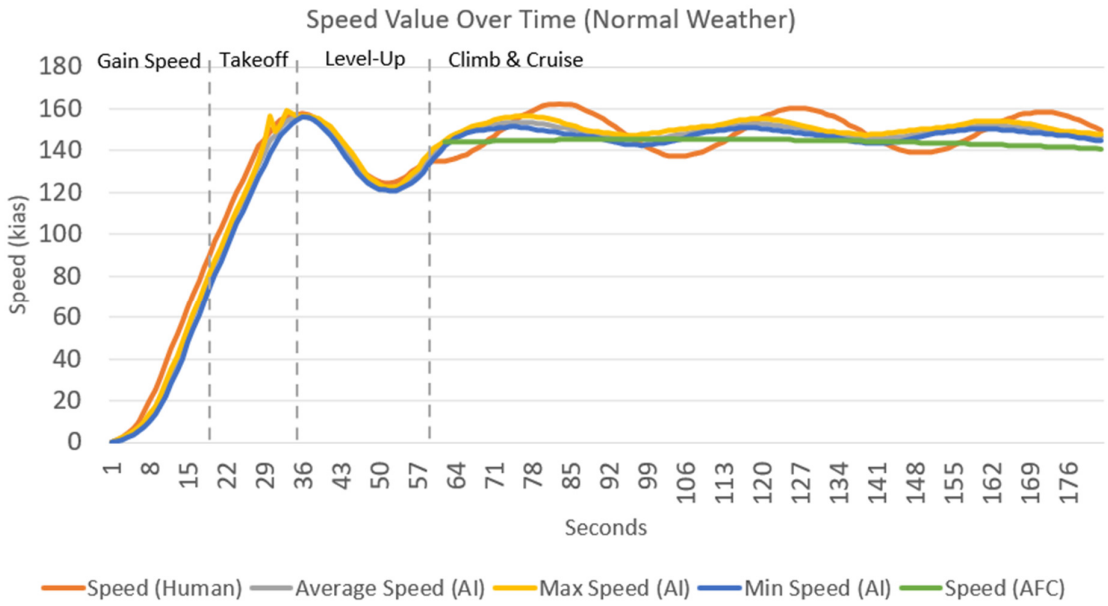

Fig. 3.11. (Exp. 1) A comparison between the human pilot, the aircraft's AFC/Autopilot, and the Intelligent Autopilot's average, maximum, and minimum speed over time.

Fig. 3.12 shows the autonomously controlled aircraft immediately after take-off, while Fig. 3.13 shows the ANNs' outputs of the IAS where throttle is set to full power, and the elevators are set to a deflection of 0.2 degrees.

Fig. 3.12. The autonomously controlled aircraft immediately after take-off.

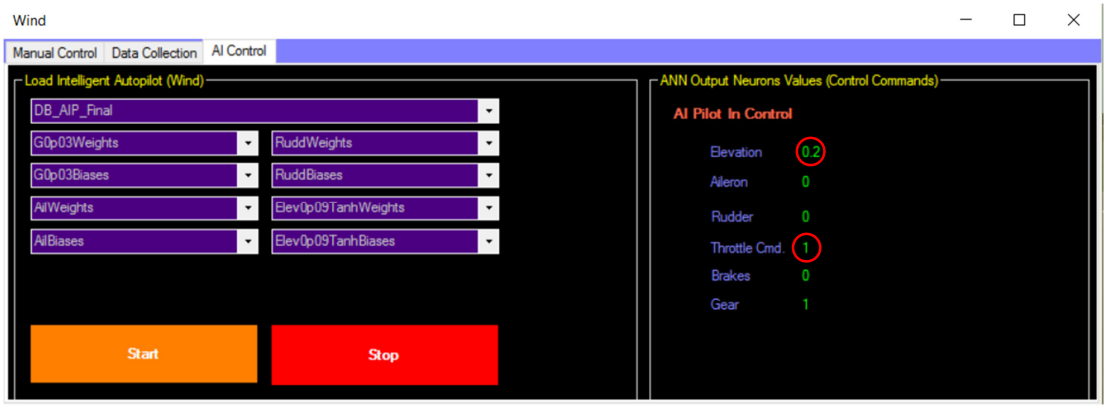

Fig. 3.13. The right part of the IAS shows the ANN's outputs or control commands applied to the aircraft at the flight moment shown in Fig. 3.12.

Fig. 3.14 shows the autonomously controlled aircraft shortly after take-off, while Fig. 3.15 shows the ANNs' outputs of the IAS causing the gear to retract.

Fig. 3.14. The autonomously controlled aircraft shortly after take-off with the gear being retracted.

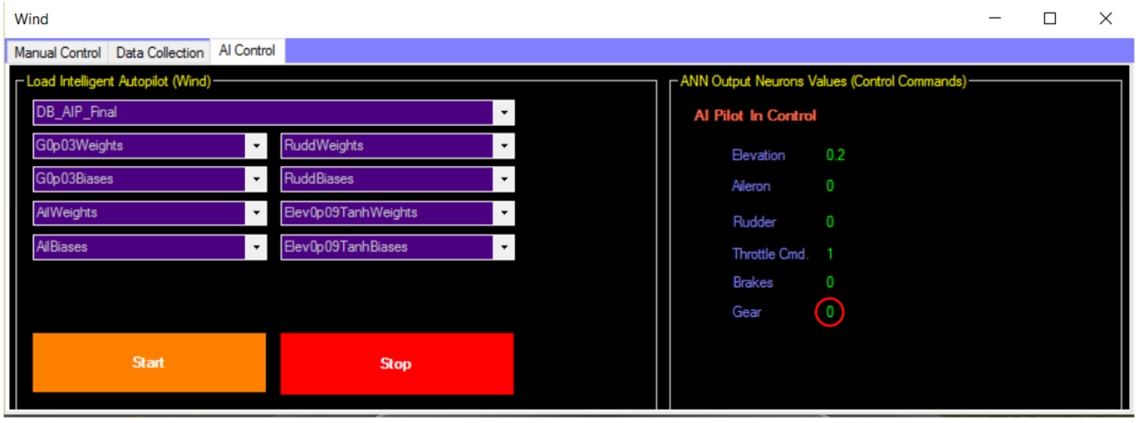

Fig. 3.15. The right part of the IAS shows the ANN's outputs or control commands applied to the aircraft at the flight moment shown in Fig. 3.14. Here, the ANNs sent the command to retract gear.

Fig. 3.16 shows the autonomously controlled aircraft during the smooth ascend and cruise phase, while Fig. 3.17 shows the ANNs' outputs of the IAS suitable for this flight phase.

Fig. 3.16. The autonomously controlled aircraft during the smooth ascend and cruise phase.

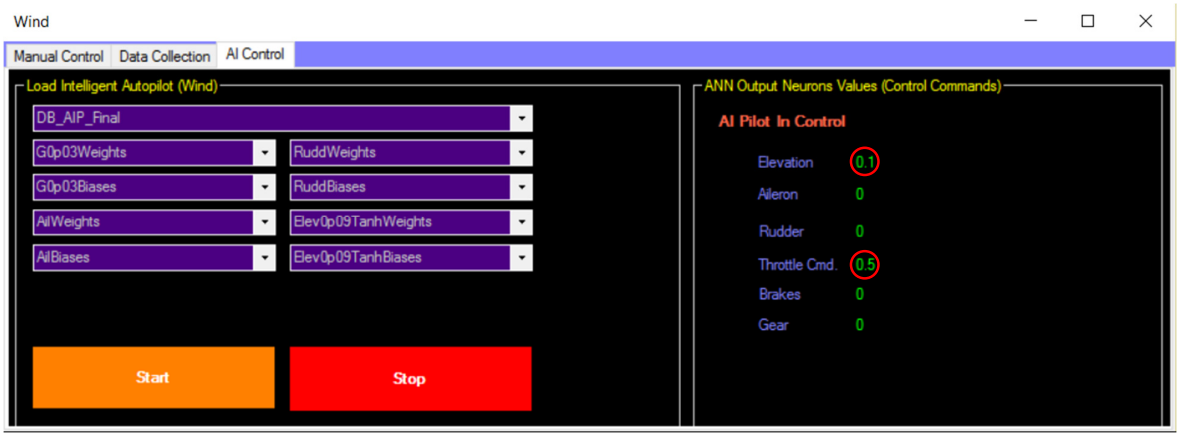

Fig. 3.17. The right part of the IAS shows the ANN's outputs or control commands applied to the aircraft at the flight moment shown in Fig. 3.16. Here, the IAS decreased the throttle and maintained a smooth ascend eventually leading to the cruise phase.

3.4.2.2 Experiment 2 (Stormy Weather Condition)

Two models were generated with MSE values as table 3.5 shows. Table 3.6 lists the accuracy assessment results by comparing the behaviour of IAS with the behaviour of the human pilot in the stormy weather experiment. Table 3.7 lists the accuracy assessment results

TABLE 3.5
THE RESULTING MEAN SQUARED ERROR VALUES OF THE MODELS GENERATED AFTER TRAINING THE RELEVANT ANNS UNDER STORMY WEATHER.

ANN Model	MSE	Weather Condition
Rudder control model	0.009	Storm
Aileron control model	0.07	Storm

TABLE 3.6
IAS ACCURACY ASSESSMENT RESULTS IN STORMY WEATHER. MAE IS USED TO MEASURE THE ERROR BETWEEN THE OUTPUTS OF THE IAS AND THE HUMAN PILOT WHEN PERFORMING SEVEN DIFFERENT TASKS. MAD GIVES THE VARIABILITY OR STATISTICAL DISPERSION FOR THE IAS AND THE HUMAN PILOT.

Command Control/Flight Status	Weather Condition	(MAE)	(MAD) - Human	(MAD) - AI
Throttle Command	Stormy	0.080	0.146	0.122
Gear Command	Stormy	0.063	0.349	0.355
Elevation Command	Stormy	0.017	0.034	0.030
Altitude	Stormy	350.963	1297.031	1153.647
Speed	Stormy	15.757	36.520	41.347
Rudder Command	Stormy	0.041	0.093	0.133
Aileron Command	Stormy	0.006	0.086	0.080

TABLE 3.7
IAS ACCURACY ASSESSMENT RESULTS IN STORMY WEATHER. MAE IS USED TO MEASURE THE ERROR BETWEEN THE OUTPUTS OF THE IAS AND THE AIRCRAFT'S AFC WHEN PERFORMING TWO DIFFERENT TASKS. MAD GIVES THE VARIABILITY OR STATISTICAL DISPERSION FOR THE IAS AND THE AIRCRAFT'S AFC.

Command Control/Flight Status	Weather Condition	(MAE)	(MAD) - AFC	(MAD) - AI
Altitude	Stormy	316.952	804.572	783.771
Speed	Stormy	12.322	3.349	4.212

by comparing the behaviour of IAS with the behaviour of the aircraft's AFC in the stormy weather experiment.

Fig. 3.18, 3.19, and 3.20 illustrate the IAS control commands compared with the human pilot in the stormy weather experiment. Fig. 3.21 and 3.22 illustrate altitude and speed over time comparisons between the human pilot, the IAS, and the aircraft's AFC in the stormy weather experiment. Fig. 3.23 generated from sample heading/rudder data, illustrates a comparison between the human pilot and IAS heading correction attempts using the rudder. Fig. 3.24 generated from sample roll/aileron data illustrates the comparison between the human pilot and the IAS roll correction attempts using the ailerons.

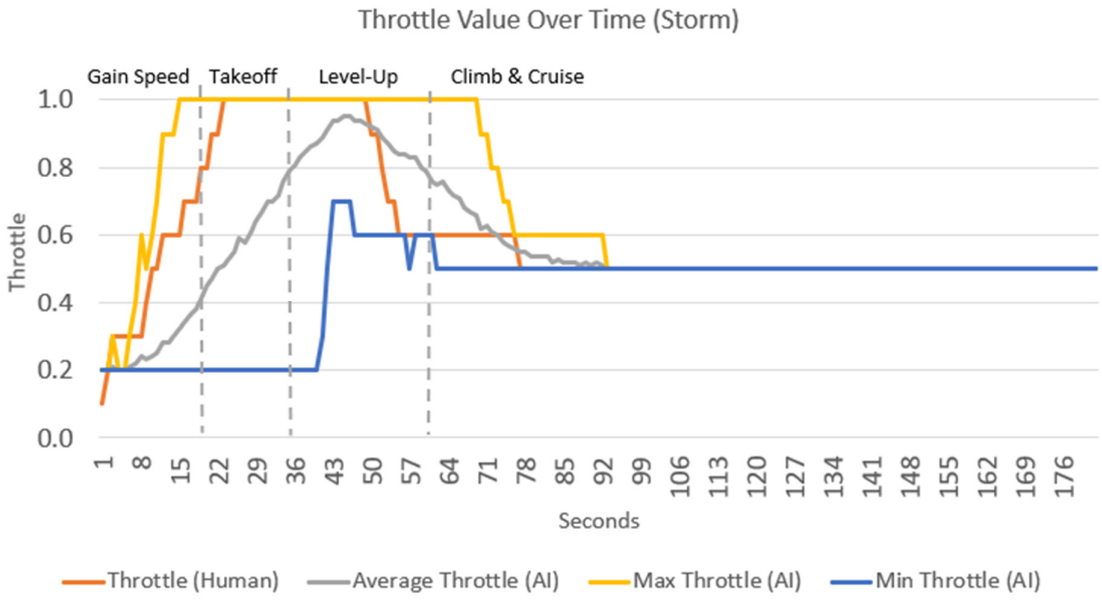

Fig. 3.18. (Exp. 2) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum throttle commands over time during the four flight phases –separated by dotted lines- as illustrated in Fig. 3.6. Throttle value 1 refers to 100% throttle.

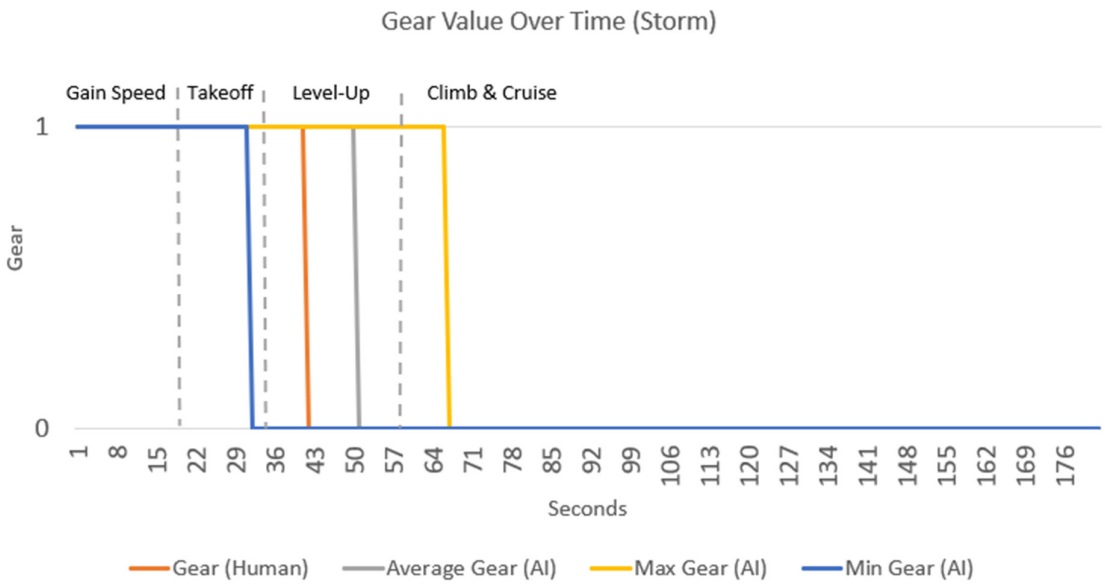

Fig. 3.19. (Exp. 2) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum gear commands over time. Gear value 1 refers to deployed gear, and 0 refers to retracted gear.

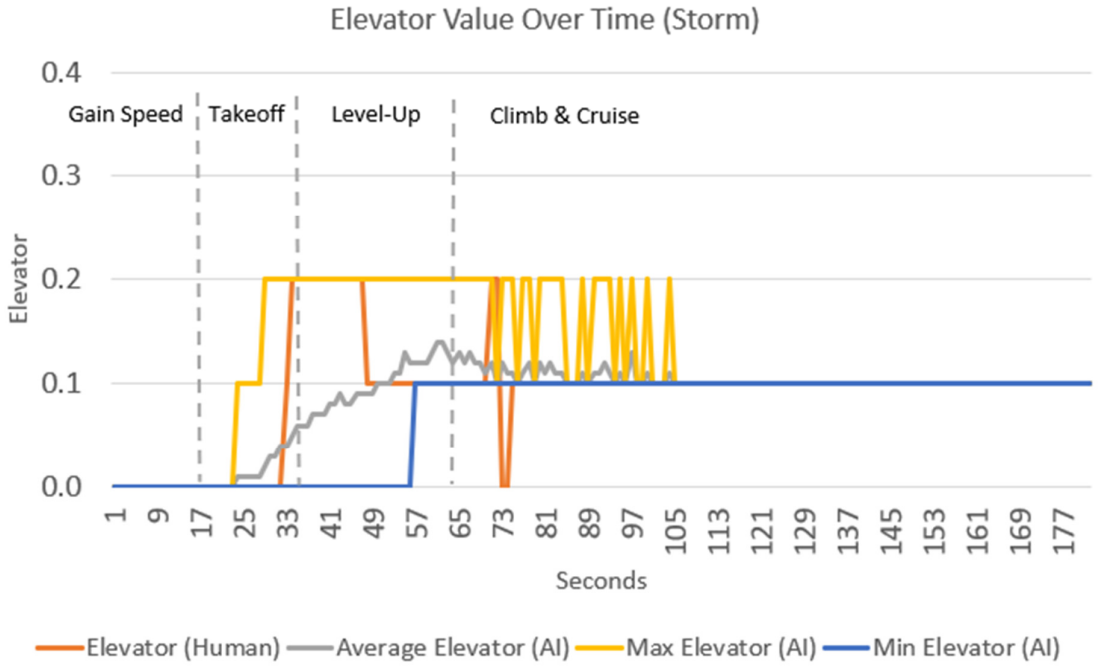

Fig. 3.20. (Exp. 2) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum elevator commands over time. An elevator value of 0.2 refers to 20 degrees deflection of the elevators.

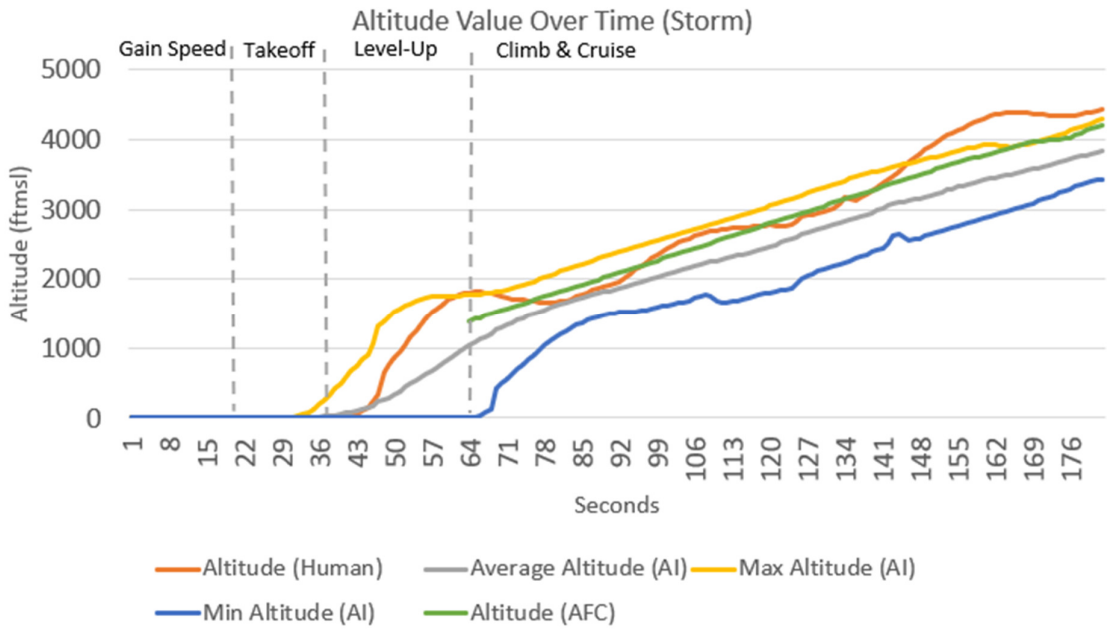

Fig. 3.21. (Exp. 2) A comparison between the human pilot, the aircraft's AFC/Autopilot, and the Intelligent Autopilot's average, maximum, and minimum altitude over time.

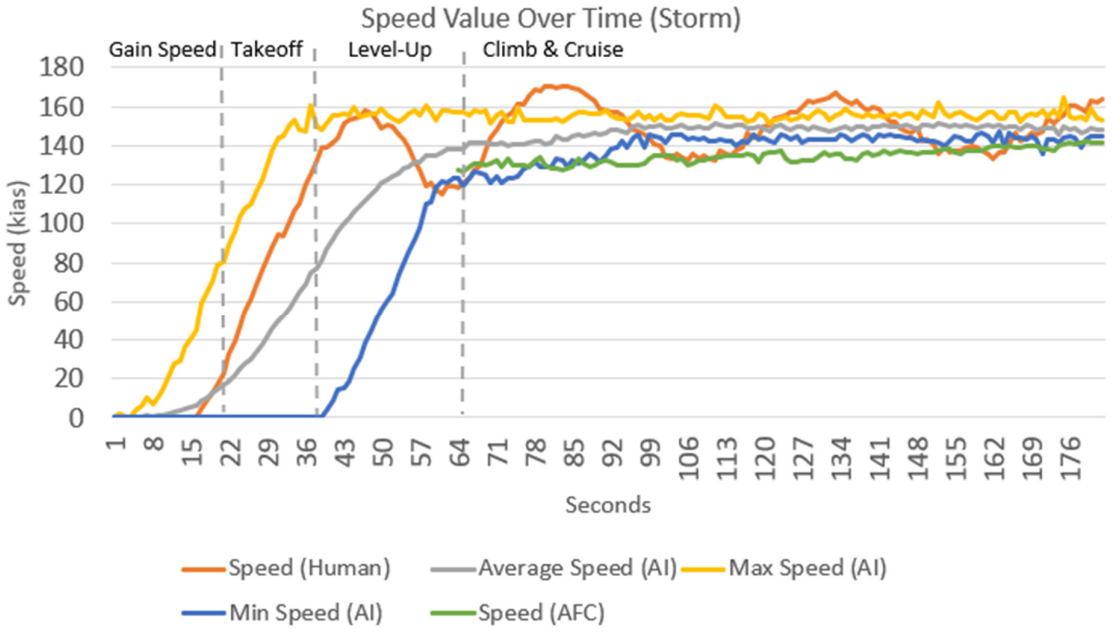

Fig. 3.22. (Exp. 2) A comparison between the human pilot, the aircraft's AFC/Autopilot, and the Intelligent Autopilot's average, maximum, and minimum speed over time.

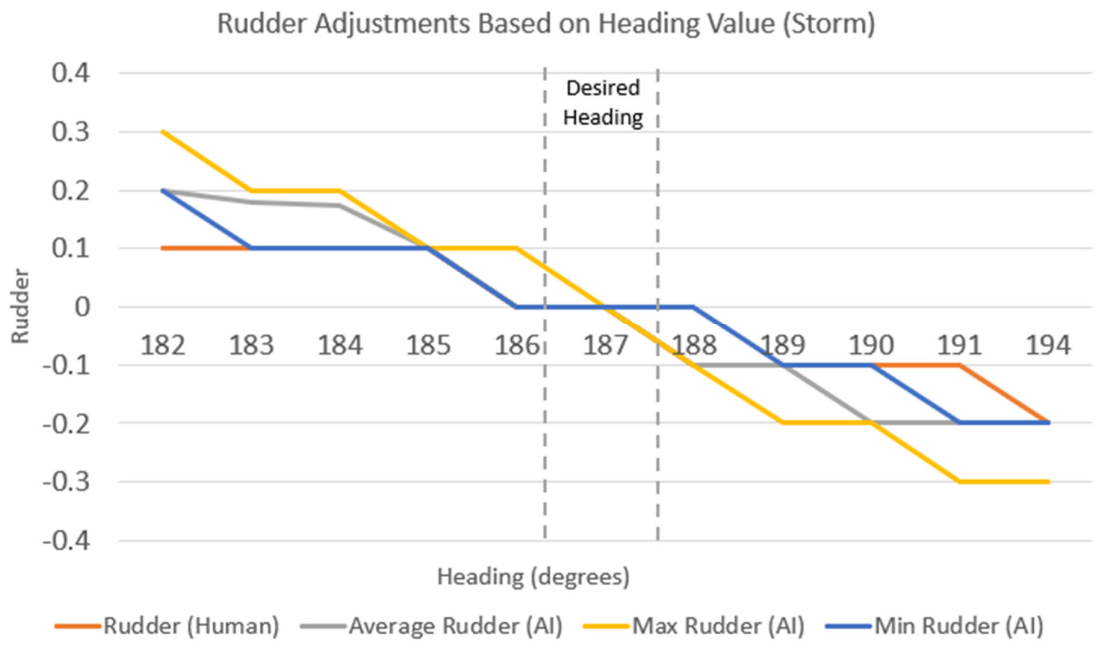

Fig. 3.23. (Exp. 2) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum heading correction attempts. The middle part between the two dotted lines is the area where no corrections are required (based on a heading of 187 degrees). The right part illustrates a deviation in heading towards the right, while the left part illustrates a deviation in heading towards the left. A rudder value of 0.2 refers to 20 degrees deflection of the rudder.

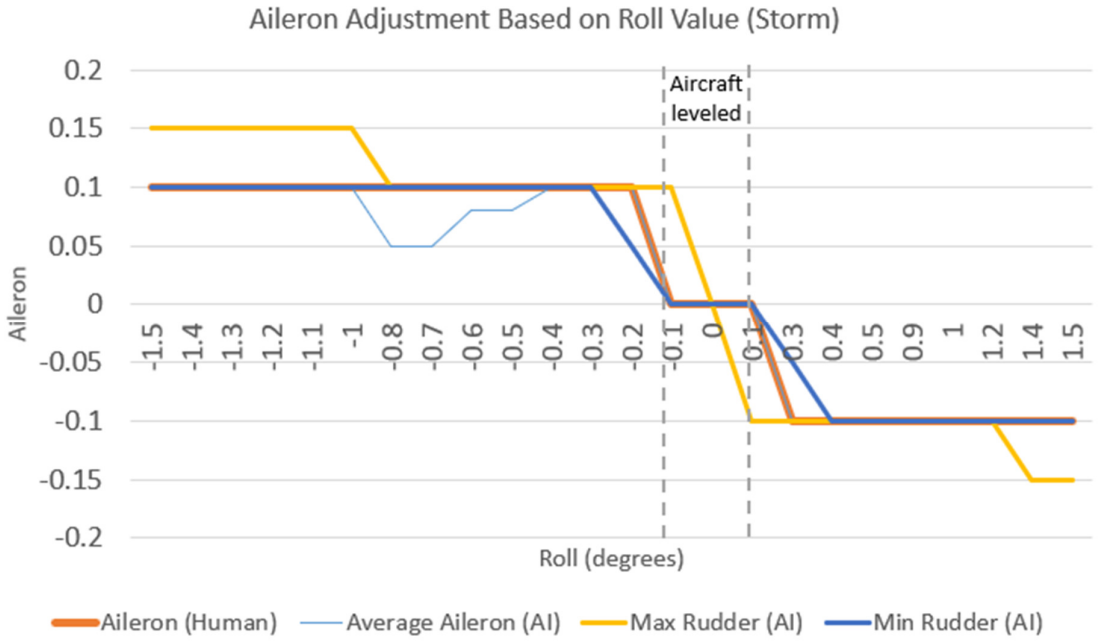

Fig. 3.24. (Exp. 2) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum roll correction attempts. The middle part between the two dotted lines is the area where no corrections are required. The right part illustrates a deviation in roll towards the right, while the left part illustrates a deviation in roll towards the left. An aileron value of 0.1 refers to 10 degrees deflection of the ailerons.

Fig. 3.25 shows the autonomously controlled aircraft during taxi (speed gain) before take-off. The aircraft is affected by strong crosswind blowing from the right side, which pushed the aircraft to the left of the runway's centre. Fig. 3.26 shows the ANNs' outputs of the IAS where rudders are controlled by the system to correct the aircraft's heading deviation.

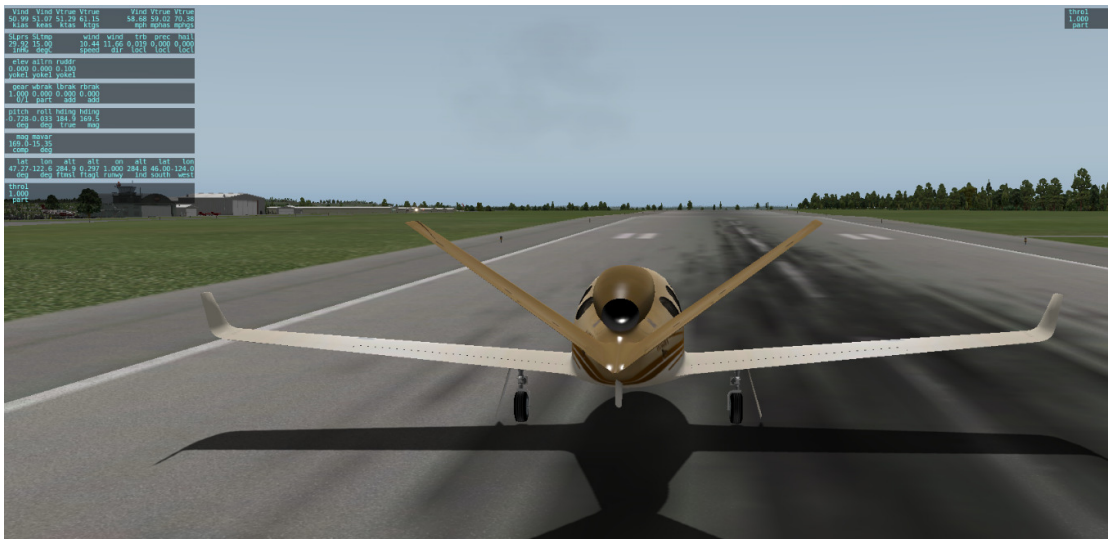

Fig. 3.25. The autonomously controlled aircraft during taxi (speed gain) before take-off. Here, crosswind blowing from the right side caused the aircraft to deviate from the centre of the runway to the left.

Fig. 3.26. The right part of the IAS shows the ANN's outputs or control commands applied to the aircraft at the flight moment shown in Fig. 3.25. Here, the IAS set the rudders to 0.1 degrees (turn right) to counter the effect of crosswind blowing from the right side.

Fig. 3.27 shows the autonomously controlled aircraft after take-off where the aircraft's roll is affected by the strong winds, while Fig. 3.28 shows the ANNs' outputs of the IAS where ailerons are controlled by the system to correct the aircraft's roll deviation.

Fig. 3.27. The autonomously controlled aircraft after take-off where the aircraft's roll is affected by the strong winds causing it to roll left.

Fig. 3.28. The right part of the IAS shows the ANN's outputs or control commands applied to the aircraft at the flight moment shown in Fig. 3.23. Here, the IAS set the ailerons to 0.05 degrees (roll right) to counter the effect of strong winds causing the aircraft to roll left.

3.4.2 Analysis

As can be seen in Figs 3.7 to 3.11, experiment 1 (calm weather condition) presented very desirable results. The IAS was capable of imitating the human pilot's actions and behaviour. The latter is supported by the results presented in tables 3.3 and 3.4 which show a difference of no more than 4% and 3% consecutively in the values of the Mean Absolute Deviation (MAD) when comparing the IAS with the human pilot and the aircraft's AFC as well. In addition, the small Mean Absolute Error (MAE) values in tables 3.3 and 3.4 show the consistency of the performance of the IAS when manipulating the different controls. However, the altitude MAE value is relatively high which indicates a need to enhance the performance of the IAS when maintaining altitude.

As can be seen in Figs 3.18 to 3.24, experiment 2 (stormy weather condition) showed the ability of IAS to imitate rapid stabilization actions, and generalize well in unseen conditions. The system used the calm weather models to fly in stormy conditions gracefully. The latter is supported by the results presented in tables 3.6 and 3.7 which show a difference of no more than 10% and 3% consecutively in the values of the Mean Absolute Deviation (MAD) when comparing the IAS with the human pilot and the aircraft's AFC as well. In addition, the small Mean Absolute Error (MAE) values in tables 3.6 and 3.7 show the consistency of the performance of the IAS when manipulating the different controls. However, the altitude MAE value is relatively high which also indicates a need to enhance the performance of the IAS when maintaining altitude.

The system was able to imitate multiple human pilot's skills and behaviour after being presented with very limited examples (1 example for throttle, gear, and brakes, 1 example for elevator control, 1 example for aileron control, and 2 examples for rudder control). The results show that the Intelligent Autopilot System continued to stabilize the aircraft in difficult weather condition as Fig. 3.24 shows, while the AFC of the simulated aircraft disengaged itself multiple times.

It should be mentioned that given his lack of flying experience, the human pilot found it difficult to provide stable demonstrations as shown by the oscillations, but despite receiving this data as training, the IAS learned to fly smoothly - indeed smoother than the human pilot as can be seen in Figs 3.11 and 3.22.

The complete learning process starting from the demonstration of the specific task by the human pilot, and ending with the automatic generation of the learning model takes less than 20 minutes.

Informal trials were also performed with the IAS in which the aircraft was put into a variety of situations that it had not been trained to handle (e.g., a stall, inversion, etc.). In all cases the IAS was able to stabilize the aircraft safely on its own.

The results prove that the first objective of using Artificial Neural Networks to learn low-level and high-level piloting skills and abilities from training datasets representing demonstrations given by human teachers, was achieved.

However, at this point, the IAS is not capable of handling emergencies such as engine problems, emergency landing, etc. It is also not capable of performing complete flights which must include navigating, and landing.

3.5 Summary

To summarize, the objective of proving the ability of Artificial Neural Networks to learn basic piloting tasks was achieved through developing multiple ANNs each designed and trained to handle a specific control problem by generating models from training datasets that captured demonstrations performed by a human teacher in a flight simulator. The proposed training methodology which relies on capturing the demonstrations by the interface of the IAS and storing them as training datasets in the database for offline learning proved to be efficient and effective. In addition, processing the data before training by scaling the inputs and outputs without having to normalize them, and applying the Sigmoid activation function for datasets with positive values, and the Hyperbolic Tangent function for datasets containing negative values generated excellent models with low Mean Squared Error values. The generated models were able to capture low-level and high-level skills and abilities that enabled the IAS to perform basic flights under calm and severe weather conditions. This provides evidence to support the hypothesis of this work aimed towards proving the possibility to teach a flight control system piloting skills.

4. PROTOTYPE 2 (HANDLING EMERGENCIES)

After the first objective of using Artificial Neural Networks to learn basic piloting tasks from training datasets representing demonstrations given by human teachers was achieved, the purpose of the second prototype was to achieve the objective of teaching ANNs complex piloting tasks. The tasks represent the ability to handle multiple emergency situations including Rejected Takeoff (RTO), engine failure and fire, and emergency landing, in addition to maintaining a desired altitude. Performing these tasks represent new abilities that the first prototype of the IAS did not have. Since this work aims to teach autopilots of large jets (mainly airliners) how to perform different piloting tasks, and since the work in the previous chapter (3. Prototype 1 (Methodology & Basic Flying)) proved the possibility of teaching an autopilot of a light aircraft how to perform piloting tasks, the work in this chapter presents the transition from a light aircraft to an airliner in the flight simulator.

The work in this chapter was published in the 2016 IEEE Symposium Series on Computational Intelligence (SSCI), Athens, Greece. (Appendix B).

To achieve the second objective mentioned above, the IAS should imitate the behaviour of the human pilot when handling emergency situations including Rejected Takeoff (RTO), engine(s) failure or fire while airborne, emergency landing, and maintaining cruise altitude. The previously gained skills from prototype 1 which allowed the IAS to perform autonomous basic flights such as takeoff and cruise along with the components developed to achieve those skills remain without change. In this chapter, new skills should be learned, and their relevant components must be developed to equip the IAS with the capability to handle a difficult problem within this work's scope of autonomous flying, which is the ability to handle emergency situations. Each emergency situation should be considered as a segregated problem for which specific components must be developed to allow the IAS to manipulate a number of controls such as reverse thrust, fuel valve, and fire extinguishing system to handle the different emergency scenarios. In addition, it is required to include a new component which should act as managing or supervising program that can detect the different emergency situations, and based on that, activate certain components suitable for the given situation.

To achieve the desired behaviours mentioned in the paragraph above, the IAS should learn four new tasks. Out of the four tasks, three are related to emergency situations (handling rejected takeoff, emergency landing, and single-engine failure or fire). The reason for choosing

these specific emergency situations is the ability of the flight simulator X-Plane to simulate them. X-Plane can also simulate other emergency situations such as electric failures, hydraulic failures, etc., however, such emergency situations require understanding how the electric and hydraulic systems of the aircraft operate, and how to handle their potential faults which do not fall under the scope of this work. The fourth task that the IAS should learn is the ability to maintain a given cruise altitude since prototype 1 was not capable of handling such important task. New ANNs were then assigned each task.

The approach of breaking down control problems by isolating each control task, identifying the aircraft's control interfaces or surfaces that are used to handle each task, and designing and training dedicated ANNs for each task is followed in this chapter since this approach proved to yield good results as the experiments conducted on the first prototype showed, which is in line with the literature (*2.1.2.3 Multiple Artificial Neural Networks & Sensors*).

The problem of handling rejected takeoff was segregated as a subtask of the main task of handling emergencies. The control interfaces of the aircraft that are used to handle this subtask are the brakes, throttle, and reverse thrust. In addition to applying the brakes, the throttle is increased to full, and reverse thrust is applied to project thrust in the opposite direction to bring the aircraft to a full stop. Therefore, a new ANN should be designed and trained to manipulate these control interfaces. To achieve this, the ANN should be trained to take speed and engine status as inputs, and based on these inputs, generate control commands to the brakes, throttle, and reverse thrust as outputs.

The second emergency subtask is handling emergency landings when all engines fail while airborne. In this case, the aircraft should maintain a positive pitch to continue gliding while speed and altitude decrease gradually until the point of impact on the ground to avoid a nose-first scenario. The control surfaces of the aircraft that are used to handle this subtask are the elevators. Since the elevators are used to control the aircraft's degree of pitch to insure a smooth emergency landing, a new ANN should be designed and trained to manipulate the elevators. To achieve this, the ANN should be trained to take pitch as an input, and generate control command to the elevators as output.

The third emergency subtask is handling engine(s) fire or single-engine failure. The control interfaces of the aircraft that are used to handle the fire situation subtask are the fire extinguisher, throttle, and fuel valve. As soon as fire is detected, the fire extinguisher should be used simultaneously with the fuel valve control interface to extinguish the fire and cut the

supply of fuel. In addition, the throttle is increased to full to burn the fuel left in the engine(s) on fire. Therefore, a new ANN should be designed and trained to manipulate these control interfaces. To achieve this, the ANN should be trained to take fire sensor reading as input, and generate control commands to the fire extinguisher, fuel valve, and throttle as outputs. Regarding single-engine failure while airborne, no additional ANNs were designed since the aircraft can continue flying although one engine has failed, therefore, the same ANNs from prototype 1 (chapter 3) are used to continue flying.

The final task is maintaining a given cruise altitude. According to the literature [1], the engine(s) thrust can be used to alter the altitude of the aircraft by increasing the throttle which increases lift and causes the aircraft to climb, or by decreasing the throttle which decreases lift and causes the aircraft to sink. In addition, the elevators can be used to maintain a horizontal pitch degree during climb and sink. In this case, the control interface and surfaces of the aircraft that are used to maintain altitude are the throttle and the elevators, therefore, two new ANNs should be designed and trained to take altitude as input, and generate control command to the throttle as output, and take pitch as input, and generate control command to the elevators as output.

In addition, ANN 1 from the first prototype is now called the Taxi Speed Gain ANN, and it is now dedicated for the taxi speed gain/ground run phase only. This is achieved by training the ANN to handle just one flight phase (taxi speed gain/ground run) instead of all, which represents a further problem break-down that not only breaks down control tasks, but also, follows the break-down of flight phases (as Fig. 3.6 from Chapter 3 shows) and assigns the relevant ANNs to each phase instead of all. The Taxi Speed Gain ANN is modified by removing the unnecessary altitude input which is not needed anymore since no altitude maintenance is required during the taxi speed gain/ground run phase. The same applies to the gear output which is removed since it is needed during the takeoff phase when the gear is retracted. ANN 2 from the second prototype, which was used during all flight phases to control pitch, is now called the Takeoff ANN, and it is only used during the takeoff phase to achieve the initial positive climb. This ANN is modified by removing the unnecessary speed input since full throttle is applied during the takeoff phase. In addition, gear and throttle outputs are added to the elevators output of the Takeoff ANN since they are needed during the takeoff phase.

Nine feedforward Artificial Neural Networks now comprise the core of the IAS. The additional six ANNs were designed based on the same approach mentioned in section (*3.2*

Training). Each ANN is designed and trained to handle specific controls and tasks. The ANNs are: Taxi Speed Gain ANN which controls brakes and throttle during the ground-run phase, Takeoff ANN which controls gear, elevators, and throttle during takeoff, Rejected Takeoff ANN which controls brakes, throttle, and reverse thrust to reject/abort takeoff if necessary, Aileron ANN (from prototype 1), Rudder ANN (from prototype 1), Cruise Altitude ANN which maintains altitude by controlling throttle, Cruise Pitch ANN which maintains levelled pitch by controlling the elevators, Fire Situation ANN which activates the fire extinguisher, closes fuel valves, and controls throttle in case of engine fire, and Emergency Landing Pitch ANN which maintains a given pitch when gliding during emergency landing. The inputs and outputs which represent the gathered data and relevant actions, and the topologies of the ten ANNs are illustrated in Fig. 4.1.

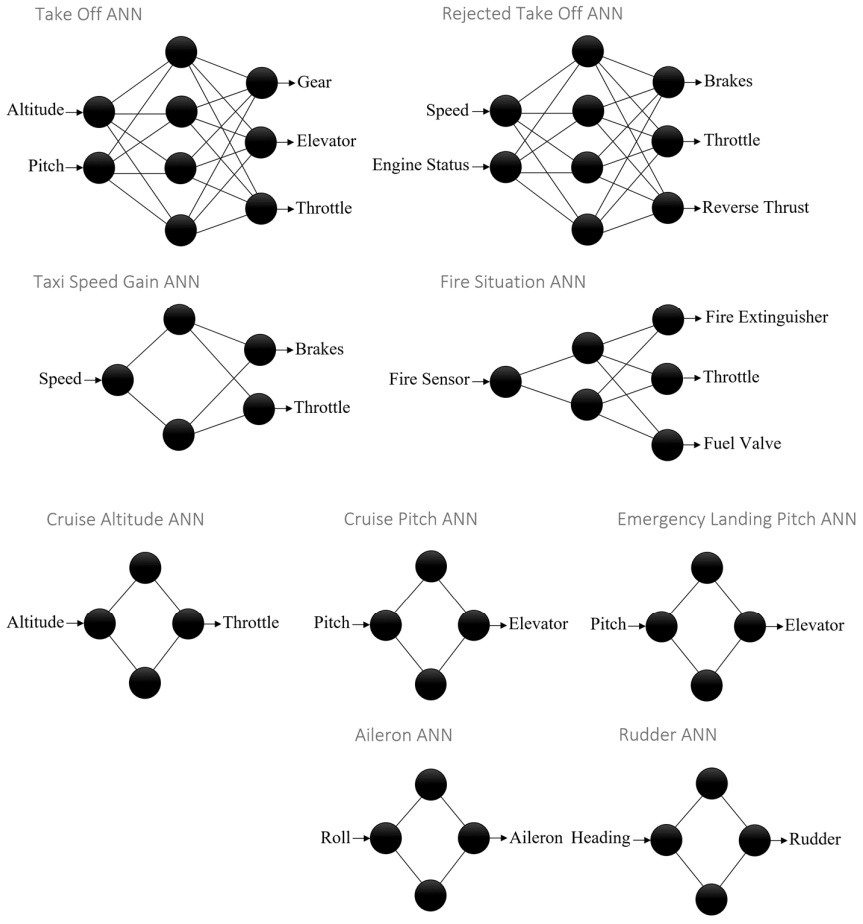

Fig. 4.1. Inputs, outputs, and the topologies of the ten ANNs representing the core of the Intelligent Autopilot System. Each ANN is designed and trained to handle a specific task.

Following the approach of breaking down the problem and flight phases by designing ANNs that are only used during specific flight phases, requires a method to control these ANNs. This method should be able to detect the current flight phase, and based on it, perform certain tasks. To achieve this, a straightforward method should be developed since designing a comprehensive flight management system does not fall under the scope of this work because it would be infeasible for one researcher to achieve during a PhD and would distract from the hypothesis. The method should perform continuous checks to prove certain conditional statements to be true or false. Based on the latter (true or false), certain tasks should be performed. For example, if the aircraft is on the runway before takeoff, the method should be able to detect that the current flight phase is taxi speed gain/ground run by returning "true" for this flight phase, and returning "false" for the remaining flight phases, then, activating the Taxi Speed Gain ANN, and deactivating the other ANNs that are relevant to the remaining flight phases. This method was tested to ensure its robustness by exposing it to the different flight phases in the flight simulator, and ensuring it can detect the correct flight phase, and activate the relevant ANNs. To achieve this, the Flight Manager program was designed and added to the Intelligent Autopilot System.

The Flight Manager is a program which resembles a Behaviour Tree [146]. The purpose of the Flight Manager is to manage the ten ANNs of the IAS by deciding which ANNs are to be used simultaneously at each moment given the flight phase. The Flight Manager starts by receiving flight data from the flight simulator through the interface of the IAS, detects the flight condition and phase by examining the received flight data, and decides which ANNs are required to be used given the flight condition (normal/emergency/fire situation) and phase (taxi speed gain/takeoff/cruise/emergency landing). Fig. 4.2 illustrates the process which the Flight Manager follows. The Flight Manager receives flight data including engine(s) readings, and fire status which are used to detect emergencies. In addition, the Flight Manager uses speed, and altitude data to decide when to switch from the taxi speed gain phase to the takeoff phase, and from the latter to the cruise phase. The thresholds of engine readings, speed, and altitude which dictate a switch between phases or conditions can be changed to suite multiple types of aircraft, weights, etc. The thresholds used for the work in this chapter were chosen based on empirical testing that was conducted to identify the thresholds which are suitable for the aircraft used in this chapter as table 4.1 shows. However, these thresholds are corrected in chapter 7 with the assistance of an experienced pilot, and here the precise values are secondary to the main aim of developing a working system. The Flight Manager program was iteratively

developed using literature on appropriate procedures and flight phases where necessary and improved via preliminary experiments. Based on understanding what the flight phases are and how transitions from one flight phase to the next should be carried out, and based on understanding what procedures are carried out when the selected emergency situations happen, the Flight Manager program performs continuous checks to prove the conditional statements (true or false) that dictate the transitions to the next flight phase, or the transition from a normal flight to handling an emergency by applying the appropriate procedures. The procedures that the Flight Manager program applies are represented by activating the appropriate ANNs for each flight phase or emergency situation, and stopping the other ANNs that are suitable for the other phases or situations as Fig. 4.2 shows.

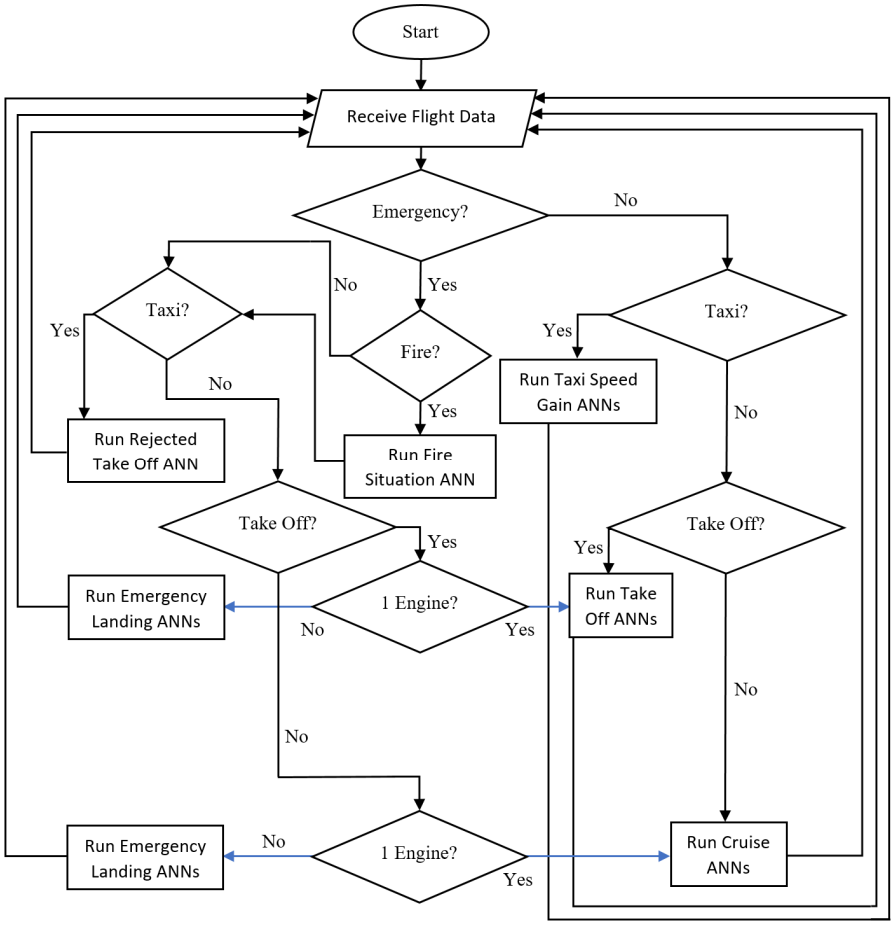

Fig. 4.2. A Flowchart illustrating the process which the Flight Manager program follows to decided which ANNs are to be used. The Rudder ANN is constantly active during the ground-run/taxi speed gain phase, the Cruise Pitch, and Cruise Altitude ANNs are constantly active during the cruise phase, and the Aileron ANN is constantly active during the takeoff and cruise phases.

TABLE 4.1
THE DIFFERENT THRESHOLDS TYPES & VALUES USED BY THE FLIGHT MANAGER PROGRAM TO DETECT THE FLIGHT PHASE OR CONDITION, AND APPLY THE APPROPRIATE DECISION

Threshold Type and Value	Detected Phase/Condition and Decision
If both engines RPM is below 1000 rpm	All engines failure – Rejected Takeoff/Emergency Landing
If there is an RPM difference between engines of 10 rpm or more	Single engine failure - Rejected Takeoff/Emergency Landing
If engine(s) forces reading is below 5000 lb	Single or all engine(s) failure - Rejected Takeoff/Emergency Landing
If speed during taxi speed gain/ground run is equal to or above 130 knots	Transition from taxi speed gain/ground run to takeoff
If altitude is equal to or above 4000 ft.	Transition from takeoff to cruise

4.1 Experiments on Prototype 2

Here, the new approach is to segment the training dataset of taxi speed gain, takeoff, and climb into three different sets that are handled separately by three ANNs (Taxi Speed Gain ANN, Takeoff ANN, and Cruise ANN) instead of just one ANN. Prototype 2 also introduces four new ANNs in order to learn flight emergency procedures for the first time.

In order to assess the effectiveness of the new approach, the Intelligent Autopilot System was tested in four experiments:

1. Rejected takeoff

2. Emergency landing

3. Maintaining a cruising altitude

4. Handling single-engine failure/fire while airborne.

Each experiment is composed of 20 attempts by the IAS to perform autonomously under the given conditions.

The human pilot who provided the demonstrations is the author. The simulated aircraft used for the experiments is a Boeing 777 since unlike the simple process of flight which is relatively universal between aircraft, the procedures for emergency procedures may differ among different types of aircraft, and since this work is ultimately aimed at airliners, it now makes sense to transition to this more complex aircraft.

4.1.1 Rejecting Takeoff

The purpose of this experiment is to assess the behaviour of the IAS compared with the behaviour of the human pilot when a Rejected Takeoff (RTO) is required.

4.1.1.1 Data Collection

In this experiment, the human pilot used the IAS Interface to perform the following in the flight simulator: reject takeoff when one engine fails or catches fire, and when two engines fail or catch fire (one demonstration for each scenario). The flight simulator was set to simulate the failure or fire conditions for one or two engines immediately after the user presses a hot key on the keyboard. Rejecting takeoff is performed by going to full reverse thrust and engaging brakes. In case of fire, the human pilot turned off the fuel valve, turned on the fire extinguishing system, and went to full throttle to burn the fuel left in the engine(s). While the pilot performed the demonstration, the Interface collected speed and engine status as inputs, and brakes, throttle, and reverse thrust control data as outputs. The Interface stored the collected data in the database as the training dataset for the Rejected Takeoff ANN. The Interface also collected fire sensor readings as input, and fire extinguisher, throttle, and fuel valve control data as outputs. The Interface stored the collected data in the database as the training dataset for the Fire Situation ANN.

4.1.1.2 Training

For this experiment, the Rejected Takeoff ANN, and the Fire Situation ANN were trained until low Mean Squared Error (MSE) values were achieved (below 0.001). Since single-hidden-layer ANNs are used for training, and since the training datasets are small given that a single demonstration was provided, training requires a short time to be completed (under 10 minutes).

4.1.1.3 Autonomous Control

After training the ANNs on the relevant training datasets, the aircraft was reset to the runway in the flight simulator to test autonomous RTO multiple times under different scenarios (one and two engine(s) failure and fire), the simulator was set to simulate the desired emergency scenario, and the IAS was engaged. When the flight manager detects the emergency, it stops the Taxi Speed Gain ANN, and runs the Rejected Takeoff ANN and the Fire Situation ANN simultaneously to reject takeoff and handle fire autonomously. Through the Interface, ANNs receive: 1. Relevant flight data from the flight simulator as inputs, and 2. Coefficients of the relevant models from the database to predict and output control commands that are sent to the flight simulator. This process allows the IAS to autonomously perform the learned task:

rejecting takeoff if necessary. This was repeated 20 times for each scenario to assess performance consistency.

4.1.2 Emergency Landing

The purpose of this experiment is to assess the behaviour of the IAS compared with the behaviour of the human pilot when a forced or emergency landing is required.

4.1.2.1 Data Collection

In this experiment, the human pilot used the IAS Interface to perform the following in the flight simulator: emergency landing when two engines fail or catch fire (one demonstration for each scenario). The flight simulator was set to simulate the failure or fire conditions for two engines immediately after the user presses a hot key on the keyboard. Emergency landing is performed by maintaining a controlled glide using the elevators to ensure a gradual loss of speed and altitude without stalling the aircraft by maintaining a slight positive pitch. If there is any power left in the engines, the throttle is used to aid the gliding phase. In case of fire, the human pilot turned off the fuel valve, and turned on the fire extinguishing system. In this scenario going to full throttle to burn the fuel left in the engines is not possible since both engines do not have sufficient power. While the pilot performed the demonstration, the Interface collected pitch as input, and elevator control data as output. The Interface stored the collected data in the database as the training dataset for the Emergency Landing Pitch ANN. The Interface also collected altitude as input, and throttle control data as output. The Interface stored the collected data in the database as the training dataset for the Emergency Landing Altitude ANN.

4.1.2.2 Training

For this experiment, the Emergency Landing Pitch ANN, and the Emergency Landing Altitude ANN were trained until low Mean Squared Error (MSE) values were achieved (below 0.001 for the Emergency Landing Pitch ANN and below 0.2 for the Emergency Landing Altitude ANN). Since single-hidden-layer ANNs are used for training, and since the training datasets are small given that a single demonstration was provided, training requires a short time to be completed (under 10 minutes).

4.1.2.3 Autonomous Control

After training the ANNs on the relevant training datasets, the aircraft was reset to the runway in the flight simulator to test autonomous emergency landing multiple times under different scenarios (both engines failure or fire), the simulator was set to simulate the desired emergency

scenario, and the IAS was engaged. After the IAS took the aircraft airborne, and when the flight manager detects the emergency, it stops the Takeoff ANN (during climb), or the cruise ANNs, and runs the Emergency Landing Pitch ANN, and the Emergency Landing Altitude ANN simultaneously to maintain a controlled glide while descending to the ground. Through the Interface, the ANNs receive: 1. Relevant flight data from the flight simulator as inputs, and 2. Coefficients of the relevant models from the database to predict and output control commands that are sent to the flight simulator. This process allows the IAS to autonomously perform learned task: emergency landing by maintaining a controlled glide. This was repeated 20 times for each scenario to assess performance consistency.

4.1.3 Maintaining a Cruising Altitude

The purpose of this experiment is to assess the behaviour of the IAS compared with the behaviour of the human pilot while maintaining a desired cruising altitude.

4.1.3.1 Data Collection

In this experiment, the human pilot used the IAS Interface to maintain a cruising altitude in the flight simulator by increasing and decreasing the throttle, and by using the elevator to maintain a fairly levelled pitch (one demonstration). While the pilot performed the demonstration, the Interface collected altitude as input, and throttle control data as output. The Interface stored the collected data in the database as the training dataset for the Cruise Altitude ANN. The Interface also collected pitch as input, and elevator control data as output. The Interface stored the collected data in the database as the training dataset for the Cruise Pitch ANN.

4.1.3.2 Training

For this experiment, the Cruise Altitude ANN, and the Cruise Pitch ANN were trained until low Mean Squared Error (MSE) values were achieved (below 0.02 and 0.001 respectively). Since single-hidden-layer ANNs are used for training, and since the training datasets are small given that a single demonstration was provided, training requires a short time to be completed (under 10 minutes).

4.1.3.3 Autonomous Control

After training the ANNs on the relevant training datasets, the aircraft was reset to the runway in the flight simulator to test the ability of maintaining a desired cruise altitude autonomously, and the IAS was engaged. After the IAS took the aircraft airborne, continued to climb, and reached the proximity of the desired altitude, the system's ability to maintain the given altitude

was observed. Through the Interface, the ANNs receive: 1. Relevant flight data from the flight simulator as inputs, and 2. Coefficients of the relevant models from the database to predict and output control commands that are sent to the flight simulator. This process allows the IAS to autonomously perform learned task: maintain a desired cruising altitude. This was repeated 20 times for each scenario to assess performance consistency.

4.1.4 Handling Single-Engine Failure/Fire while Airborne

The purpose of this experiment is to assess the behaviour of the IAS in case of an engine failure or fire while airborne.

4.1.4.1 Data Collection

In this experiment, the human pilot did not provide an explicit demonstration for the single-engine failure. Instead, it was intended to test the already trained ANNs, and determine whether their models are able to generalize well in this new scenario where the failed engine creates a drag, and forces the aircraft to descend, and creates a yaw deviation towards the failed engine's side.

4.1.4.2 Training

For this experiment, the previously trained models of the Cruise Altitude ANN, the Cruise Pitch ANN, and the rudder ANN from prototype 1 were used.

4.1.4.3 Autonomous Control

After setting the simulator to simulate the desired emergency scenario (single-engine failure or fire), and after the IAS took the aircraft airborne, when the flight manager detects the emergency, it continues to use the same ANNs (Takeoff ANN, or cruise ANNs), and runs the Fire Situation ANN if fire is detected, to fly autonomously using the power left from the engine that operates normally. Through the Interface, the ANNs receive: 1. Relevant flight data from the flight simulator as inputs, and 2. Coefficients of the relevant models from the database to predict and output control commands that are sent to the flight simulator. This was repeated 20 times for each scenario to assess performance consistency. Throughout all the experiments, the Rudder and Aileron ANNs from prototype 1 are used normally during the different phases.

4.2 Results of Experiments on Prototype 2

The following section describes the results of the conducted tests. The 20 attempts by the IAS to handle each scenario autonomously were averaged and compared with the performance of the human pilot when applicable.

4.2.1 Experiment 1 (Rejecting Takeoff)

Two models were generated with the MSE values as table 4.2 shows. Fig. 4.3 illustrates the behaviour of the IAS when controlling the transition of flight modes under normal conditions, while Fig. 4.4 illustrates the behaviour of the IAS when engine(s) failure or fire is detected and a Rejected Takeoff (RTO) is performed. The results of the 20 experiments showed consistency by following the correct procedure in each experiment with a 100% accuracy rate.

TABLE 4.2
THE RESULTING MEAN SQUARED ERROR VALUES OF THE MODELS GENERATED AFTER TRAINING THE RTO & FIRE SITUATION ANNS

ANN	MSE
Rejected Takeoff ANN	0.000999
Fire Situation ANN	0.000999

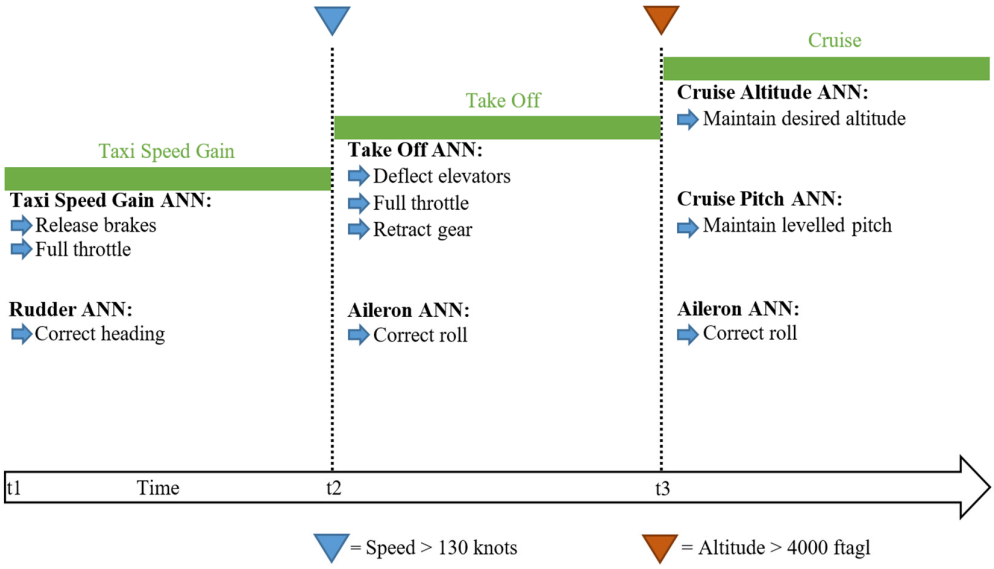

Fig. 4.3. 20 observations of the behaviour of the IAS when controlling the transition of flight modes under normal conditions. The IAS was able to execute the transitions between flight phases and activate the relevant ANNs at the desired transition points (t2, t3) with an accuracy of 100%.

Fig. 4.4. (Rejected Takeoff experiment) 20 observations of the behaviour of the IAS when engine(s) failure or fire is detected and a Rejected Takeoff (RTO) is performed. The IAS was able to activate the relevant ANNs at the desired transition point (t2) with an accuracy of 100%. The Fire Situation ANN was activated only when fire is detected.

4.2.2 Experiment 2 (Emergency Landing)

Two models were generated with the MSE values as table 4.3 shows. Fig. 4.5 and 4.6 illustrate a comparison between the human pilot and the IAS while maintaining a positive pitch during emergency landing, and their altitude (sink rate). The pitch Mean Absolute Deviation (MAD) results (0.024 for the IAS and 0.196 for the human pilot) show less deviation and a steady behaviour of the IAS due to the good model fit as can be seen in Fig. 4.5. Fig. 4.7 illustrates the behaviour of the IAS when both engines failure or fire is detected and a forced or emergency landing is performed. The results of the 20 experiments showed consistency by following the correct procedure in each experiment with a 100% accuracy rate. Fig. 4.8 shows the G-Load factor before and upon impact during ten different emergency landings performed by the IAS, and one manually induced crash landing. The G-Load factor is the ratio of the lift of a given aircraft to its weight which provides a measure of the stress to which the aircraft structure is subjected, and its unit is referred to as g [1]. The flight simulator X-Plane measures the G-Force effect on the aircraft's frame. It can be set to display an alert message to the user when the safe threshold is exceeded which means the aircraft's frame was wrecked. When the latter happens, the simulation session ends. For this experiment, X-Plane was set to simulate full G-Load effects on the aircraft.

TABLE 4.3
THE RESULTING MEAN SQUARED ERROR VALUES OF THE MODELS GENERATED AFTER TRAINING THE EMERGENCY LANDING ANNS

ANN	MSE
Emergency Landing Pitch ANN	0.000997
Emergency Landing Altitude ANN	0.196117

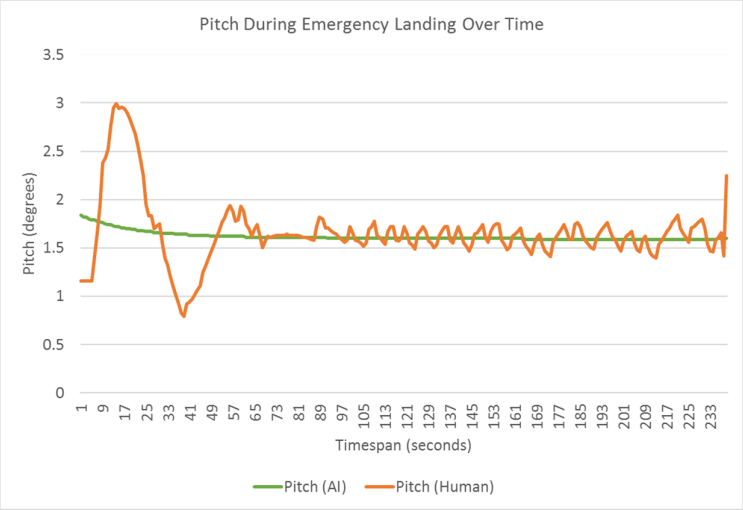

Fig. 4.5. (Emergency landing experiment) A comparison between the human pilot and the Intelligent Autopilot System's pitch during emergency landing. Since the human pilot demonstrator struggled to maintain a desired positive pitch due to lack of experience, the good fit of the generated model resulted in a steady performance which is better than the human teacher.

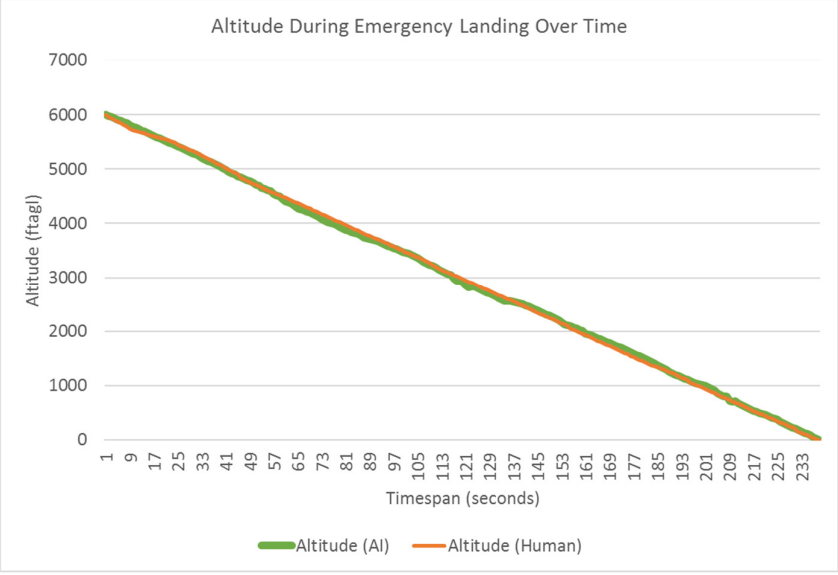

Fig. 4.6. (Emergency landing experiment) A comparison between the human pilot and the Intelligent Autopilot System's altitude during emergency landing. The results show a close sink rate of about 1500 ftagl per minute.

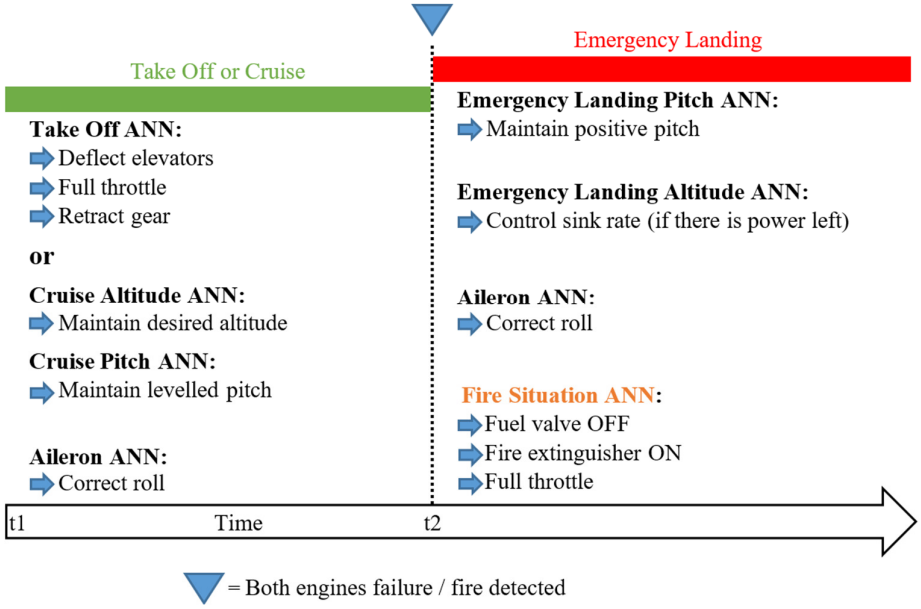

Fig. 4.7. (Emergency landing experiment) 20 observations of the behaviour of the IAS when both engines failure or fire is detected during either take off or cruise, and an emergency landing is performed. The IAS was able to activate the relevant ANNs at the desired transition point (t2) with an accuracy of 100%. The Fire Situation ANN is used only when fire is detected.

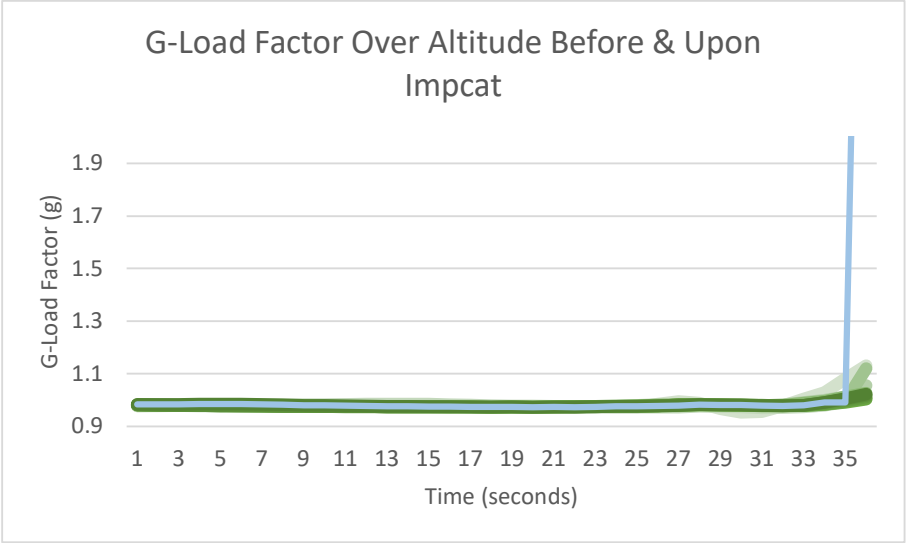

Fig. 4.8. (Emergency landing experiment) The G-Load factor that the aircraft experienced just before impact (airborne) and upon ground impact. The different shades of green represent the G-Load factor during 10 different emergency/crash landings performed by the IAS. The blue line represents the G-Load factor resulting from a manual emergency/crash landing performed by the author.

4.2.3 Experiment 3 (Maintaining a Cruise Altitude)

Two models were generated with the MSE values as table 4.4 shows. Fig. 4.9 and 4.10 illustrate a comparison between the human pilot and the IAS while maintaining a desired cruising altitude. The altitude Mean Absolute Deviation (MAD) results (85.8 for the IAS and 204.58 for the human pilot) shows less deviation of altitude and a steady behaviour of the IAS due to the good model fit as can be seen in Fig. 4.9.

TABLE 4.4
THE RESULTING MEAN SQUARED ERROR VALUES OF THE MODELS GENERATED AFTER TRAINING THE CRUISE ANNS

ANN	MSE
Cruise Altitude ANN	0.017574
Cruise Pitch ANN	0.000835

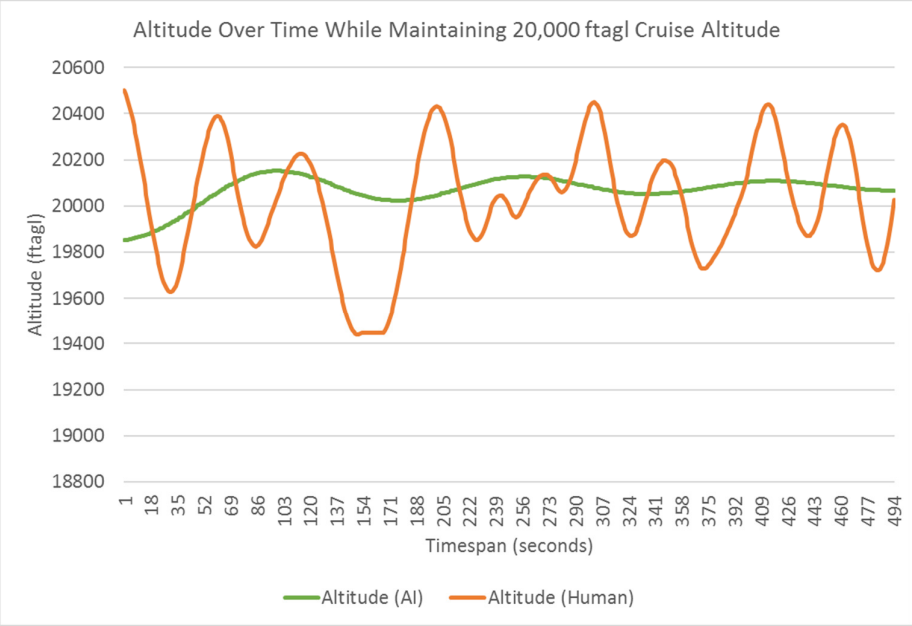

Fig. 4.9. (Maintaining a cruise altitude experiment) A comparison between the human pilot and the Intelligent Autopilot System's altitude during cruising. While the human pilot demonstrator struggled to maintain a desired cruise altitude of 20,000 ftagl, the IAS performed better due to the good fit of the generated model.

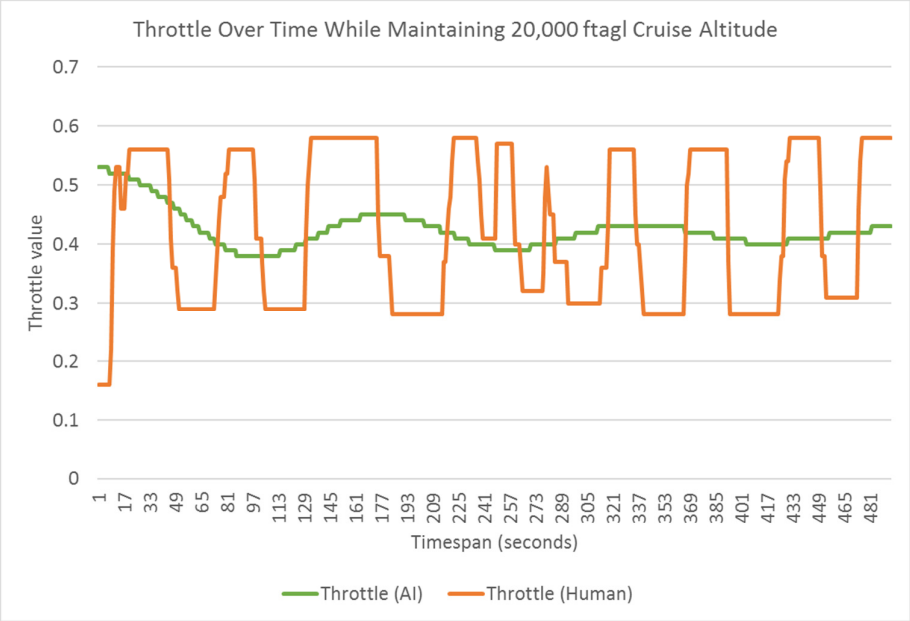

Fig. 4.10. (Maintaining a cruise altitude experiment) The IAS manipulation of throttle to maintain a desired cruise altitude of 20,000 ftagl compared with the human pilot. The IAS manipulated the throttle smoothly compared with the human pilot due to the good fit of the generated model.

4.2.4 Experiment 4 (Handling Single-Engine Failure/Fire while Airborne)

As mentioned above (*4.1.4.1 Data Collection*), the human pilot did not provide an explicit demonstration for the single-engine failure scenario. Instead, it was intended to test the already trained ANNs, and determine whether their models are able to generalize well in this new scenario's experiment. Fig. 4.11 illustrates the behaviour of the IAS when a single-engine fails or catches fire during takeoff or cruise. The system was intended to carry on flying, apply the rudder ANN from prototype 1, and run the Fire Situation ANN in case of fire. The results of the 20 experiments showed consistency by following the correct procedure in each experiment 100% of the time. Fig. 4.12 illustrates how the IAS continues to fly while losing altitude gradually compared with the aircraft's autopilot under the same situation.

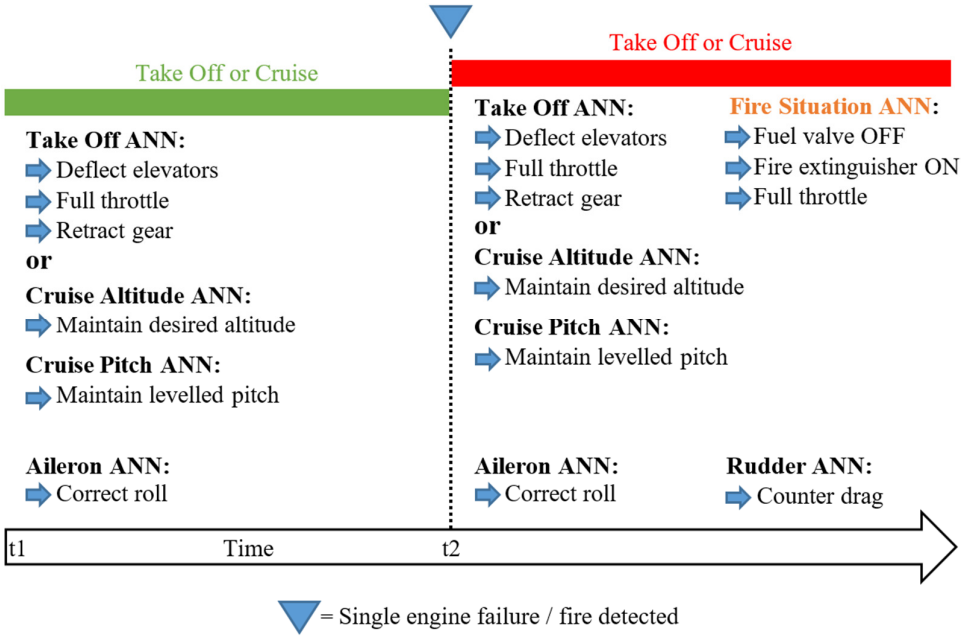

Fig. 4.11. (Handling single-engine failure/fire experiment) 20 observations of the behaviour of the IAS when a single engine failure or fire is detected during either take off or cruise. The IAS was able to activate the relevant ANNs at the desired transition point (t2) with an accuracy of 100%. The ANNs used during Take Off or Cruise perform the same tasks as Fig. 4.3 shows, while the Aileron ANN continues to correct roll. The Fire Situation ANN is used only when fire is detected.

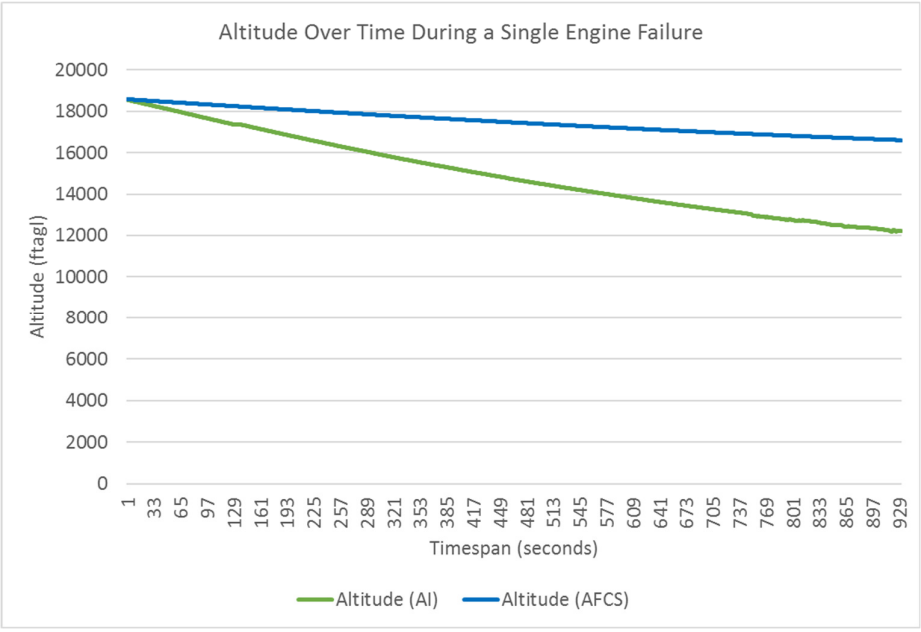

Fig. 4.12. (Handling single-engine failure/fire experiment) Comparing the altitude loss rate of the IAS and the aircraft's AFCS. Since the AFCS is not aware of the single engine failure situation, it compensates by increasing the throttle aggressively, which results in a smaller altitude loss rate, but might put excessive stress on the single operating engine.

4.3 Analysis

Fig. 4.4 shows the rejected takeoff experiment where the IAS was capable of imitating the human pilot's actions and behaviour with an accuracy of 100% by following the correct procedure in each experiment accurately. The resulted ability of the Flight Manager program to handle the transitions from each flight phase to the next, and the ability to detect and handle the emergency situations mentioned in this chapter with an accuracy of 100% proved the suitability of the straightforward design approach which relies on conditional statements to detect the different flight phases and emergency situations.

Fig. 4.5 to 4.7 (the emergency landing experiment) show the ability of the IAS to imitate the human pilot's demonstration of controlling an emergency landing. In fact, due to the good fit of the generated model, the IAS performed better than the human pilot by maintaining the required pitch degree (around 1.5 degrees) without the oscillations present in the demonstration of the human pilot as Fig. 4.5 shows. The IAS was able to perform the learned sink rate which enabled the aircraft to hit the ground smoothly without being severely wrecked. Fig. 4.8 illustrates the G-Load factor that the aircraft experienced during multiple emergency/crash landings performed by the IAS where the G-Load factor remained below 1.3 g (1.3 times the aircraft's weight) compared to more than 4 g (4 times the aircraft's weight) resulting from a manual emergency landing performed by the author. The latter results show that the performance of the IAS when maintaining the desired pitch and sink rate insured smooth crash-landings upon impact. Having a dedicated ANN to control the aircraft's pitch during emergency/crash landings by maintaining a positive pitch of around 1.5 degrees as Fig. 4.5 shows, resulted in the ability to perform emergency/crash landings successfully without wrecking the aircraft's structure. It should be mentioned that selecting a suitable landing surface is not within the scope of this work.

Fig. 4.9 and 4.10 (maintaining a cruise altitude experiment) show the ability of the IAS to learn how to use the throttle and the elevators to maintain a given altitude. They illustrate the ability of the IAS to perform better than the human pilot teacher due to the achieved good fit of the generated models. Segregating the altitude maintenance task to two sub-tasks by having a dedicated ANN to control the throttle based on altitude, another dedicated ANN to control the elevators based on the aircraft's pitch, and having both ANNs operate simultaneously resulted in the ability to handle the new task of altitude maintenance as the results show

compared to prototype 1 which was not equipped with the capability of maintaining cruise altitude.

As can be seen in Fig. 4.11 and 4.12, the IAS was capable of using the already learned models to continue flying while gradually losing altitude due to a single engine failure. The aircraft's standard autopilot maintained a better altitude by aggressively increasing the thrust of the remaining operational engine to maintain the manually selected cruise speed. By doing so, additional lift was provided which allowed for the better altitude maintenance when a single engine failed compared to the IAS. This shows the importance of having the capability of maintaining speed which is required during all the different flight phases as well. Therefore, this capability should be added to the IAS to achieve better autonomous piloting capabilities.

The IAS was able to imitate multiple human pilot's skills and behaviour after being presented with very limited examples. This is due to the approach of segmenting the problem of autonomous piloting while handling uncertainties into small blocks of tasks, and assigning multiple ANNs specially designed and trained for each task, which resulted in the generation of models with small error values as tables 4.2, and 4.3 show.

The results proved the hypothesis that ANNs can learn complex tasks representing the ability to handle multiple emergency situations including Rejected Takeoff (RTO), engine failure and fire, and emergency landing, in addition to the newly added ability of maintaining a desired cruise altitude.

4.4 Summary

To summarize, the objective of teaching Artificial Neural Networks how to handle complex emergency tasks was achieved by developing additional ANNs each designed and trained to handle specific tasks representing the ability to handle multiple emergency situations including Rejected Takeoff (RTO), engine failure and fire, and emergency landing. In addition, the task of maintaining a desired altitude is now one of the capabilities of the IAS compared to Prototype 1 which was not able to do so. The design and training approach of the IAS enabled the smooth transition to a more complex and larger aircraft with multiple engines (Boeing B777). The introduction of the Flight Manager program provided the ability to manage the different ANNs that comprise the IAS by continuously monitoring the flight condition, and reacting accordingly by activating the suitable ANNs given the flight condition or phase. The work in this chapter provides additional evidence to support the hypothesis of this work aimed towards proving the possibility to teach a flight control system piloting skills.

5. PROTOTYPE 3 (NAVIGATION & LANDING)

After the first and second objectives of using Artificial Neural Networks to learn basic flying and handle complex tasks such as emergency situations were achieved, the purpose of the third prototype was to achieve the objective of teaching ANNs how to takeoff from airport A, navigate to airport B, and land safely. Performing these tasks represent new abilities that the first and second prototypes of the IAS did not have.

The work in this chapter was published in the 2017 IEEE Symposium Series on Computational Intelligence (SSCI), Hawaii, USA. (Appendix B).

To achieve the third objective mentioned above, the IAS should imitate the behaviour of the human pilot when banking the airplane to follow the flight course. The IAS should also imitate the behaviour of the human pilot when performing final approach and landing in airports. While the previously gained skills from prototypes 1 and 2 along with the components developed to achieve those skills remain without change, new skills should be learned, and their relevant components must be developed to equip the IAS with the capability to handle a challenging problem within this work's scope of autonomous flying, which is the ability to navigate and land safely. The problem should be segregated into two different flight phases (final approach, and landing) based on the problem break-down approach followed in this work. For each flight phase, specific components must be developed to handle the new tasks of final approach and landing by identifying the required control interfaces and surfaces that are used during these flight phases, and developing new ANNs that are dedicated to each segregated task. In addition, a modification should be performed on the Aileron ANN from prototype 1 to learn the new task of following a flight course represented by path lines between GPS coordinates.

The task of autonomous navigation and landing was segregated into four subtasks. The first subtask is following a navigation path based on GPS waypoints, which is the standard navigation method used by airliners nowadays [1]. The control surfaces of the aircraft that are used to handle this subtask were identified based on the literature [1], which are the ailerons that are used to control the aircraft's roll degree. Therefore, ANN 3 from prototype 1 (chapter 3) should be modified to account for path following in addition to controlling the aircraft's roll degree. To achieve this, the modified ANN should take the roll degree and an additional value which indicates how far the aircraft is from the path line representing the flight course as inputs, and generate control commands to the ailerons as output. The second subtask is handling the

final approach flight phase. This subtask is segregated into three additional subtasks. The first is controlling the aircraft's pitch during final approach which is performed using the aircraft's elevators. To achieve this, a new ANN should be designed and trained to take the aircraft's pitch degree as input, and generate the control commands as outputs to the elevators which were identified as the appropriate control surfaces for this tasks according to the literature [1]. The second is controlling the aircraft's glide towards the landing runway. The control interface and surfaces that are used for this subtask are the throttle, the flaps, and the elevators trim [1]. Therefore, a new ANN should be designed and trained to take the altitude and speed as inputs, and generate control commands to the throttle, flaps, and the elevator's trim as outputs. The third is controlling the aircraft's gear before landing. To achieve this, a new ANN should be designed and trained to take altitude as input, and based on that, generate control command to the aircraft's gear as output. The final subtask is performing landing procedures after touchdown. To achieve this, a new ANN should be designed and trained to take speed as input, and based on that, generate control commands to the aircraft's brakes, reverse thrust, and speed brakes as outputs. These control interface and surfaces are used to bring the aircraft to a full stop after touchdown [1].

Fourteen feedforward Artificial Neural Networks now comprise the core of the IAS. The additional four ANNs were designed based on the same approach mentioned in section (*3.2 Training*). Each ANN is designed and trained to handle specific controls and tasks. The new ANNs are: Final Approach ANN which controls throttle, flaps, and elevators trim to maintain altitude descending during the final approach phase, Final Approach Pitch ANN which controls the elevators to maintain a positive pitch during the final approach phase, Gear ANN which engages landing gear when a certain altitude is reached, and Landing ANN which controls reverse thrust, brakes, and speed brakes after touchdown to slow down the aircraft on the landing runway, and bring it to a full stop. The inputs and outputs which represent the gathered data and relevant actions, and the topologies of the ANNs are illustrated in Fig. 5.1.

The Flight Manager program from prototype 2 was extended to give it the ability to generate a navigation course for the Aileron ANN to follow, and handle the flight phases necessary for landing, which are the final approach phase, and the landing phase.

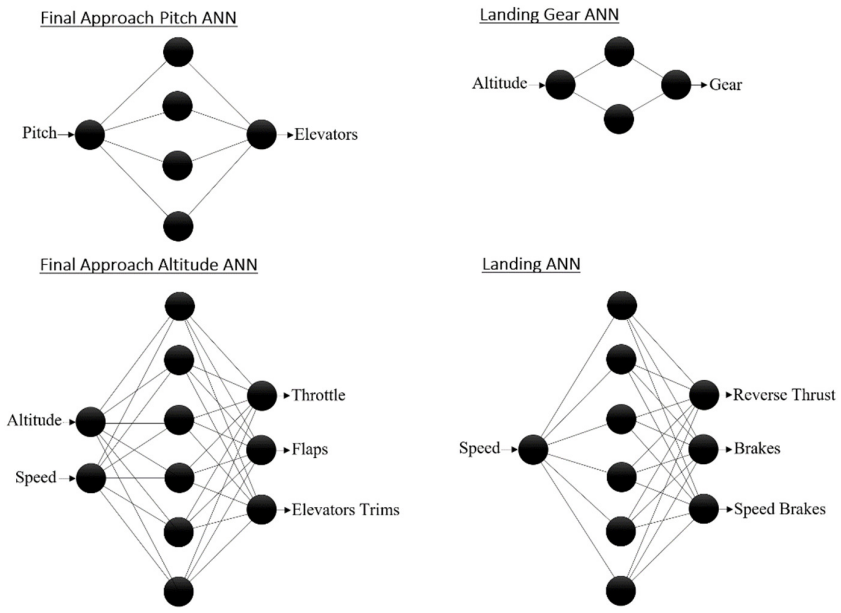

Fig. 5.1. Inputs, outputs, and the topologies of the new ANNs of prototype 3.

The Flight Manager generates a flight course to the destination airport of choice based on stored GPS waypoints as Fig. 5.2 illustrates by applying (5.1) [147] to calculate the bearing (heading) between the GPS coordinates (latitude and longitude) of the waypoints.

$$\Phi = atan2(sin(\Delta\lambda)cos(\Phi_2), cos(\Phi_1)sin(\Phi_2)cos(\Delta\lambda)) \quad (5.1)$$

where Φ_1 is the latitude of the start point (waypoint 1), Φ_2 is the latitude of the end point (waypoint 2), and $\Delta\lambda$ is the difference between the longitudes of the end point (waypoint 2) and the first point (waypoint 1).

The program constantly measures the deviation between the aircraft's position and the current path line of the flight course represented by the angle between the line that starts at the location point of the aircraft and ends at the location point of the next waypoint, and the line that starts at the location point of the previous waypoint and ends at the location point of the next waypoint as Fig. 5.3 illustrates.

The Flight Manager calculates the difference between the bearing of the path line to be intercepted, and the aircraft's current bearing, then, it adds the angle to the bearing difference as a momentum value (5.2). As the aircraft's current bearing becomes closer to the desired bearing, and as the angle becomes smaller, the difference becomes smaller as well, which leads to a gradual interception of the path line, and avoids undesired undershooting or overshooting

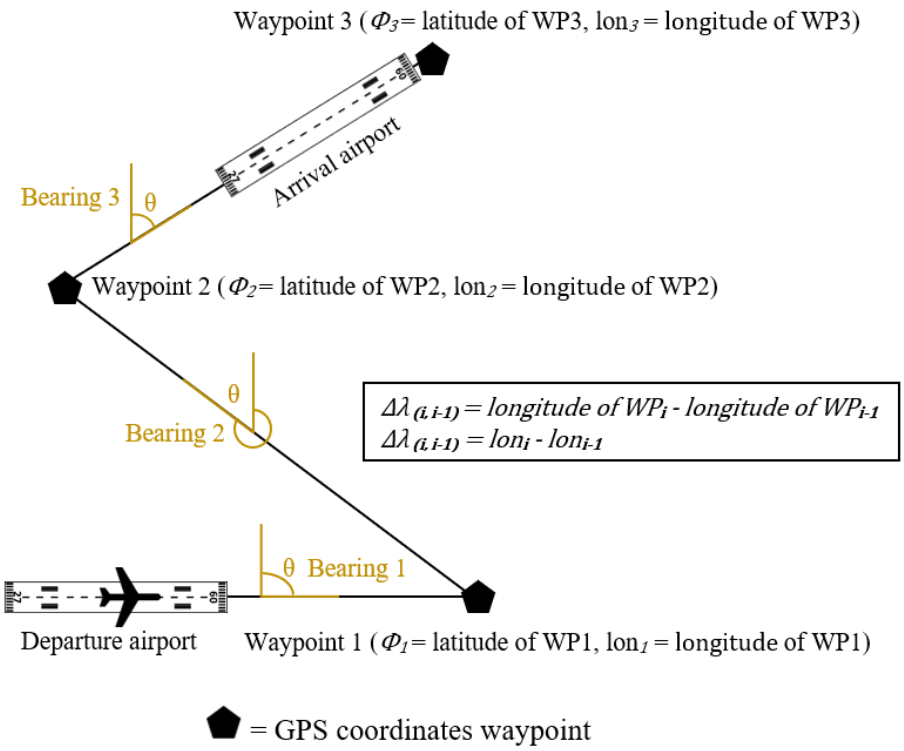

Fig. 5.2. A flight course from a departure airport to a landing airport consisting of three path lines and their bearings/headings, which connect the three pre-stored GPS coordinates waypoint.

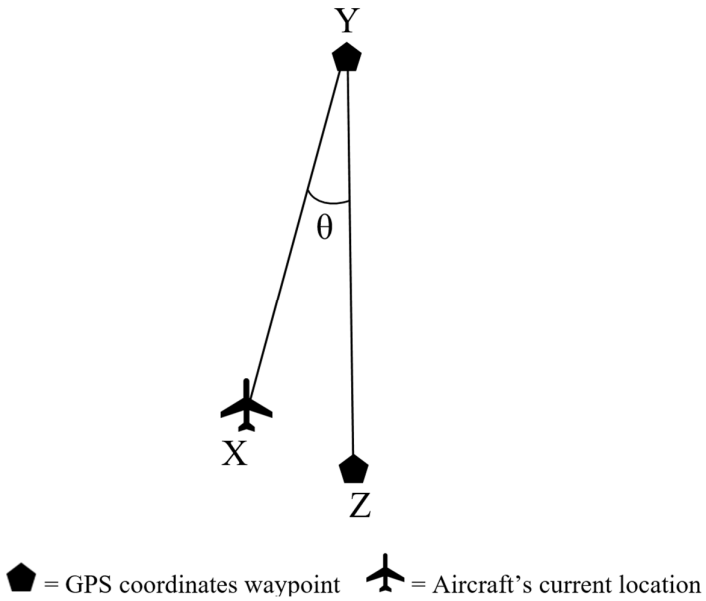

Fig. 5.3. The angle between the line from the aircraft's location X and the next waypoint Y, and the line from the previous waypoint Z and the next waypoint Y.

as Fig. 5.4 illustrates. The difference is fed to the modified Aileron ANN from prototype 1, which now takes the difference in addition to roll as inputs, and outputs the appropriate ailerons control commands as Fig. 5.5 shows.

$$ailerons\ ANN\ input = (current\ bearing - desired\ bearing) + angle \quad (5.2)$$

The Top of Descent (TOD) is the point at which descending towards the destination airport is initiated. In this work, the Flight Manager calculates the TOD by multiplying the altitude by 0.003 [3]. If the result is less than the distance (in kilometres) to the landing runway, then the TOD is reached, and the descending process starts.

The Glideslope is an altitude slope of a given degree, which leads to a touchdown on the landing runway. The Flight Manager generates a virtual altitude slope by dividing the distance to the runway by 10. The latter method is used based on preliminary empirical testing, however, this method is updated in chapter 7 by using the standard glideslope interception technique that utilizes the elevator's trim. Fig. 5.6 illustrates how the Flight Manager generates the glideslope.

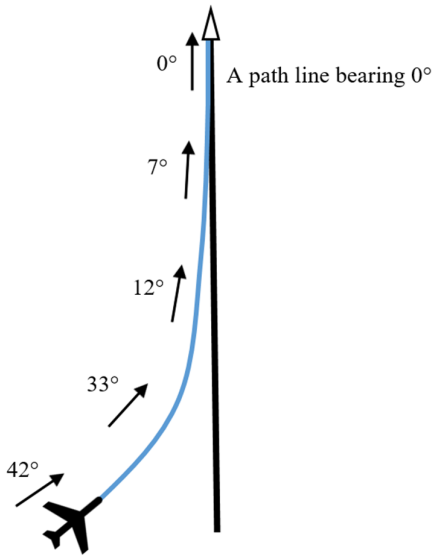

Fig. 5.4. An example illustrating how the Flight Manager updates the bearing to be followed based on the difference between the bearing of the path line to be intercepted, and the aircraft's bearing. The angle between the aircraft and the path line is added to the difference to ensure a gradual interception.

[3] How to compute the TOD (Top of Descent) - Thumb rule. https://community.infinite-flight.com/ [accessed 2018]

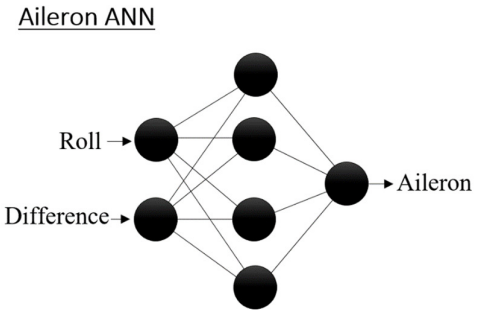

Fig. 5.5. The modified Aileron ANN which now takes the difference in bearings (angle added to it) in addition to roll to predict the appropriate aileron control command during navigation.

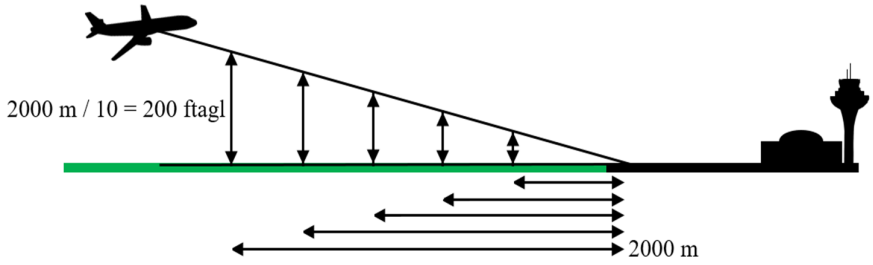

Fig. 5.6. The Glideslope generated by continuously calculating the altitude during the final approach descent which leads to a touchdown on the landing runway. The desired altitude is the distance to the runway divided by 10.

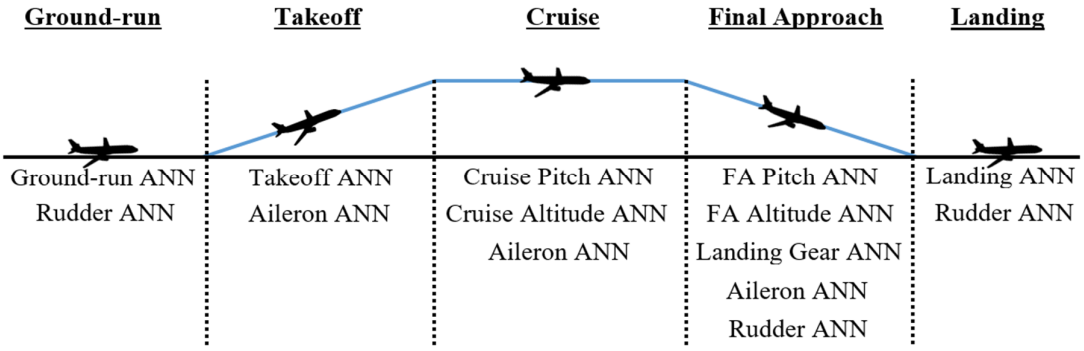

Fig. 5.7. The ANNs used during the different phases of the flight.

Fig. 5.7 illustrates the ANNs used during the different flight phases, and Fig. 5.8 illustrates the process which the Flight Manager follows to handle the different flight phases from takeoff to landing.

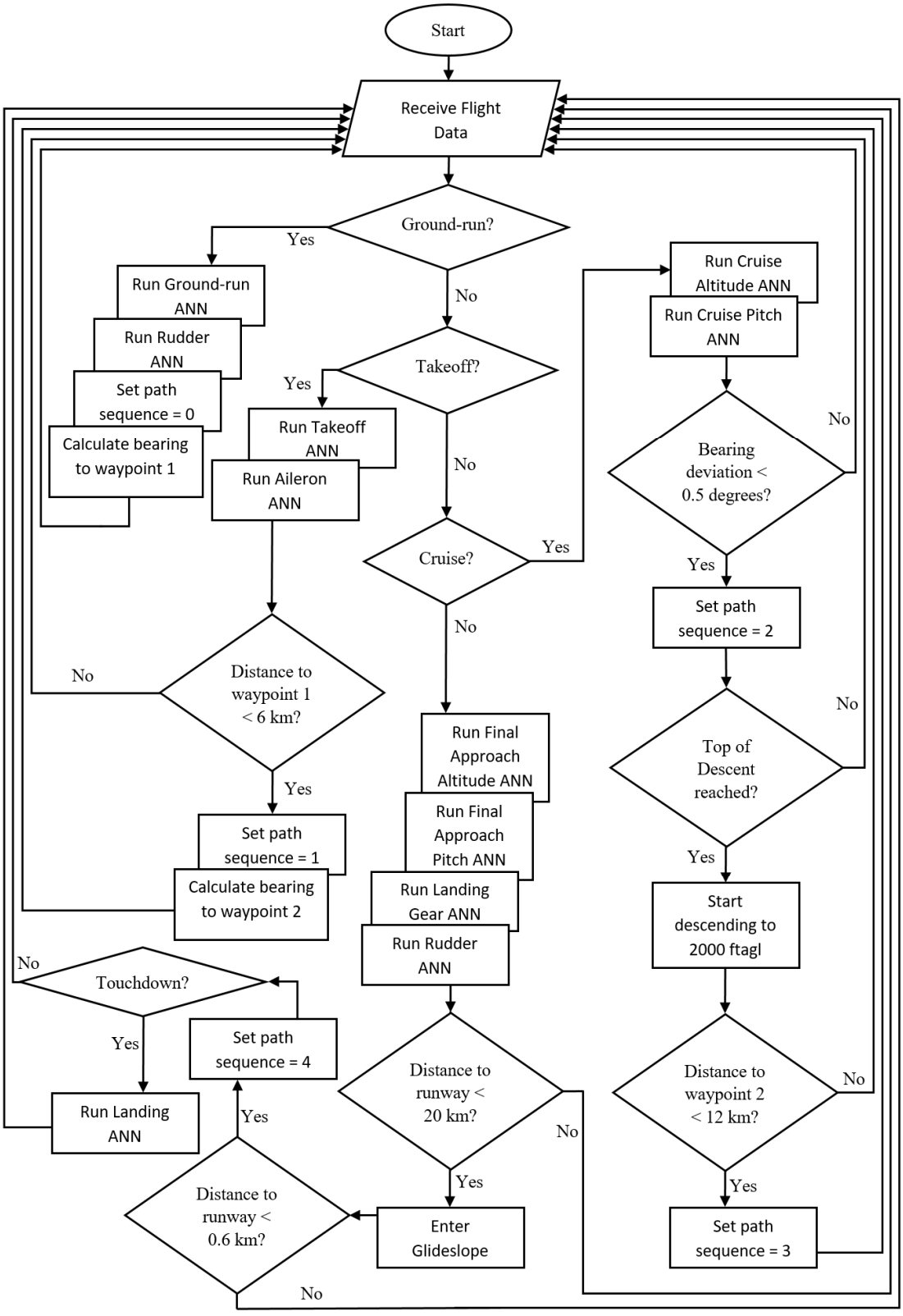

Fig. 5.8. A Flowchart illustrating the process which the Flight Manager program follows to decided which ANNs are to be used, and how to handle flight phases and navigation points transitions.

5.1 Experiments on Prototype 3

This section discusses the experiments conducted on the modified Aileron ANN which can now bank, and intercept a path line, in addition to controlling the roll degree. This section also discusses the experiments conducted on the new ANNs that are used during the final approach, and landing phases.

The experiments were conducted under calm weather conditions with nil wind speed.

To assess the effectiveness of the proposed approach, the Intelligent Autopilot System was tested in three experiments:

1. Banking turn and path line interception

2. Final approach

3. Landing.

Each experiment is composed of 50 attempts by the IAS to perform autonomously under the given conditions. The human pilot who provided the demonstrations is the author. The simulated aircraft used for the experiments is a Boeing 777 as it is intended to experiment using a complex and large model with more than one engine rather than a light single-engine model.

5.1.1 Banking Turn and Path Line Interception

The purpose of this experiment is to assess the behaviour of the IAS compared with the behaviour of the human pilot when performing a banked turn, and to assess the path line interception technique.

5.1.1.1 Data Collection

In this experiment, the human pilot used the IAS Interface to change the aircraft's heading by performing a banked turn through maintaining a roll of 25 to 35 degrees which is the standard roll degree for airliners when banking to change the flight course [1]. While the pilot performed the demonstration, the Interface collected roll and difference values as inputs, and aileron control value as output. The Interface stored the collected data in the database as the training dataset for the Aileron ANN.

5.1.1.2 Training

For this experiment, the Aileron ANN was trained until a low Mean Squared Error (MSE) value was achieved (below 0. 1). Since single-hidden-layer ANNs are used for training, and

since the training datasets are small given that a single demonstration was provided, training requires a short time to be completed (under 10 minutes).

However, when the aircraft is close to the path line to be intercepted, a large banking turn of 25 to 35 degrees of roll can cause the aircraft to constantly overshoot the path line instead of intercepting it smoothly. So, instead of training an additional Aileron ANN that performs banking turns through smaller degrees of roll, the same generated ANN model can be stimulated differently to alter its behaviour by feeding its input neuron with values that are much smaller than the values present in the training dataset. The latter exploits the generalization effect which causes the model to behave differently based on the unseen smaller inputs. To achieve this in this experiment, before feeding the difference input neuron of the Aileron ANN with the difference value, the difference is reduced to just 30% of its actual value, which was found through extensive preliminary experiments. This approach was tested between two waypoints represented by a straight path line.

5.1.1.3 Autonomous Control

After training the ANN on the relevant training dataset, the aircraft was reset to the runway in the flight simulator to test autonomous banking turn and path line interception. After takeoff, when the Flight Manager updates the path sequence of the flight course, calculates the new heading, the angle, and the difference, the Aileron ANN performs a banked turn to minimize the difference, and eventually, intercept the path line gradually. Through the Interface, the ANN receives: 1. relevant flight data from the flight simulator as inputs, and 2. coefficients of the relevant models from the database to predict and output control commands that are sent to the flight simulator. This process allows the IAS to autonomously perform the learned task: autonomous banking turn and path line interception. This was repeated 50 times to assess performance consistency.

5.1.2 Final Approach

The purpose of this experiment is to assess the behaviour of the IAS compared with the behaviour of the human pilot during the final approach phase.

5.1.2.1 Data Collection

In this experiment, the human pilot used the IAS Interface to perform the following in the flight simulator: maintain a positive pitch of about 3 to 4 degrees during the final approach phase to decrease airspeed without causing a stall, and to ensure a flare immediately after touchdown, engage full flaps when the airspeed is less than 260 knots, and engage the landing

gear when the altitude decreases to 1500 ftagl. The desired descent altitude is continuously updated by the Flight Manager. While the pilot performed the demonstration, the Interface collected airspeed and altitude as inputs, and flaps as output. The Interface stored the collected data in the database as the training dataset for the Final Approach Altitude ANN. The Interface also collected altitude as input, and landing gear control data as output. The Interface stored the collected data in the database as the training dataset for the Landing Gear ANN.

5.1.2.2 Training

For this experiment, the Final Approach Altitude ANN, and the Landing Gear ANN were trained until low Mean Squared Error (MSE) values were achieved (below 0.01). Since single-hidden-layer ANNs are used for training, and since the training datasets are small given that a single demonstration was provided, training requires a short time to be completed (under 10 minutes).

5.1.2.3 Autonomous Control

After training the ANNs on the relevant training datasets, the aircraft was reset to the runway in the flight simulator to test the autonomous final approach procedures. After entering the final approach flight phase, and when the desired airspeed is reached, the Final Approach Altitude ANN engages flaps, and when the desired altitude is reached, the Landing Gear ANN engages the landing gear. Through the Interface, the ANNs receives: 1. relevant flight data from the flight simulator as inputs, and 2. coefficients of the relevant models from the database to predict and output control commands that are sent to the flight simulator. This process allows the IAS to autonomously perform the learned final approach procedures. This was repeated 50 times to assess performance consistency.

5.1.3 Landing

The purpose of this experiment is to assess the behaviour of the IAS compared with the behaviour of the human pilot when performing landing procedures.

5.1.3.1 Data Collection

In this experiment, the human pilot used the IAS Interface to perform the landing procedures immediately after touchdown, by engaging reverse thrust, brakes, and speed brakes. While the pilot performed the demonstration, the Interface collected airspeed as input, and reverse thrust, brakes, and speed brakes control data as outputs. The Interface stored the collected data in the database as the training dataset for the Landing ANN.

5.1.3.2 Training

For this experiment, the Landing ANN was trained until low Mean Squared Error (MSE) values were achieved (below 0.01). Since single-hidden-layer ANNs are used for training, and since the training datasets are small given that a single demonstration was provided, training requires a short time to be completed (under 10 minutes).

5.1.3.3 Autonomous Control

After training the ANN on the relevant training dataset, the aircraft was reset to the runway in the flight simulator to test the ability of performing the landing procedures autonomously, and the IAS was engaged. After the IAS took the aircraft airborne, navigated to the destination airport, and touched down, the system's ability to perform the landing procedures of engaging reverse thrust, brakes, and speed brakes was observed. Through the Interface, the ANN receives: 1. relevant flight data from the flight simulator as inputs, and 2. coefficients of the relevant models from the database to predict and output control commands that are sent to the flight simulator. This process allows the IAS to autonomously perform learned landing procedures. This was repeated 50 times to assess performance consistency.

5.2 Results of Experiments on Prototype 3

The following section describes the results of the conducted tests.

5.2.1 Experiment 1 (Banking turn and path line interception)

One model was generated for the Aileron ANN with an MSE value of 0.0954. Fig. 5.9 illustrate a comparison between the human pilot and the IAS when performing a banked turn to change the aircraft's bearing by 145 degrees over a period of 40 seconds. The Mean Absolute Deviation (MAD) results of the roll degrees over time (5.02 for the IAS (average) and 4.34 for the human pilot) show a close behaviour between the system and its teacher. Fig. 5.10 illustrates the smaller roll degrees when intercepting a path line after altering how the difference input neuron is stimulated by reducing the input's value to just 30% of its actual value. Fig. 5.11 illustrates how altering the input value by reducing it to 10% and 50% causes the Ailerons ANN to generate different control commands to the ailerons which resulted in different roll degree values of around 3 degrees (10% of the full roll degree of 30), and around 15 degrees (50% of the full roll degree of 30) without having to retrain the ANN to produce those different roll degrees. Fig. 5.12 shows that the reduction of 30% was selected after comparing different reduction rates where applying 30% of the actual value resulted in the desired interception of the path line without overshooting. Since the experiments took place

while intercepting the same path line of the flight course under the same weather conditions (nil wind speed), and since many of the experiments took place in the same segment of the path line, many lines overlap as Fig. 5.10 illustrates.

Fig. 5.13 illustrates how applying the reduced degrees of roll when constantly banking (correcting bearing) to intercept a path line, ensure a steady and gradual interception. Fig. 5.14 illustrates the flight course that the IAS generated and followed autonomously.

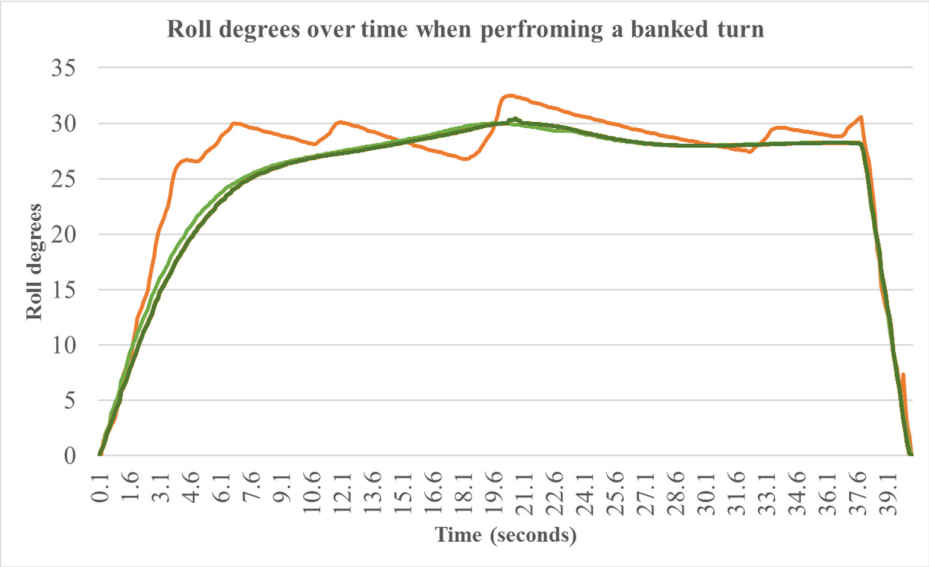

Fig. 5.9. (Banking turn and path line interception experiment). A comparison between the human pilot (orange line) and the 50 attempts by the IAS (overlapping green lines) to perform a banked turn to change the aircraft's bearing by 145 degrees over a period of 40 seconds. The good fit of the generated model allowed the IAS to maintain a steadier change of roll degrees compared with the human pilot.

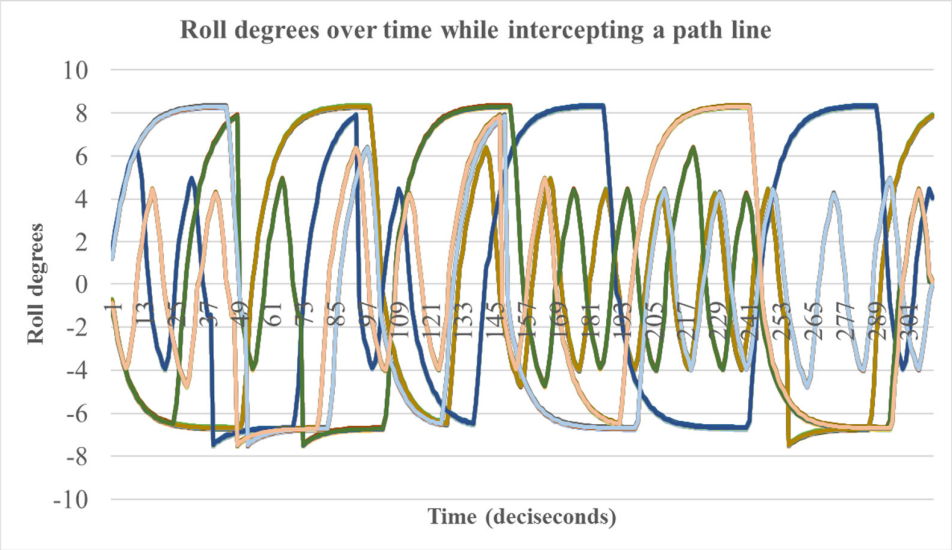

Fig. 5.10. (Banking turn and path line interception experiment). Smaller degrees of roll when banking continuously to intercept a path line. For the 50 attempts (each colour line corresponds to one flight experiment, however, most lines are overlapped), the roll is below 9 degrees (suitable for small turns while intercepting) compared with a maximum of 31 (suitable for major bearing change) as Fig. 5.9 illustrates.

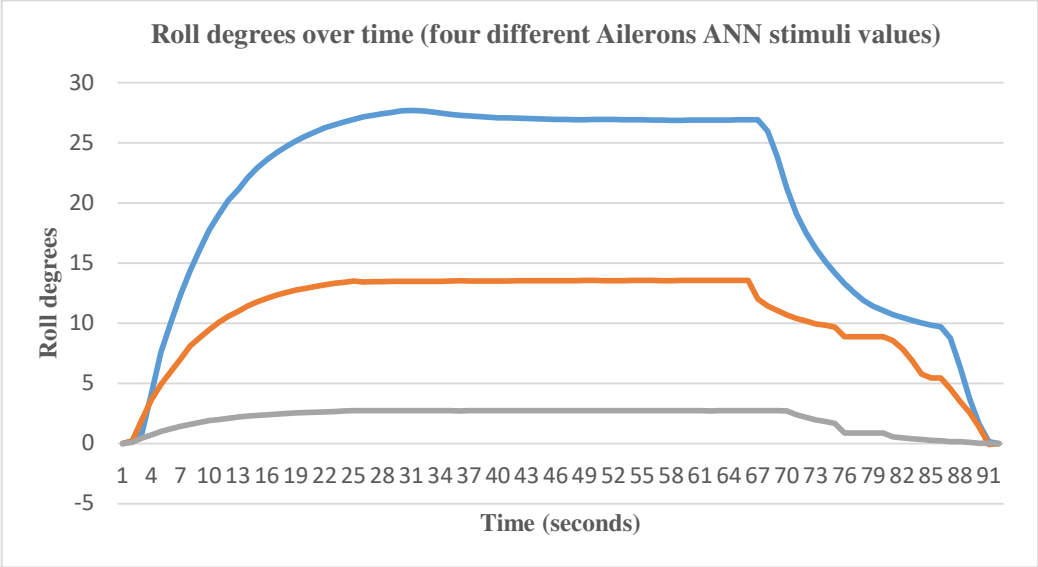

Fig. 5.11. (Banking turn and path line interception experiment). Changing the stimuli or the input of the Ailerons ANN causes the ANN to generate different control command outputs to the aircraft's ailerons, which results in different roll degrees. The blue line represents keeping the input as it is which produces a roll degree of about 30 as per the training of the Ailerons ANN. The orange line represents a reduction of 50% which produces a roll degree of about 15. The grey line represents a reduction of 10% which produces a roll degree of about 3.

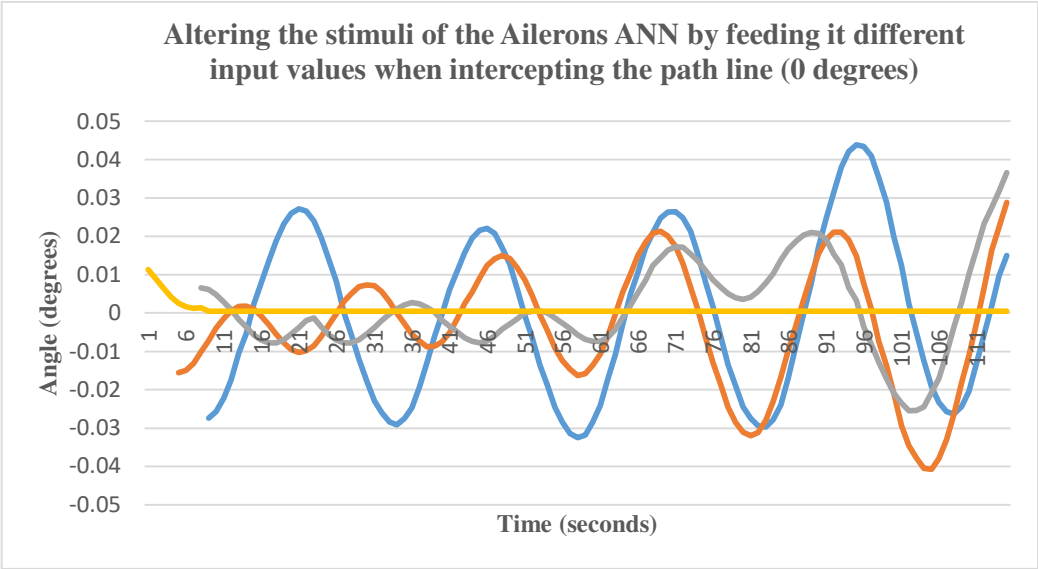

Fig. 5.12. (Banking turn and path line interception experiment). Changing the stimuli or the input of the Ailerons ANN causes the aircraft to behave differently when intercepting the path line (0 degrees). The blue line illustrates keeping the input without altering which causes the aircraft to overshoot since the Ailerons ANN is generating roll degrees of around 30. The orange line illustrates reducing the input of the Ailerons ANN to 50% of its value which causes the aircraft to overshoot as well. The grey line illustrates reducing the input of the Ailerons ANN to 40% of its value which still causes the aircraft to overshoot. The yellow line illustrates reducing the input of the Ailerons ANN to 30% of its value which allows the aircraft to intercept the path line smoothly.

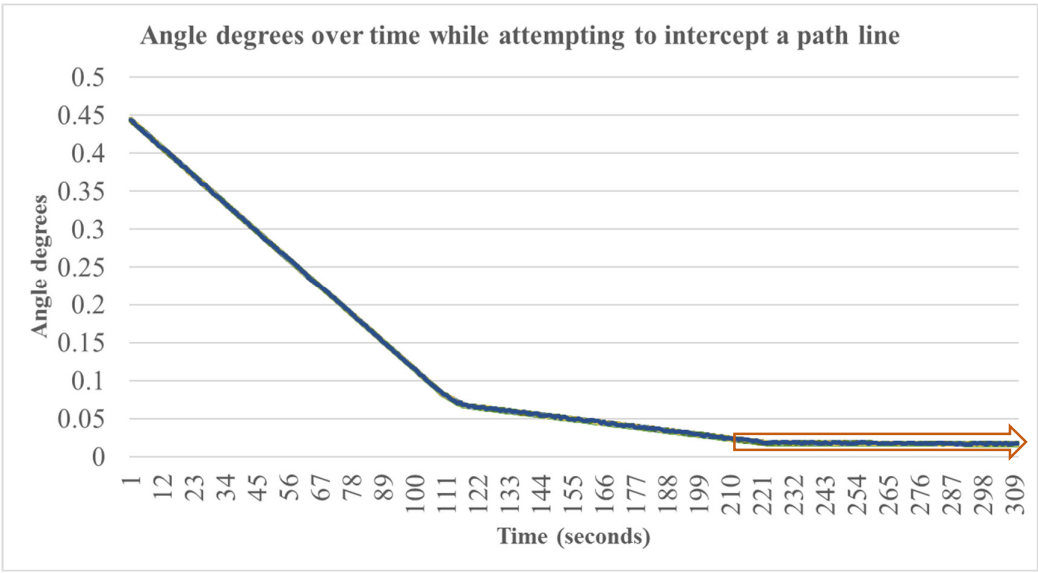

Fig. 5.13. (Banking turn and path line interception experiment). 50 attempts (all lines are overlapped) to gradually intercept and follow a path line (orange arrow). The interception attempt is represented by the gradual decrease of the angle between the aircraft and the path line.

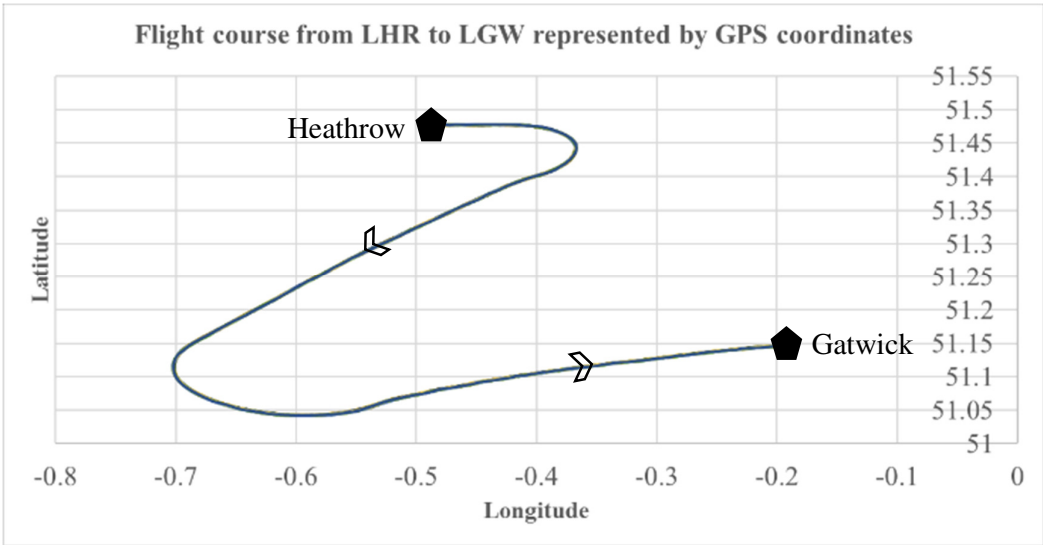

Fig. 5.14. (Banking turn and path line interception experiment). The 50 flight courses (overlapped lines) flown autonomously by the IAS, starting with takeoff from London Heathrow airport, and landing at Gatwick airport. Since the distance between the two airports is short for an airliner, the generated flight course accounts for the distance required to perform the final approach phase, and therefore, follows an initial path away from Gatwick.

5.2.2 Experiment 2 (Final Approach)

Two models were generated for this experiment, the Final Approach Altitude ANN model with an MSE value of 0.0034, and the Landing Gear ANN model with an MSE value of 0.0046. Fig. 5.14 illustrate a comparison between the human pilot and the IAS when extending the flaps after the appropriate airspeed is reached. Fig. 5.15 illustrates a comparison between the human pilot and the IAS when engaging the landing gear after the appropriate altitude is achieved. The lines representing the human behaviour, and the 50 attempts by the IAS in Fig. 5.14 and 5.15 overlap. Fig. 5.16 illustrates the glideslope followed during the final approach phase in 50 experiments resulting in lines that overlap.

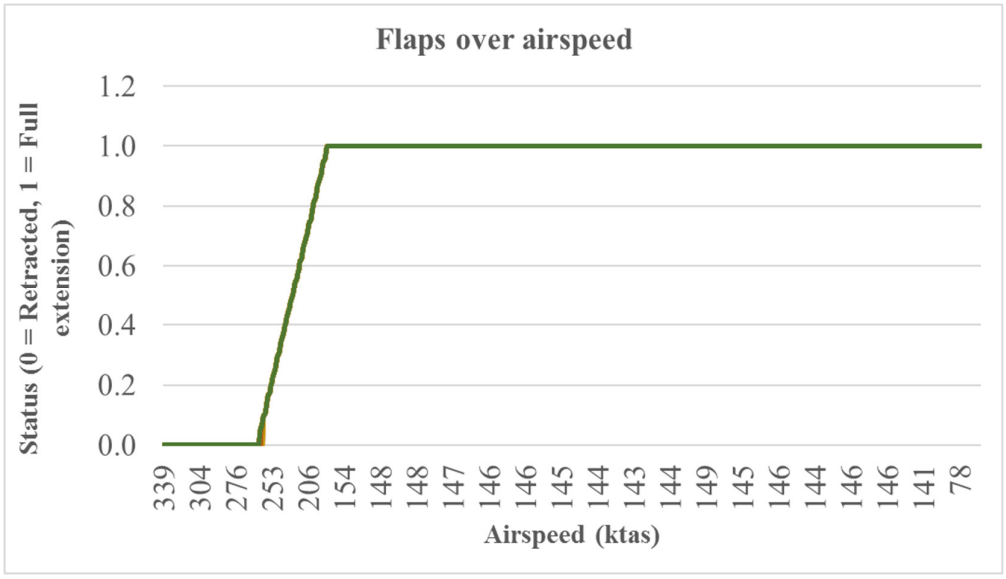

Fig. 5.15. (Final approach experiment). A comparison between the human pilot and 50 attempts by the IAS (overlapped lines) when extending flaps immediately after an airspeed of 260 (ktas) is reached. The results show the similarity between the behaviour of the human pilot and the IAS during the 50 attempts.

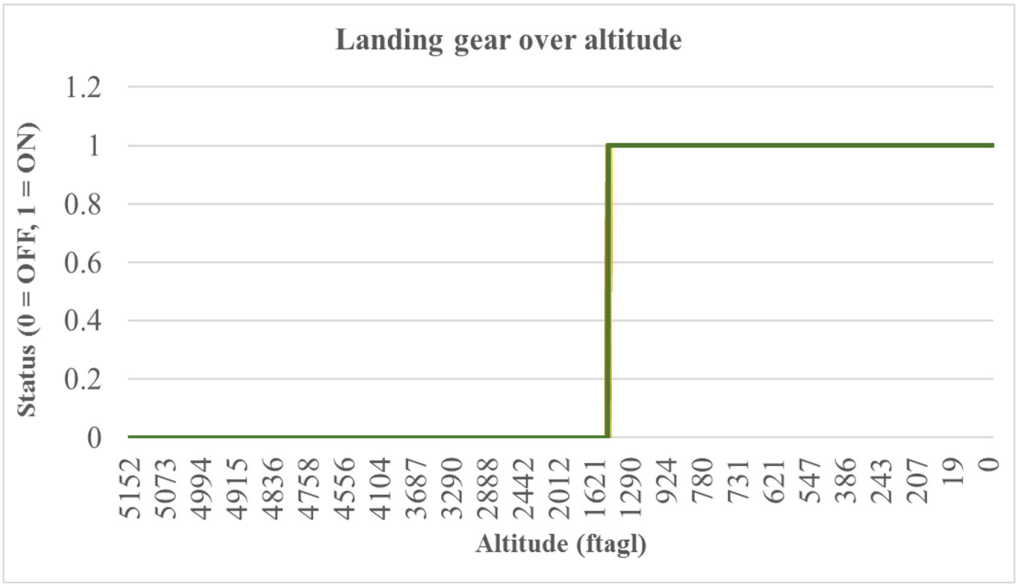

Fig. 5.16. (Final approach experiment). A comparison between the human pilot and 50 attempts by the IAS (overlapped lines) when engaging the landing gear immediately after an altitude of 1500 (ftagl) is reached. The results show the similarity between the behaviour of the human pilot and the IAS during the 50 attempts.

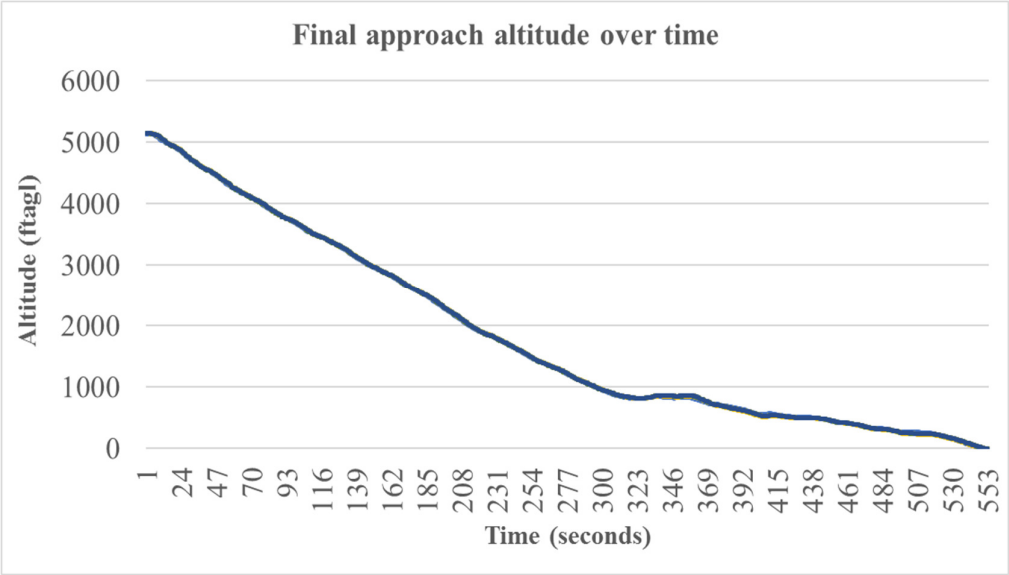

Fig. 5.17. (Final approach experiment). 50 attempts (overlapped lines) by the IAS to follow the final approach glideslope. The glideslope is adjusted by the IAS after descending to an altitude below 1000 (ftagl) to compensate for the additional drag generated by the landing gear.

5.2.3 Experiment 3 (Landing)

One model was generated for the Landing ANN with an MSE value of 0.003. Fig. 5.17 illustrate a comparison between the human pilot and the IAS when engaging the reverse thrust, brakes, and speed brakes immediately after touchdown. The lines representing the human behaviour, and the 50 attempts by the IAS in Fig. 5.18 overlap.

5.3 Analysis

As can be seen in Fig. 5.9 (banking turn and path line interception experiment), the IAS was not only able to imitate the behaviour of its human teacher when performing a banked turn by maintaining a certain degree of roll, it was also able to perform better by being able to maintain a steadier change of roll degrees, which is due to the good fit of the generated model. The new method of changing the stimuli of the ANN proved its ability to alter the ANN's behaviour without having to retrain it or generate a different learning model.

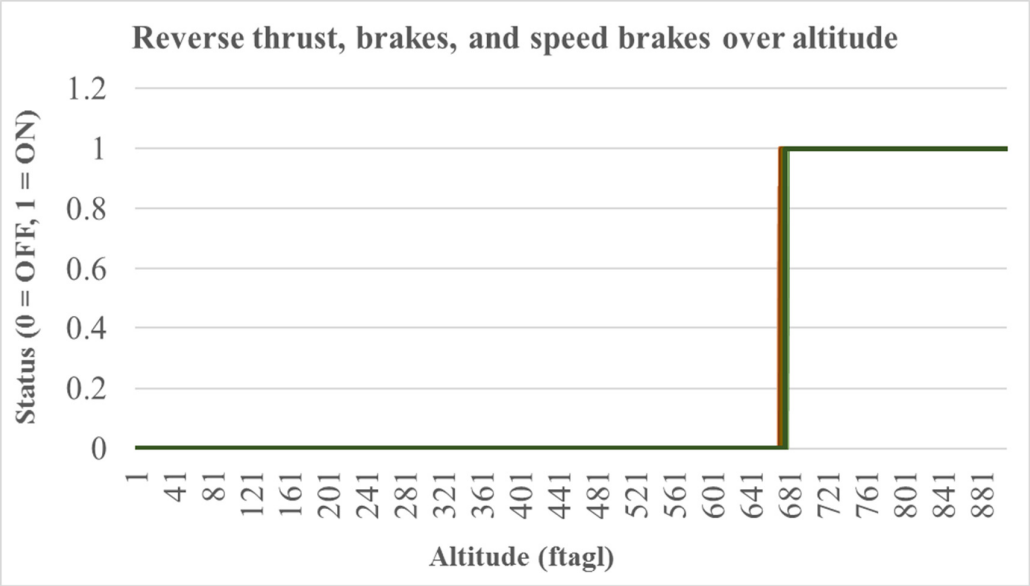

Fig. 5.18. (Landing experiment). A comparison between the human pilot and 50 attempts by the IAS (overlapped lines) when engaging reverse thrust, brakes, and speed brakes immediately after touchdown. The results show the similarity between the behaviour of the human pilot and the IAS during the 50 attempts.

Fig. 5.10 (banking turn and path line interception experiment) shows how changing the stimuli represented by the difference value which passes through the input neuron of the Ailerons ANN through reducing it by a given percentage caused the ANN to behave differently. The latter can be seen as the much smaller degrees of roll (around 8 degrees) maintained by the Ailerons ANN although its generated learning model was trained to maintain larger degrees of roll (around 30 degrees). This can be seen in Fig. 5.11 as well, which illustrates the effectiveness of the proposed stimuli-altering method that eliminates the need to retrain the Aileron ANN to produce different roll degrees. Choosing the appropriate reduction rate of the input of the Ailerons ANN (30% of the original input value) insured the smooth interception of the path line without overshooting as Fig. 5.12 illustrates. The smaller degrees of roll maintained when banking or correcting the aircraft's bearing to intercept a path line, allowed the aircraft to gradually intercept and follow the path line while avoiding overshooting as Fig. 5.13 (banking turn and path line interception experiment) shows, and therefore, the IAS was able to follow the generated flight course as Fig. 5.14 (banking turn and path line interception experiment) shows, throughout all the experiments.

The IAS was capable of imitating the human pilot's actions and behaviour when performing the procedures of the final approach phase, by extending the flaps only when a certain airspeed

is reached, and engaging the landing gear only when a certain altitude is reached as Fig. 5.15 and 5.16 (final approach experiment) show. The method covered in prototype 2, which is followed by the ANN that is responsible for maintaining a given altitude, proved to be adequate for handling a rapidly changing desired altitude which is continuously updated by the Flight Manager during the final approach phase. The latter generated a glideslope that led to a touchdown on the landing runway, and was maintained by the IAS. However, as soon as the extra drag caused by the extracted landing gear generated a larger sink rate which could cause a premature touchdown (touching down before reaching the landing runway), the IAS was able to autonomously alter the glideslope by following a less steep degree towards the runway as Fig. 5.17 (final approach experiment) shows.

As can be seen in Fig. 5.18 (landing experiment), the IAS was capable of identically imitating the human pilot's actions and behaviour when performing the procedures of the landing phase, by engaging the reverse thrust, brakes, and speed brakes immediately after touchdown to bring the aircraft to a rapid full stop.

However, at this point, the IAS is not capable of landing under windy conditions. The presence of wind (especially crosswind) during the final approach phase causes the aircraft to drift from the centreline of the landing runway as Fig. 5.19 shows. With the lack of sufficient speed, and thrust from the engines during the final approach phase since the aim here is to descend, the Aileron ANN is not capable of following a straight path line efficiently, which is a problem that does not affect the aircraft during other phases of the flight given the availability of sufficient speed and thrust. Fig. 5.19 shows that even light wind speeds (around 10 knots) causes the aircraft to deviate from the path line although the deviation happens at a point closer to the runway compared to stronger winds. Fig. 5.19 also shows that not just crosswind (at 90 or 270 degrees) causes the aircraft to deviate, but also, 0 (or 360) degrees and 180 degrees winds causes the aircraft to deviate. Fig. 5.20 illustrates how such wind conditions deviate the aircraft from its flight course by comparing the GPS readings of the track that the aircraft took in Fig. 5.20 and Fig. 5.14 just before landing. This means that executing a safe landing at the destination airport in windy conditions is not currently possible, and therefore, a solution must be applied to aid the Ailerons ANN to keep the aircraft in a straight line during final approach even if the weather conditions are severe including the presence of strong crosswind. In addition, Fig. 5.21 illustrates that the IAS was not able to handle fluctuating wind conditions that change in speed and direction (gust and shear) which was expected since the IAS could not handle steady winds that do not change in speed or direction as Fig. 5.19 shows. However,

Fig. 5.21 shows that the presence of turbulence alone does not have a similar path deviation effect such as crosswind for example.

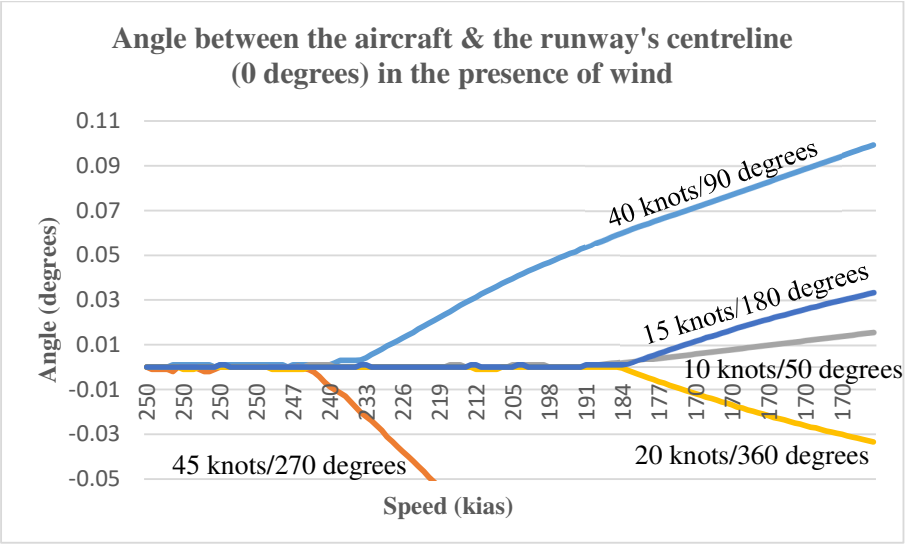

Fig. 5.19. (Landing experiment). The angle between the aircraft and the centreline of the landing runway (0 degrees) in the presence of different speeds and directions of wind. As soon as the speed of the aircraft reaches the speed of the approach flight phase (below 250 knots), the aircraft loses the required momentum to continue intercepting the runway's path line in the presence of windy conditions which causes the aircraft to drift away from the runway's centreline.

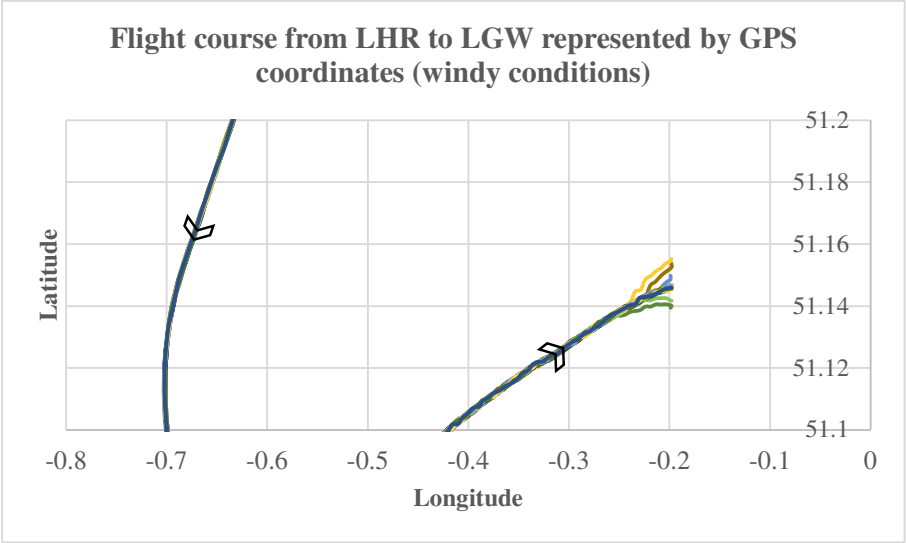

Fig. 5.20. (Banking turn and path line interception experiment). 6 flight courses (overlapped lines before the end of the flight course) flown autonomously by the IAS, starting with takeoff from London Heathrow airport, and landing at Gatwick airport. As can be seen, the wind conditions (shown in Fig. 5.19) deviates the aircraft from its GPS course just before landing due to the low speed.

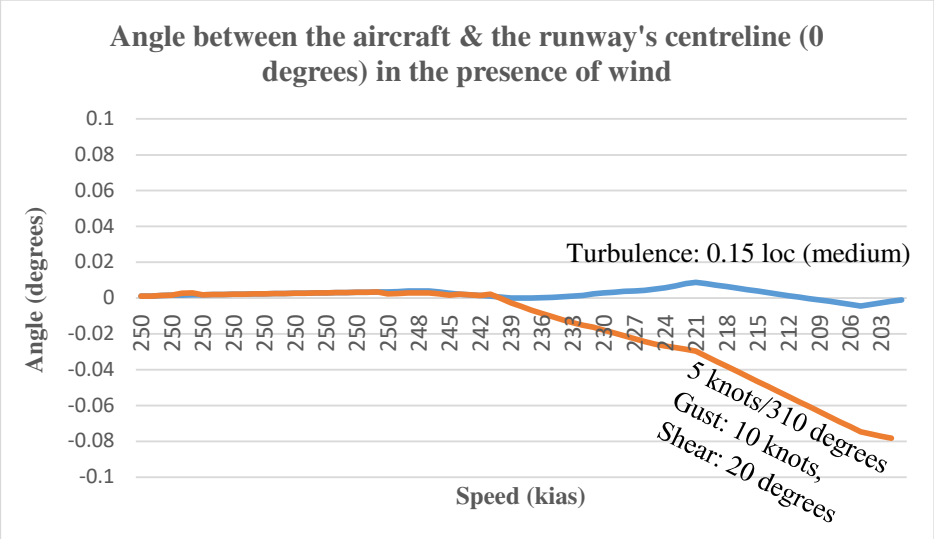

Fig. 5.21. (Landing experiment). The angle between the aircraft and the centreline of the landing runway (0 degrees) in the presence of turbulence alone (blue line), and wind gust and shear (orange line). As soon as the speed of the aircraft reaches the speed of the approach flight phase (below 250 knots), the aircraft loses the required momentum to continue intercepting the runway's path line in the presence of windy conditions which causes the aircraft to drift away from the runway's centreline.

5.4 Summary

To summarize, the objective of teaching Artificial Neural Networks how to handle complex piloting tasks including the ability to takeoff from airport A, navigate to airport B, and land safely, was achieved. This was achieved by developing a GPS based navigation algorithm, and teaching the Ailerons ANN how to bank to change heading, and to maintain a flight path. Additional ANNs were designed and trained to handle the flight phases associated with landing, and the Flight Manager program was extended to handle these new phases. This provides additional evidence to support the hypothesis of this work aimed towards proving the possibility to teach a flight control system piloting skills.

6. PROTOTYPE 4 (SEVERE WEATHER LANDING & GO-AROUND)

After achieving the first, second, and third objectives of using Artificial Neural Networks to learn basic flying, handle complex tasks such as emergency situations, and learn how to fly from one airport to another by autonomously navigating and landing safely, the purpose of the fourth prototype was to achieve the objective of handling severe weather landing, and go-around (aborting landing, then reattempting). The severe weather conditions during final approach and landing include the presence of strong crosswind, wind shear, gust, and turbulence. In addition, higher-level safety would indicate the need to equip the Intelligent Autopilot System with the ability to safely abort landing by performing a go-around manoeuvre if carrying on with landing would jeopardise the safety of the flight. Performing these tasks represent new abilities that the first, second, and third prototypes of the IAS did not have.

The work in this chapter was published in the 2017 International Workshop on Research, Education and Development on Unmanned Aerial Systems (RED-UAS), Linköping, Sweden. (Appendix B).

To achieve the fourth objective mentioned above, the IAS should be able to handle severe weather conditions during final approach and landing, especially strong crosswind which pushes the aircraft away from the centre line of the landing runway given the relatively low speed of the aircraft during these flight phases. In addition, the IAS should also be able to abort landing in case of undesired conditions, and attempt a go-around manoeuvre. While the previously gained skills from prototypes 1, 2, and 3 along with the components developed to achieve those skills remain without change, new skills should be added, and their relevant components must be developed to equip the IAS with the capability to handle the quite challenging problem of landing safely in the presence of severe weather conditions.

In the previous chapter (Prototype 3), the ability of landing in a given airport was added to the IAS, however, as can be seen in the analysis of the previous chapter (especially Figs 5.19, 5.20, and 5.21), the IAS was not able to maintain the path line representing the centreline of the landing runway in the presence of windy conditions during final approach. This is due to the inability of the Ailerons ANN to account for such external forces pushing the aircraft away from the centreline of the landing runway. The Ailerons ANN were designed and trained to follow a given bearing degree between two GPS points as the previous chapter explains, however, if the desired bearing degree is achieved regardless of how far the aircraft is from the

landing runway, the Ailerons ANN would continue to maintain that bearing degree even if the aircraft is not aligned with the landing runway. Therefore, the Ailerons ANN should be aided by a new method which gives them the ability to account for the deviation from the centreline of the landing runway by countering the effect of the external forces represented by wind. The new method should be able to dynamically update and follow different bearing degrees during final approach to counter the drift from the centreline of the landing runway. Since the distance between the aircraft and the path line can be calculated as an angle as the previous chapter explains, the new method should utilize this angle by calculating its rate of change which represents how fast the aircraft is drifting towards or away from the centreline. The latter can be used by the IAS to determine the appropriate bearing degree to follow and intercept the centreline of the landing runway.

To achieve this, the Bearing Adjustment ANN was introduced to predict the necessary adjustment of the aircraft's bearing based on the drift rate towards or away from the path line to be intercepted. Preliminary empirical testing was conducted to select a suitable drift rate which insures a steady drift towards the centreline (angle of 0 degrees) when the angle is already small (the aircraft is close to the centreline of the landing runway). The drift rate should not be fast enough to cause large overshooting which could jeopardise the attempt to align the aircraft with the runway's centreline during the risky final approach and landing flight phases. Since the IAS is capable of intercepting the centreline steadily in calm weather conditions based on the method introduced and explained in the previous chapter, which updates the bearing degree to be followed based on the angle (how far the aircraft is from the path line or the centreline of the landing runway) until the desired bearing is achieved. Therefore, the drift rate of the angle when intercepting the runway's centreline in calm weather, and captured continuously to determine its value which was found to be around 0.0025 degrees every decisecond when the aircraft is close to the centreline (the angle between the aircraft and the centreline is less than 0.1). However, the method explained in the previous chapter generates the same bearing degree given the angle degree between the aircraft and the path line every time regardless of the wind conditions. For instance, if the angle value is x, then, the generated bearing is always y, however, following bearing y might not be enough to counter the effect of crosswind and intercept the centreline, therefore, the new method should take into consideration the rate of change of the angle not just its value. The latter method allows for generating bearing degrees to be followed by the aircraft that suites the drift caused by the wind conditions. The drift rate or the rate of change (RoC) is calculated using (6.1) [148] where

$t_2 - t_1$ is the change in time from timepoint 1 to timepoint 2, and $a_2 - a_1$ is the change in angle degree from angle 1 to angle 2.

$$RoC = \frac{a_2 - a_1}{t_2 - t_1} \qquad (6.1)$$

Then, the result is added to the difference between the bearing of the path line to be intercepted and the aircraft's current bearing to generate the required bearing to be followed. The difference between the latter and the current bearing of the aircraft is fed to the modified Aileron ANN from prototype 3, which now takes the difference as input, and predicts through its output neuron, the appropriate control command to the ailerons, to bank, and intercept the path line. Fig. 6.1 illustrates the Bearing Adjustment ANN, the Ailerons ANN, and the method used to generate the necessary bearing degree to intercept the centreline of the landing runway.

The go-around manoeuvre is performed to abort landing by going to takeoff thrust levels, pulling up to climb, and retracting the landing gear [1]. This is performed when the pilot decides that proceeding with landing might be unsafe, and therefore, it is favourable to climb, go around through a given flight course which brings the aircraft back to the point that precedes the final approach phase, and reattempt landing [1].

Landing safety check techniques are used to ensure that the aircraft is within safe landing conditions, otherwise, go-around is initiated. These techniques, such as the Runway Overrun Prevention System (ROPS)[4] from Airbus, analyse multiple parameters continuously including the available landing runway data and condition to ensure safe landing.

During final approach and just before touchdown, and at a specific altitude that ensures the possibility for the aircraft to climb safely before touchdown, the Flight Manager of the IAS initiates the continuous landing safety check. Although executing a go-around can take place at different altitudes before landing or just after touchdown as well based on the decision of the captain [1], for this work, the selected altitude at which this process starts is equal to or greater

[4] Airbus ROPS. http://www.aircraft.airbus.com/support-services/services/flight-operations/fuel-efficiency-and-runway-overrun-protection-systems/ [accessed 2017]

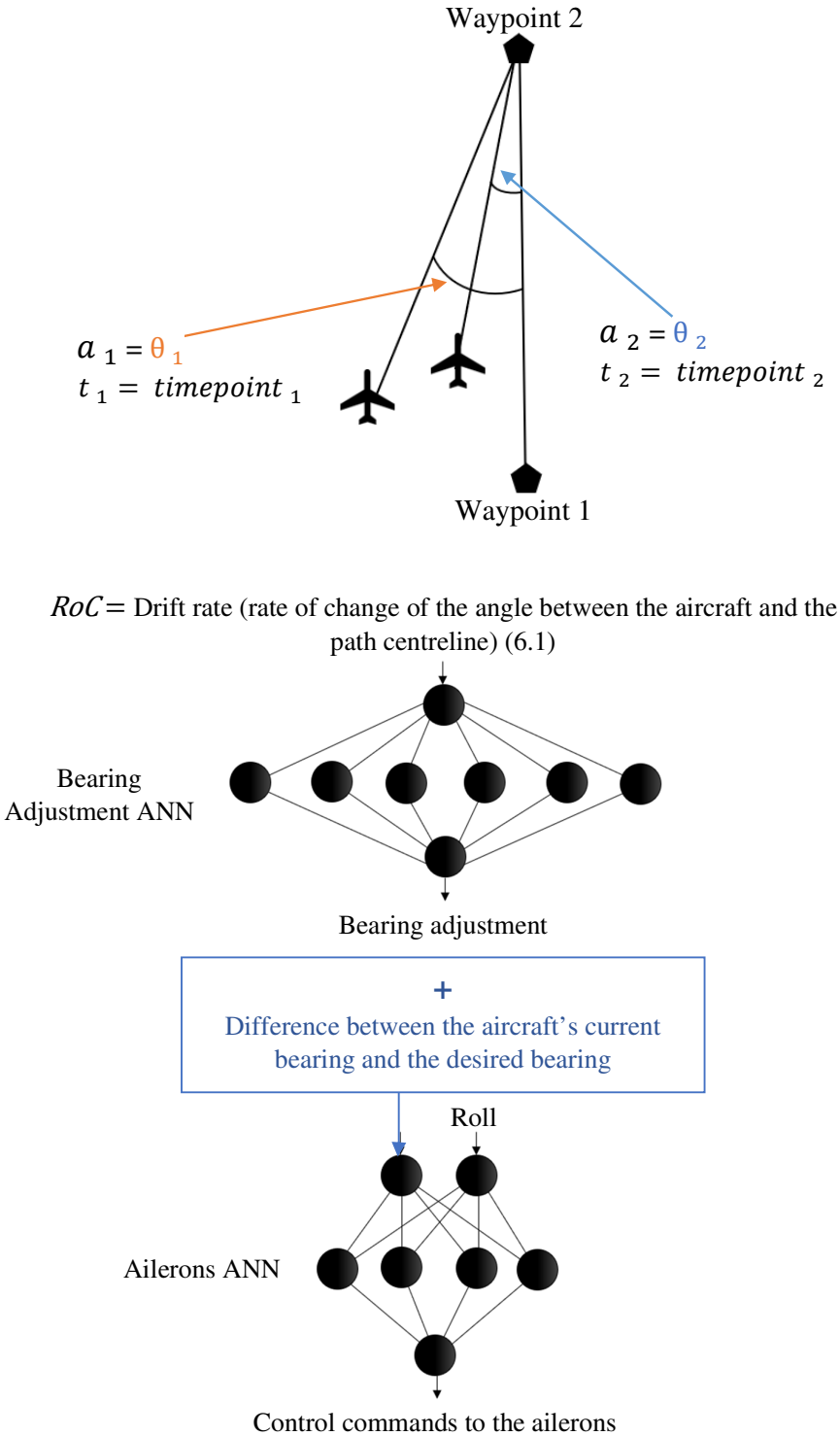

Fig. 6.1. The Bearing Adjustment ANN which takes the angle rate of change as input, and outputs the bearing adjustment value, and the Ailerons ANN which takes the output of the Bearing Adjustment ANN plus the bearing difference between the aircraft's current bearing and the desired bearing along with the roll degree, and outputs control commands to the ailerons.

than 60 (ftagl) based on preliminary empirical testing that was conducted to detect the minimum altitude from which the aircraft can takeoff and climb without touching the ground. The reason for this is to test the agility of the IAS when shifting from the final moments of the final approach flight phase where drag is high because of the low speed, fully extended flaps, and the extracted gear, to takeoff and climb which requires overcoming the existing drag. First, the Flight Manager checks if the angle between the aircraft and the centreline of the landing runway is less than a specific degree based on the runway's width. Then, it checks if the beginning of the landing runway has been reached. Finally, it checks if the remaining distance to the end of the runway is safe for landing. The parameters used during this checking process can be modified based on the available information about the landing runway such as its width and length. If the Flight Manager detects an unsafe landing, it generates a go-around flight course based on the available GPS coordinates as Fig. 6.2 illustrates, changes the flight status from final approach to takeoff, and activates the takeoff ANN. Fig. 6.3 illustrates the process which the Flight Manager follows to handle the go-around process. Fig. 6.3 shows the threshold values that were used for this work which were identified by moving the aircraft (taxi) manually on the chosen landing runway while measuring the angle to detect the maximum angle value that still falls within the runway and do not cross its side edge, in addition to the runway's length measurements. The beginning and end of the landing runway were identified by subtracting its length from the distance to the final GPS waypoint which falls at the end of the landing runway.

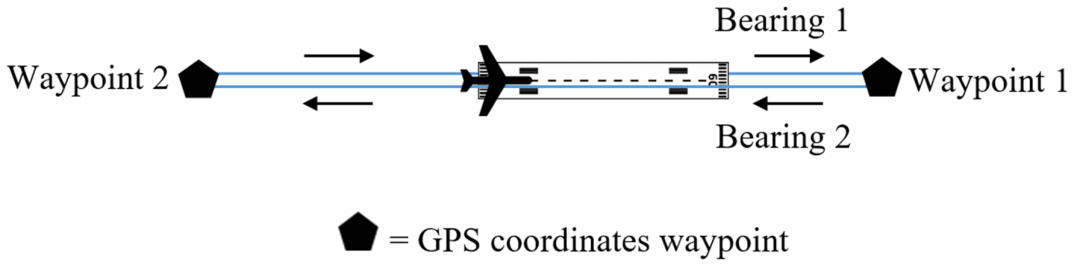

Fig. 6.2. The generated go-around flight course represented by the blue lines. The aircraft navigates to waypoint 1, then to waypoint 2, and finally, back to the landing runway.

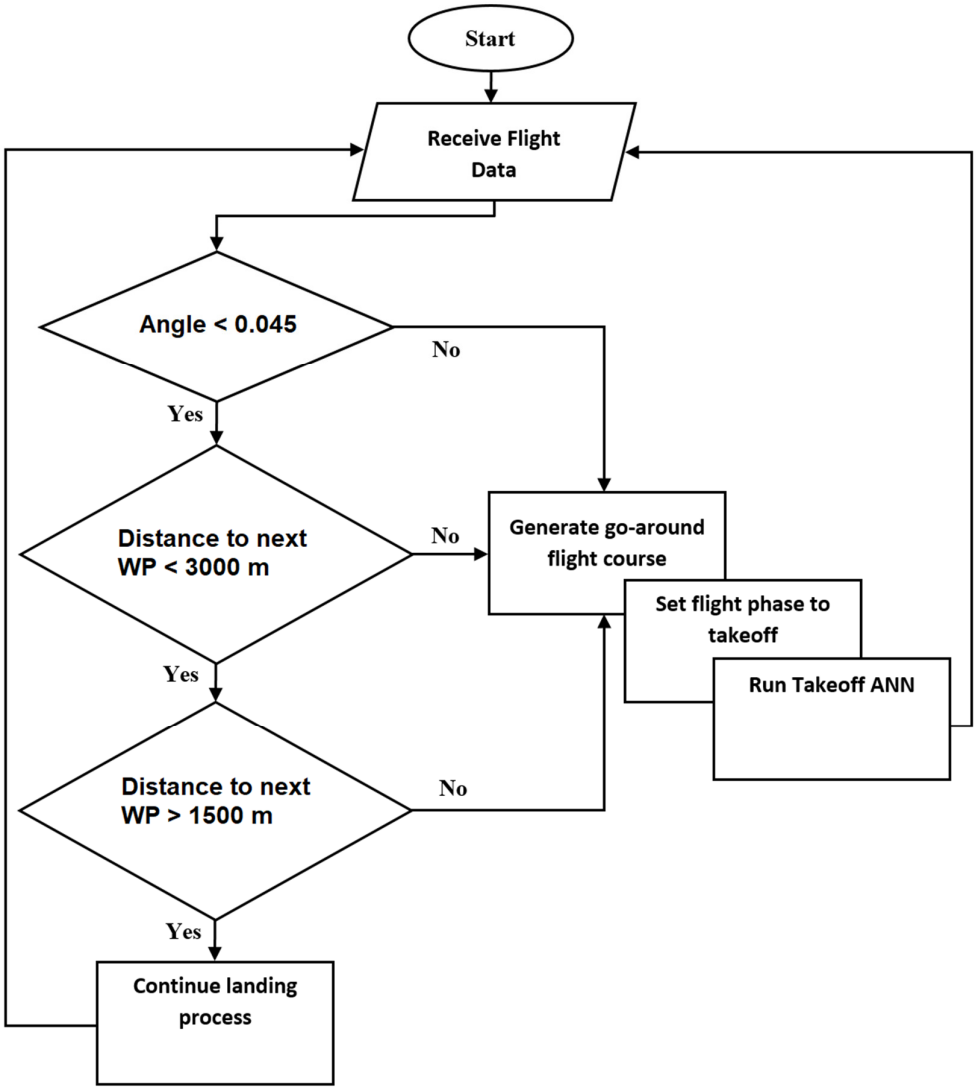

Fig. 6.3. A Flowchart illustrating the process which the Flight Manager program follows to check the landing conditions, and initiate a go-around if necessary. This illustrates the new landing components which are added to the other normal and emergency flight components of the Flight Manager program illustrated in Fig. 5.8.

6.1 Experiments on Prototype 4

This section discusses the experiments conducted on the Bearing Adjustment ANN which aids the Aileron ANN to intercept a path line under severe weather conditions. This section also discusses the experiments conducted on performing go-around.

The experiments were conducted under severe weather conditions with the presence of strong crosswind component (sometimes beyond the maximum demonstrated crosswind landing for the Boeing 777 [128]), wind shear, gust, and turbulence as table 6.2 shows.

To assess the effectiveness of the proposed approach, the Intelligent Autopilot System was tested in two experiments:

1. Path line interception during final approach

2. Go-around

The simulated aircraft used for the experiments is a Boeing 777 as it was intended to experiment using a complex and large model with more than one engine rather than a light single-engine model. The experiments are as follows:

6.1.1 Path Line Interception During Final Approach

The purpose of this experiment is to assess the behaviour of the IAS when intercepting a path line that represents the centreline of the landing runway during final approach under severe weather conditions.

6.1.1.1 Data Collection

No demonstration by the human teacher was provided for the Bearing Adjustment ANN. Instead, a synthetic dataset was generated that comprises pilot experience in a different form which is a table of values gained via testing as opposed to raw input values gained via flying in the simulator by a human demonstrator. The reason for this is the impracticality of gathering demonstration data for this specific task as the human pilot was not able to provide consistent enough values given the lack of piloting experience. In addition, this was applied with the intention to test the ability of the ANNs of the IAS to learn from different types of training datasets that do not necessarily comprise demonstrations by human teachers, but rather, synthetically generated patterns representing the desired behaviour that the IAS should learn. Although the mapping of the inputs and outputs shown in table 6.1 which are used to train the Bearing Adjustment ANN could be achieved using a different method such as conditional statements (If statements for instance), using an ANN is a faster option compared to manually coding the mapping. Furthermore, using an ANN has the advantage of utilizing generalization which is needed to handle inputs that were not included in the training dataset, which other approaches such as conditional statements cannot handle. This approach can also be applied if it is required to rapidly teach the IAS a behaviour without having to collect human pilot demonstrations, by providing the necessary data that represents the desired pilot performance without having to extract and process the data provided from a human demonstration in the simulator.

TABLE 6.1
THE SYNTHETIC DATASET GENERATED FOR THE BEARING ADJUSTMENT ANN. THE ANN IS TRAINED TO MAINTAIN A DRIFT RATE OF 0.0025.

Drift Rate	Bearing Adjustment
0.005	-0.5
0.0045	-0.4
0.004	-0.3
0.0035	-0.2
0.003	-0.1
0.0025	**0**
0.002	0.1
0.0015	0.2
0.001	0.3
0.0005	0.4
0	0.5
-0.0005	0.6
-0.001	0.7
-0.0015	0.8
-0.002	0.9
-0.0025	1

data, which could take a relatively longer time. Based on preliminary empirical testing, a small training dataset was generated for the Bearing Adjustment ANN as Table 6.1 shows.

6.1.1.2 Training

For this experiment, the Bearing Adjustment ANN was trained until a low Mean Squared Error (MSE) value was achieved (below 0.01). Since single-hidden-layer ANNs are used for training, and since the training datasets are small given that a single demonstration was provided, training requires a short time to be completed (under 10 minutes).

6.1.1.3 Autonomous Control

After training the ANN on the relevant training dataset, the aircraft was reset to the runway in the flight simulator, and the IAS was engaged to test the ability of intercepting a final approach and landing path line under severe weather conditions autonomously. After the IAS took the aircraft airborne, and navigated to the destination airport, the output of the Bearing Adjustment ANN was used to assist the Aileron ANN to intercept the final approach path line. This was repeated 50 times under different and random weather conditions as Table 6.2 shows, to assess consistency. In addition to strong wind, it was intended to test the IAS in the presence of other weather conditions that include medium turbulence which is the intensity that does not

TABLE 6.2
THE DIFFERENT WEATHER CONDITIONS USED FOR THE FINAL APPROACH PATH LINE INTERCEPTION EXPERIMENT.

Attempts Count	Wind Speed (knots)	Wind Gust (knots)	Wind Direction (degrees)	Wind Shear (degrees)
10	20	12	0	20
10	23	14	180	20
10	27	15	90	22
10	27	15	270	22
10	50	0	90	0

cause the aircraft to be uncontrollable or even damage the aircraft's structure[5]. Based on empirical testing in the simulator, a turbulence intensity value of around 0.015 provides the desired medium level intensity. In addition, it was found that a rain precipitation value of around 0.3 mm guarantees a wet runway. These turbulence and precipitation values were used uniformly throughout the experiments in this chapter.

6.1.2 Go-around

The purpose of this experiment is to assess the behaviour of the IAS when performing go-around (aborting landing, and flying back to the final approach waypoint) autonomously.

6.1.2.1 Data Collection

No demonstration by the human teacher was provided for this experiment since there was no new skills to be learned. The main purpose of this experiment is to test the ability of the Flight Manager to check the required threshold variables as Fig. 6.3 and take the appropriate decision accordingly.

6.1.2.2 Training

For this experiment, the same approach used in prototype 3 to navigate autonomously from a given point A to a given point B is applied. Therefore, no additional training was required.

6.1.2.3 Autonomous Control

The aircraft was reset to the runway in the flight simulator to test the autonomous go-around task. Just before touchdown, deviation from the path line is induced manually by stopping the IAS, and manually engaging the ailerons by the human pilot to deviate from the path line. Then, the IAS is started immediately. This approach was applied since the IAS excelled at landing

[5] Turbulence

https://www.weather.gov/source/zhu/ZHU_Training_Page/turbulence_stuff/turbulence/turbulence.htm [accessed 2019]

within the safe zone of the landing runway regardless of how severe the weather conditions as long as these conditions are not exaggerated to a no-fly condition. To assess consistency, this was repeated 10 times under different and random weather conditions with minimum wind speed of 20 knots up to 35 knots, and random directions between 0 and 360 degrees including shear of 20 degrees.

6.2 Results of Experiments on Prototype 4

The following section describes the results of the conducted tests.

6.2.1 Experiment 1 (Path line interception during final approach)

One model was generated for the Bearing Adjustment ANN with an MSE value of 0.0089. Utilizing the output value of the Bearing Adjustment ANN to enhance the path line interception performance, resulted in the system flying the aircraft using a technique known as crabbing, where although the aircraft flies in a straight line, the nose of the aircraft is pointed towards a bearing different from the bearing of the landing runway's centreline due to wind conditions as Fig. 6.4 illustrates. Unlike other systems where this technique must be explicitly hard-coded, here, the IAS naturally discovered the technique itself.

Figs. 6.5a, 6.5b, and 6.5c illustrate the different bearings the IAS followed under random severe weather conditions as table 6.2 shows, compared with the bearing of the landing runway (326 degrees), where Fig. 6.5a show bearings followed when the aircraft was pushed to the left side of the landing runway's centreline, which happens in the presence of east crosswind for example, and vice versa (Fig. 6.5b). Fig. 6.5c show the bearings the IAS followed under a sustained weather condition with a constant crosswind of 50 knots at 90 degrees. Fig. 6.6 illustrates the average rate of change of the angle when drifting towards the path line. Fig. 6.7 illustrates the angle representing the difference between the aircraft's position, and the centreline of the landing runway based on finding the width of the runway as explained above and shown in Fig. 6.3 (angle < 0.045 and > -0.045). Figs. 6.8 and 6.9 illustrate the crabbing manoeuvre performed by the IAS.

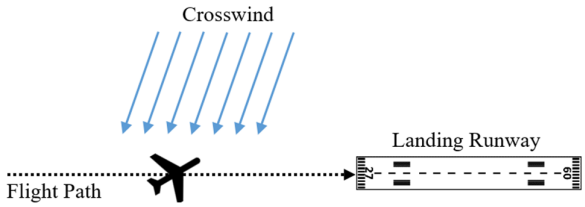

Fig. 6.4. An illustration of crabbing, where although the aircraft flies in a straight line, the nose of the aircraft is pointed towards a bearing different from the bearing of the landing runway's centreline.

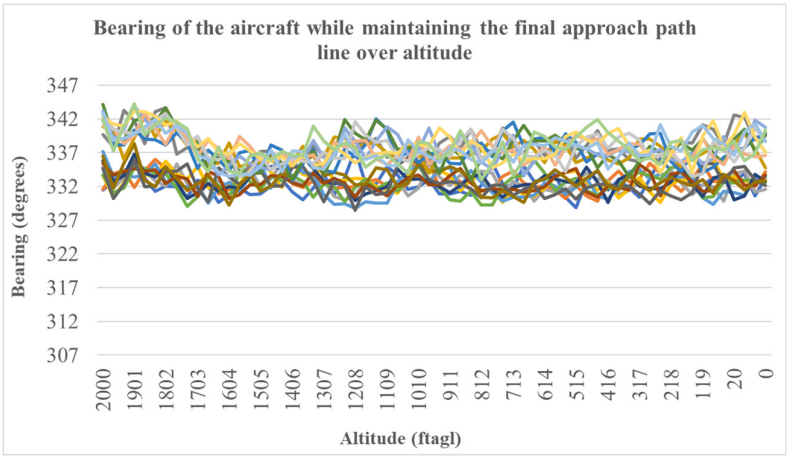

Fig. 6.5a. 20 attempts showing Aircraft bearings (crabbing) during final approach under severe weather conditions at table 7.2 shows, compared with the bearing of the landing runway (326 degrees). The aircraft was pushed to the left side of the landing runway's centreline.

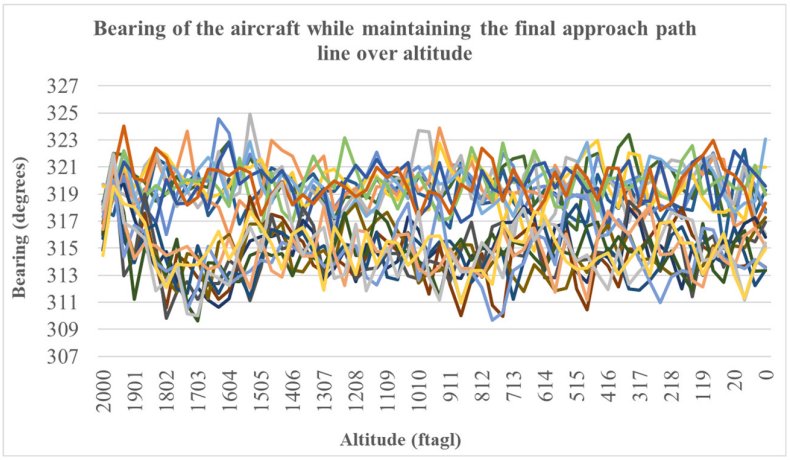

Fig. 6.5b. 20 attempts showing Aircraft bearings (crabbing) during final approach under severe weather conditions at table 7.2 shows, compared with the bearing of the landing runway (326 degrees). The aircraft was pushed to the right side of the landing runway's centreline.

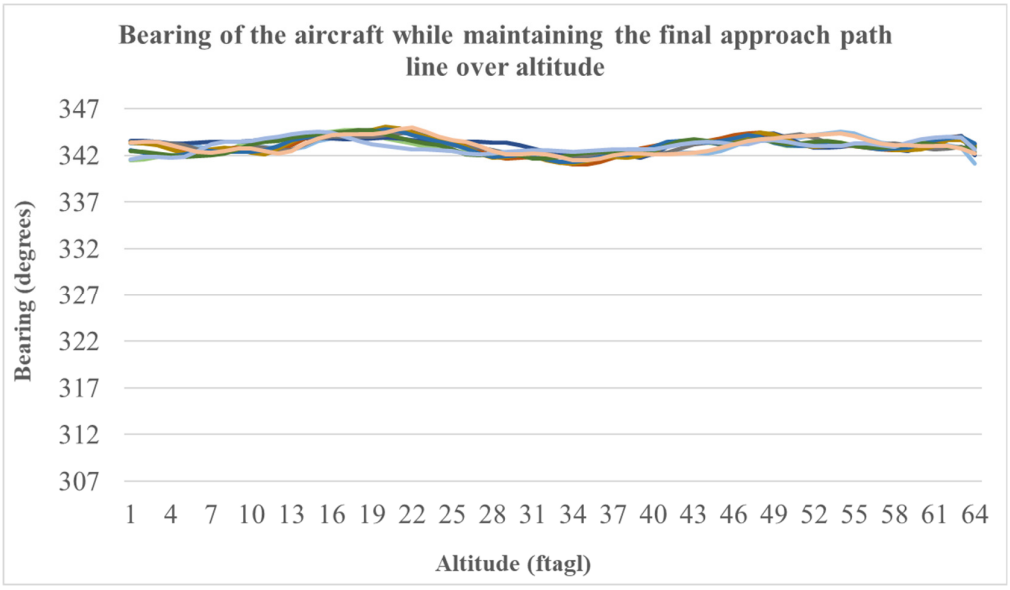

Fig. 6.5c. 10 attempts showing Aircraft bearings (crabbing) during final approach under severe weather conditions at table 7.2 shows, compared with the bearing of the landing runway (326 degrees). The aircraft was pushed to the left side of the landing runway's centreline.

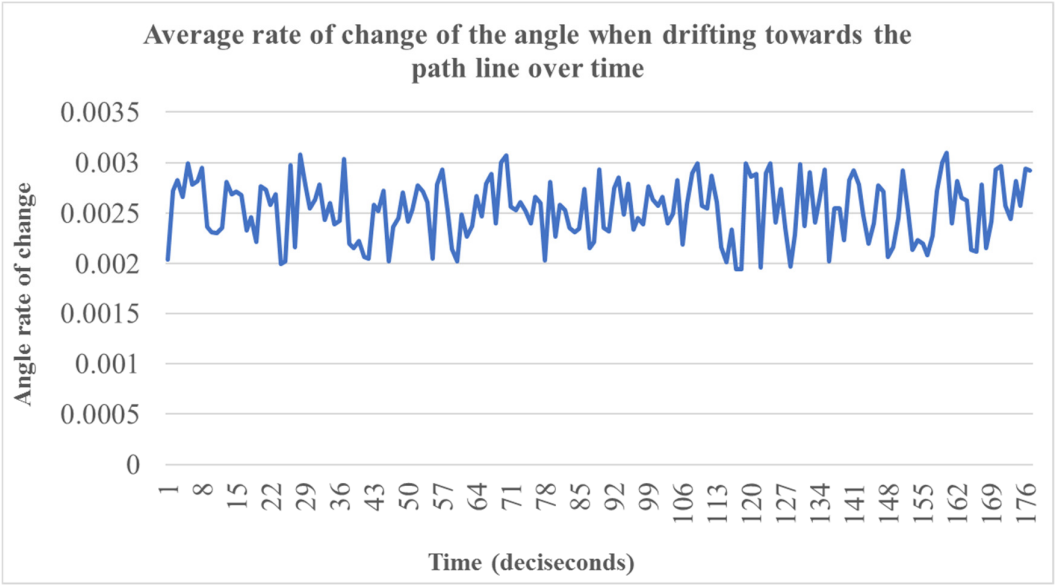

Fig. 6.6. The average rate of change of the angle when drifting towards the path line in the presence of random and severe weather conditions at table 6.2 shows, compared with a desired rate of change of 0.0025 degrees every decisecond.

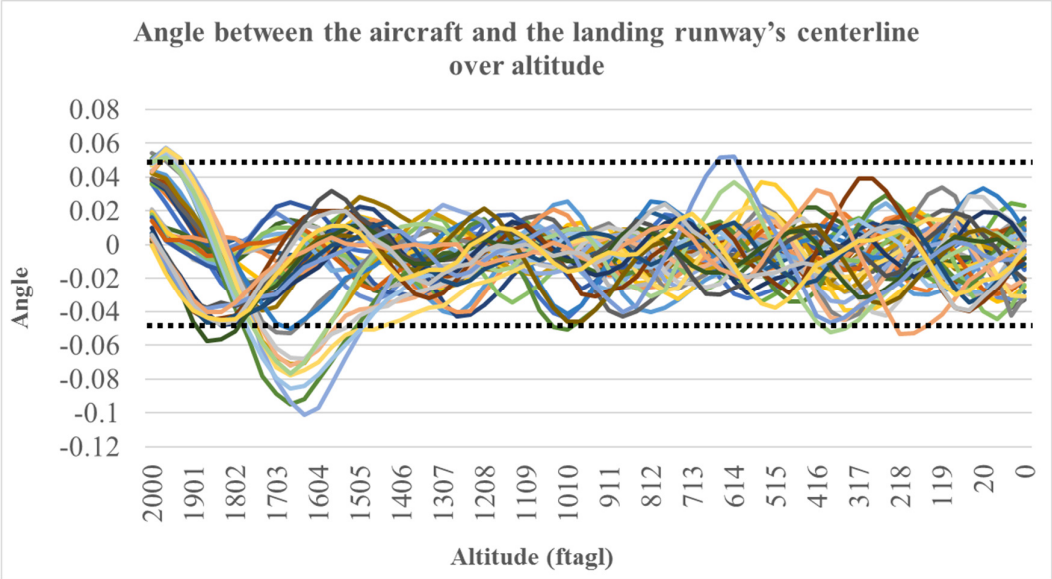

Fig. 6.7. 50 lines showing angle values between the aircraft's position, and the centreline of the landing runway (0 degrees) of all the attempts illustrated in Figs. 6.5a, 6.5b, and 6.5c. Based on the width of the landing runway used in the experiments, a safe touchdown angle is between 0.045 and -0.045, which is the area between the dotted lines (landing runway's safe touchdown zone).

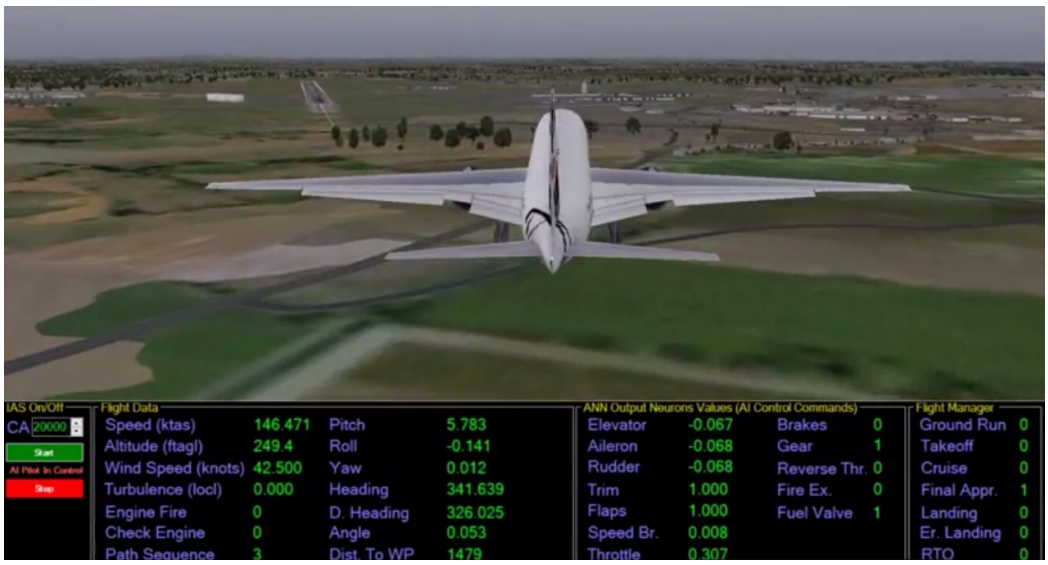

Fig. 6.8. The crabbing manoeuvre performed by the IAS during final approach, and before touchdown.

Fig. 6.9. The crabbing manoeuvre performed by the IAS on touchdown.

6.2.2 Experiment 2 (Go-around)

No new models were generated for this experiment. Fig. 6.10 illustrates the flight paths that the IAS followed autonomously back to the landing runway. Since no strict go-around path was applied, the IAS followed two different paths based on the aircraft's location with respect to the landing runway's centreline, where a position on the right of the runway due to wind blowing from the left would cause the IAS to bank right towards the next waypoint, and vice versa.

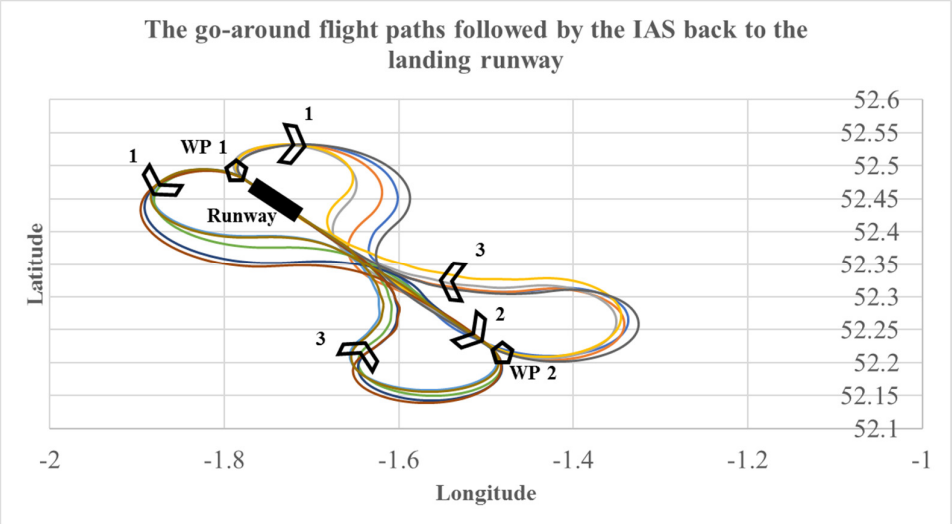

Fig. 6.10. The 10 go-around flight paths followed autonomously by the IAS back to the landing runway. The aircraft navigates to waypoint 1, then to waypoint 2, and finally, back to the landing runway. the IAS followed two different paths based on the aircraft's location with respect to the landing runway's centreline. Birmingham airport (BHX) was used.

6.3 Analysis

As can be seen in Figs. 6.5a, 6.5b, and 6.5c (Path line interception during final approach experiment), the IAS was able to produce a natural crabbing behaviour in a direction that is perpendicular to the constantly changing speed and direction of wind without being explicitly trained to do so. In addition, the IAS was able to handle persistent strong crosswind of 50 knots at 90 degrees which is beyond the demonstrated crosswind landing of a Boeing 777 as Fig. 6.5c shows. Keeping the angle rate of change close to 0.0025 degrees despite the random severe weather conditions proved the effectiveness of the Bearing Adjustment ANN as Fig. 6.6 (Path line interception during final approach experiment) illustrates. In all the attempts, the IAS was able to touchdown within the safe landing zone with respect to the centreline of the runway as Fig. 6.7 (Path line interception during final approach experiment) illustrates. This compares well with prototype 3 of the IAS without the Bearing Adjustment ANN, which was unable to land under the same conditions as Figs. 5.19, 5.20, and 5.21 (chapter 5) illustrate. Under most weather conditions the IAS was able to land successfully within the safe zone of the landing runway to the point where go-around manoeuvres were not needed, therefore, manual intervention was required to induce a go-around manoeuvre by stopping the IAS just before touchdown, manually banking the aircraft away from the centreline, then restarting the IAS immediately as Fig. 6.11 shows. The system was able to detect unsafe landings through the Flight Manager, and followed go-around paths back to the landing runway under random severe weather conditions successfully as Fig. 6.10 (go-around experiment) illustrates.

However, at this point, the IAS is not yet capable of piloting an aircraft in manner comparable with experienced human pilots of airliners since so far, the system learned from the author of this work who is not a pilot. Therefore, the next prototype of the IAS should be able to accurately mimic the behaviour of such experienced human pilots during the different flight phases from takeoff to landing. The latter requires the IAS to acquire additional capabilities that are not yet available. For instance, the IAS is not yet capable of accurately maintaining different altitudes as Fig. 4.9 illustrates (chapter 4), it is not able to maintain different speeds and manage them properly throughout the different flight phases, and it is not able to maintain different climb/sink rates since these capabilities were not added yet but are covered in the next chapter. In addition, the IAS does not manipulate the flaps correctly since it was trained to extend them fully when a certain altitude is reached before landing as Fig. 5.15 shows (chapter 5) while it should extend them gradually depending on the altitude [1]. Training the IAS to extend the flaps correctly is a task that was left for the next chapter which intends

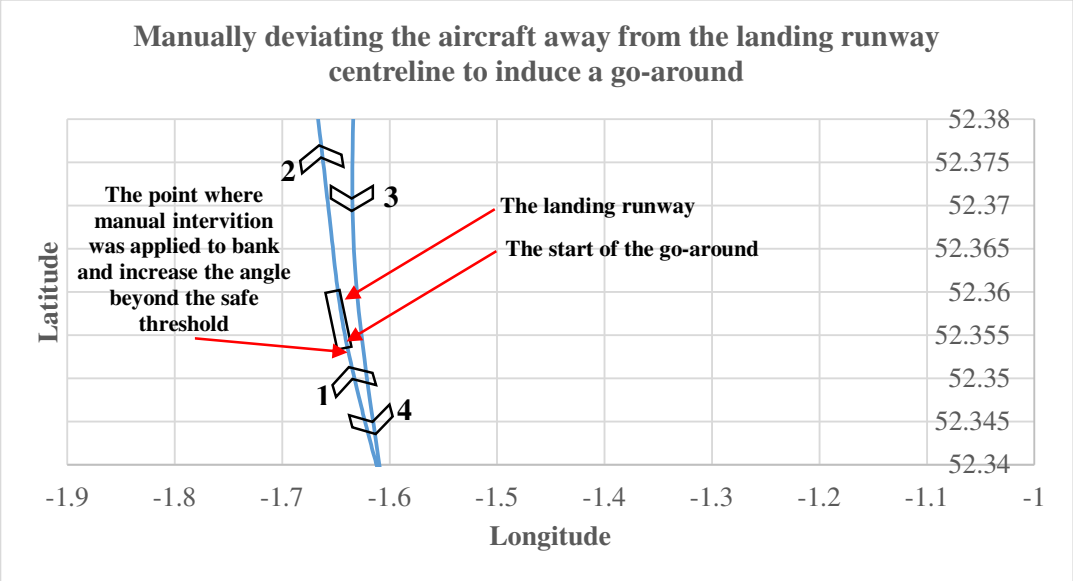

Fig. 6.11. An example of manually deviating the aircraft by the user away from the landing runway before landing to induce a go-around manoeuvre. This was done by stopping the IAS before landing, banking (in this example to the right) to increase the angle between the aircraft and the centreline of the landing runway beyond the safe threshold, and restating the IAS immediately to detect the unsafe angle and initiate a go-around.

While the current IAS can handle severe crosswinds, it is still useful to explore the limits of its ability. To achieve this, an additional series of test flights were conducted representing crosswind landings in extreme weather conditions with wind speeds up to 70 knots. Fig. 6.12 shows the results of trying to maintain the centreline of the landing runway (0 degrees) in such extreme wind conditions. It is clear that the current prototype of the IAS is not capable of handling such extreme conditions. The next chapter will explore further enhancements to the IAS to enable it to handle even these conditions.

6.4 Summary

To summarize, the objective of teaching Artificial Neural Networks how to handle severe weather landing, was achieved by introducing the Bearing Adjustment ANN which takes the drift rate of the aircraft away from the path line, into account. The latter process enabled the Ailerons ANN to handle severe weather conditions such as strong crosswind during final approach and landing. In addition, extending the capability of the Flight Manager program to detect unsafe landings, enabled the IAS to perform go-around manoeuvre, by generating a go-around flight course, and managing the ANNs to re-attempt landing. This provides additional evidence to support the hypothesis of this work aimed towards proving the possibility to teach a flight control system piloting skills.

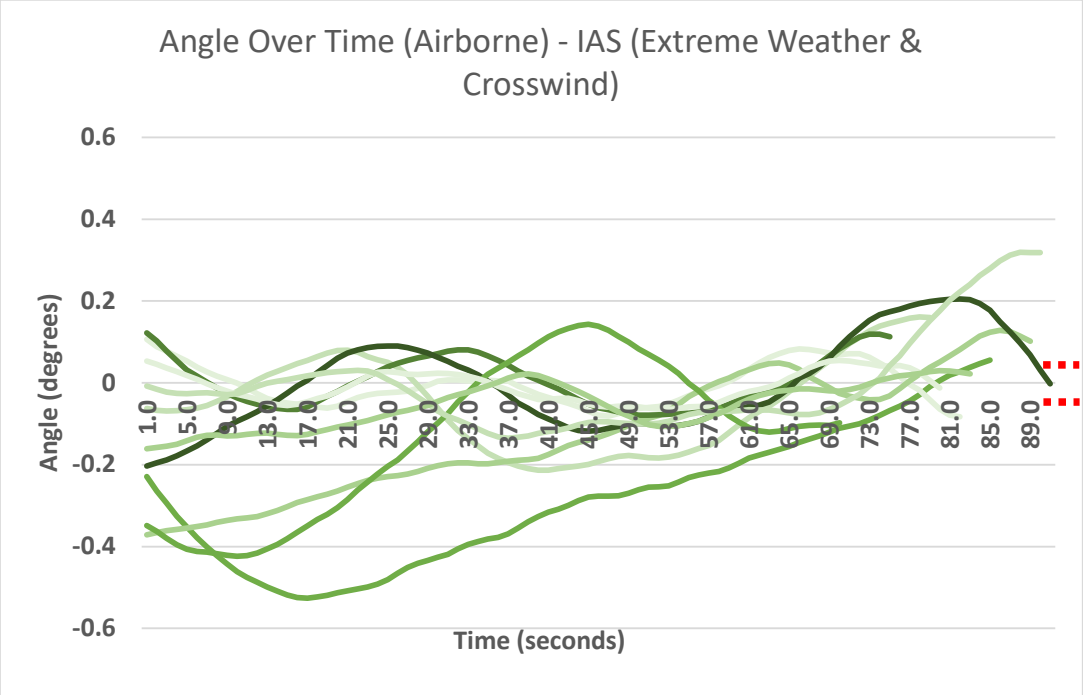

Fig. 6.12. The angle between the aircraft and the centreline of the landing runway during 10 attempts to land in extreme weather conditions. The angle must be between 0.045 and -0.045 degrees especially during the last moments of the final approach to ensure landing within the safe touchdown zone of the landing runway as the two dashed red lines show on the right side. The extreme weather conditions include 90 degrees crosswind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and strong turbulence.

7. PROTOTYPE 5 (LEARNING FROM EXPERIENCED PILOTS)

After the first, second, third, and fourth objectives of using Artificial Neural Networks to learn basic flying, handle emergency situations, learn complex flying including navigation and landing, and handle landings in severe weather conditions were achieved, the purpose of the fifth prototype was to achieve the objective of piloting an aircraft in a manner comparable with experienced human pilots of airliners. Although the previous prototypes presented cockpit autonomy capabilities, the Intelligent Autopilot System (IAS) did not fully behave like an experienced human pilot of an airliner especially when manipulating the different control surfaces to maintain desired parameters such as altitude and the final approach glideslope. This is because the IAS was trained using amateur pilot data in order to demonstrate the validity of the methods, while waiting for access to experienced pilots. In addition, the previous prototypes did not have the ability to maintain desired speeds, climb/sink rates, and correctly control the flaps settings. Furthermore, it was decided to investigate the possibility of handling not just severe weather during landing, but extreme weather conditions that are beyond the current limits and abilities of autopilots (Autoland feature) and human pilots as well.

To achieve the fifth objective mentioned above, the IAS was enhanced to mimic the behaviour of experienced human pilots of airliners by redesigning the system's Artificial Neural Networks and adding new ones to learn from new training data collected from a demonstration performed by an experienced human captain. The demonstration was used to identify the appropriate control surfaces and interfaces that are used by experienced human pilots to perform the different piloting tasks throughout the different flight phases. The tasks include the ability to control the different settings of the flaps during takeoff, approach, and final approach, maintain a certain pitch during takeoff, maintain manually selected climb rates during climb and cruise, maintain manually selected altitudes during cruise, maintain manually selected sink rates during descent, maintain the standard glideslope during approach and final approach, and maintain manually selected speeds during the different flight phases.

In this chapter, the IAS (prototype 5) was trained with Oman Air through a collaboration project to achieve the desired autonomous behaviour that can be compared with the behaviour of experienced human pilots of airliners. The human teacher who provided the demonstrations is Captain Khalid Al Hashmi, Senior Manager Crew Training at Oman Air. The simulated aircraft used for the experiments is a certified Boeing B787 Dreamliner model which is the aircraft that Captain Al Hashmi usually flies in real-life. Fig. 7.1 illustrates the IAS components

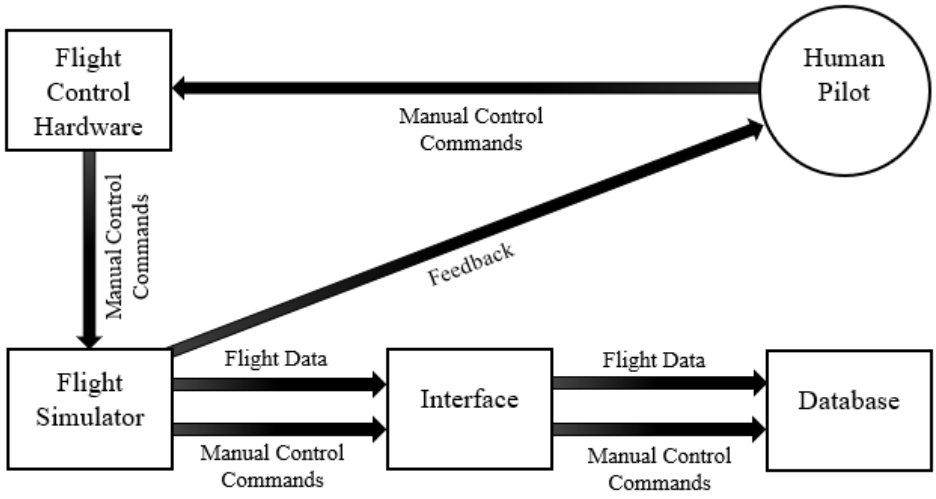

Fig. 7.1. Block diagram illustrating the IAS components used during the pilot data collection step.

used during the pilot data collection step. Since the design approach of the IAS which utilizes Supervised Learning and many small single-hidden-layer ANNs requires single demonstrations of the tasks to be learned, Captain Al Hashmi provided one demonstration of a short flight from one airport to another in X-Plane. Captain Al Hashmi took off from London Heathrow (EGLL), cruised at 10,000 ft, then landed in Birmingham (EGBB). Captain Al Hashmi followed the standard piloting procedures where he started the ground-run phase on the takeoff runway, rotated, and maintained a 15 degrees pitch angle during takeoff. Then, he engaged the aircraft's autopilot to climb to the cruise altitude of 10,000 ft, to maintain a cruise speed of 240 knots, and to follow the preloaded flight path using GPS waypoints. Immediately after reaching the Top of Descent (TOD) point, Captain Al Hashmi initiated the approach flight phase by updating the speed parameter in the aircraft's autopilot to 205 knots and starting the decent to follow the standard 3 degrees glide slope. Then, he updated the speed parameter to reach the landing speed of 150 knots before reaching the final approach flight phase. During the latter flight phases, he engaged the flaps at different altitudes to extend them to certain degrees accordingly. Finally, after the speed reached 150 knots, and at around 1,500 ft, he disengaged the autopilot, and took full control of the aircraft to continue maintaining the landing speed and the 3 degrees glideslope until touchdown.

In the demonstration provided by the experienced captain, executing the different piloting tasks was done using different techniques which utilized different control interfaces and

surfaces compared to the techniques that the IAS learned so far. First, during takeoff, the captain used the same control surfaces which are the elevators, however, he used them to maintain a pitch of 15 degrees which the IAS was not trained to maintain. Therefore, ANN 2 from prototype 1 which handled takeoff should be replaced by a new ANN (Elevators ANN) that can maintain a given pitch degree during takeoff by generating control commands to the elevators as output. The task of maintaining speed was achieved through using the throttle by the standard autopilot after it was engaged and set by the captain to maintain speed. The task of maintaining speed was not added to the capabilities of the IAS as it was intended to add it after receiving the required demonstration by the experienced captain. To achieve this, a new ANN (Throttle ANN) should be designed and trained to handle the task of maintaining speed by generating control commands to the throttle as output. To maintain different climb and sink rates, the standard autopilot used the elevators trim to achieve this task after manually choosing the desired sink or climb rate by the captain. The latter task was not added to the IAS as it was intended to add it after receiving the required demonstration by the experienced captain. To achieve this, a new ANN (Elevators Trim ANN) should be designed and trained to generate control commands to the elevators trim to maintain different climb or sink rates. The task of maintaining altitude was handled differently by the standard autopilot which used the elevators trim as well to perform this task. In comparison, the IAS used the throttle to maintain altitude which although is an acceptable technique for light aircraft, it is not the standard practice for large jets according to the experienced captain. Therefore, the Cruise Altitude ANN from prototype 2 should be replaced by a new ANN (the same ANN that can maintain climb or sink rates, which is the Elevators Trim ANN) that generates control commands to the elevators trim to maintain altitude. During final approach, the captain engaged the approach mode in the standard autopilot which maintained a glideslope of 3 degrees using the elevators trim. Before touchdown, he disengaged the standard autopilot, and continued to maintain the 3 degrees glideslope until touchdown using the same control surface. To achieve this task, a new ANN (Glideslope Elevators Trim ANN) should be developed and trained to handle this task by generating control commands to the elevators trim as output. During takeoff and final approach, the experienced captain used the flaps by manually extending the them to different degrees gradually that are suitable for takeoff and final approach as well. In comparison, the IAS was trained to use the flaps during final approach only by extending them fully before landing, therefore, a new ANN (Flaps ANN) should be designed and trained to control the flaps appropriately during the two different flight phases as per the given demonstration.

However, to avoid any undershooting or overshooting when attempting to reach a desired value such as speed, altitude, or climb rate, the experienced captain and the standard autopilot applied the necessary change using the relevant control interface using magnitudes that are suitable for the difference between the current value and the desired value. For example, when the difference between the current speed and the desired speed is big, the throttle was increased to full, then, lowered gradually as the aircraft approached the desired speed. The latter example shows that using the rate of change (introduced in the previous chapter to aid the Ailerons ANN) is necessary in all the scenarios that require shifting from a given current value to a desired value such as speed. Therefore, for each new ANN an additional ANN should be designed and trained to capture how the rate of change is managed for the different tasks to achieve the desired value. For instance, the input of the new Throttle ANN which controls the throttle to maintain speed should be the output of an ANN (Speed ROC -Rate of Change- ANN) that takes the difference between the current and desired speeds as input, and outputs the desired rate of changed that should be maintained until the desired speed is achieved. These rate of change ANNs should capture how the experienced captain and the standard autopilot manage the rates of change to achieve the desired value without undershooting or overshooting.

The data of interest that was collected and used to train the IAS are the inputs and outputs of the different ANNs illustrated in Fig. 7.2, which were used to identify the appropriate flight data (inputs), and control surfaces and interfaces (outputs) used by experienced human pilots. Fig. 7.2 illustrates how for each new ANN (except the Flaps ANN which does not require following certain rates of change), an additional ANN is developed and trained to handle the rate of change maintenance. Twenty-two feedforward Artificial Neural Networks now comprise the core of the IAS. Each ANN is designed and trained to handle specific control or task. The ANNs that are relevant to this prototype are: the Pitch Rate of Change ANN, the Elevators ANN, the Altitude Rate of Change ANN, the Elevators Trim ANN, the Speed Rate of Change ANN, the Throttle ANN, the Flaps ANN, the Roll ANN, the Ailerons ANN, the Heading ANN, the Rudder ANN, the Glideslope Rate of Change ANN, and the Glideslope Elevators Trim ANN. Table 7.1 describes the inputs and outputs of the newly developed ANNs.

In addition to designing and training the new ANNs that capture the professional behaviour of the experienced captain through the provided demonstration, an additional challenge was tackled by enhancing the capability of an ANN from the previous chapter. Similar to the challenge faced by prototype 3 which could not handle windy conditions due to the method which relied on updating the bearing degree gradually to intercept the centreline based on the

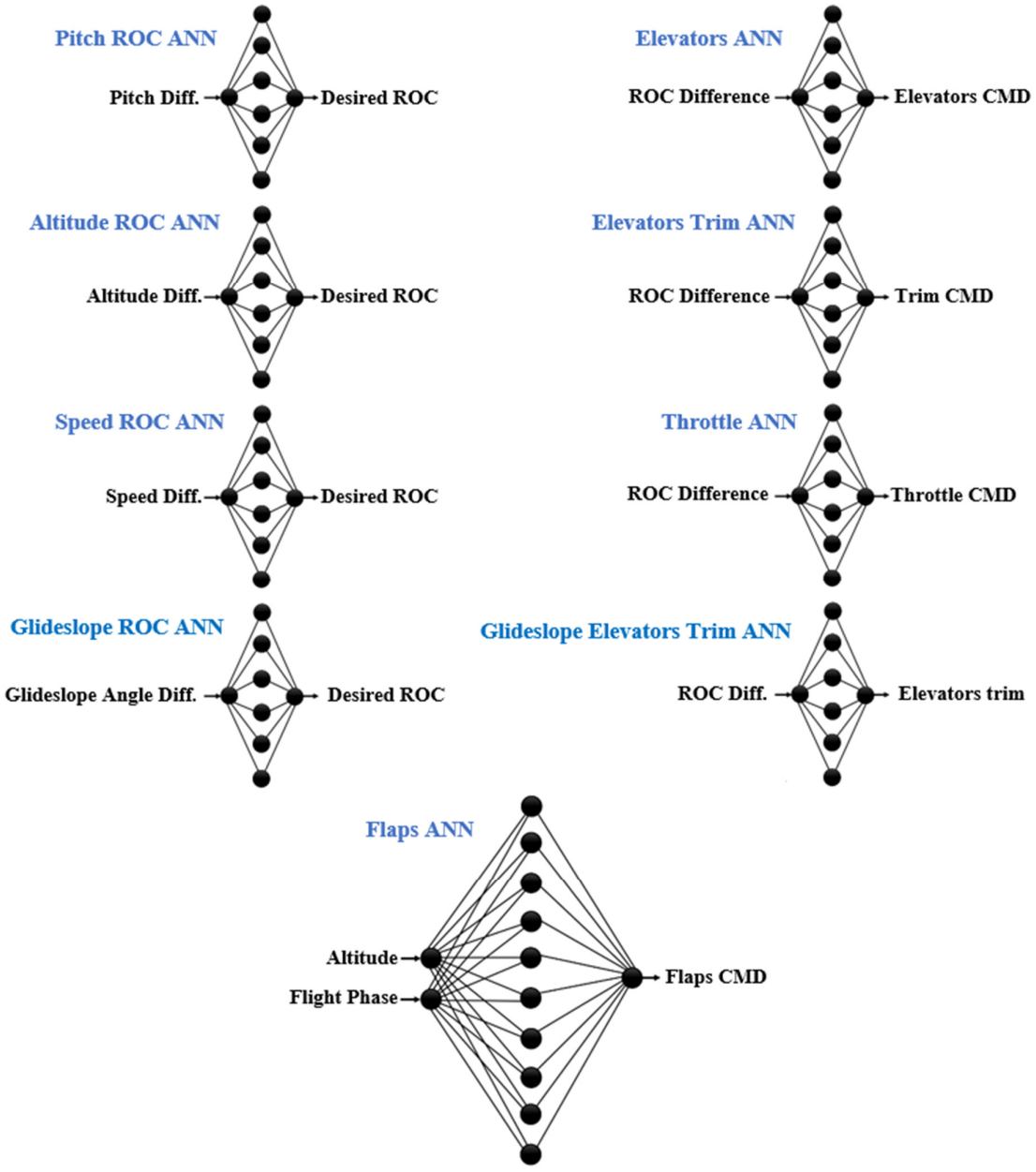

Fig. 7.2. Inputs, outputs, and the topologies of the ANNs relevant to this work. Each ANN is designed and trained to handle a specific task.

value of the angle between the aircraft and the centreline alone without considering the effect of external forces (wind), the method used in prototype 4 (Bearing Adjustment ANN) could not handle extreme wind conditions although it was able to handle severe wind conditions. The same reason applies here as well since updating the bearing degree based on the angle rate of change alone was not enough to handle the extreme external forces. This was due to the fact

that the IAS was not trained to go beyond certain degrees of roll which could provide the necessary momentum to counter the external forces. Therefore, a new method should be developed to continuously increase or decrease the roll degree until the aircraft starts to overcome the external forces, and move towards the centreline. This means that focusing on the roll degree of the aircraft instead of the bearing degree based on the angle rate of change is required to handle such extreme conditions since this method guarantees that the aircraft will keep increasing its roll degree until it intercepts the centreline successfully regardless of how strong the wind is. Therefore, the Bearing Adjustment ANN (from prototype 4) which predicted the necessary adjustment of the aircraft's bearing based on the drift rate (angle rate of change) was replaced with the Roll ANN which takes the rate of change of the angle (as calculated in prototype 6) as input, and predicts the desired roll degree to bank the aircraft towards the path-line by continuously increasing or decreasing the roll degree. The Ailerons ANN (from prototype 3) which took the difference between the bearing of the path line to be intercepted and the aircraft's current bearing was altered to take the difference between the current roll and the desired roll (predicted by the Roll ANN) as input, and predicts the appropriate command to be sent to the ailerons to bank as Fig. 7.3 illustrates.

To apply the technique which relies on the rate of change uniformly across the different relevant ANNs, and as a proactive measure to handle the expected extreme wind on tarmac after landing, the Rudder ANN should be enhanced. The method of altering the bearing (heading) degree used in the previous chapter is introduced in this chapter to the rudder control problem to maintain the centreline of the landing runway after touchdown since this method is sufficient to keep the aircraft aligned with the centreline after touchdown given the low speed, the stability represented by the aircraft being on the ground, and the proven ability -experiments of the previous chapter- of this method to perform well in more difficult problem of keeping the aircraft aligned with the centreline during final approach (airborne) in severe weather conditions. To achieve this, a new ANN (Heading ANN) was designed and trained to take the rate of change of the angle between the aircraft and the centreline as input, and predicts the desired heading degree that the aircraft should follow. In addition, ANN 4 from prototype 1 was modified to take the difference between the output of the new Heading ANN, which is the desired heading degree, and the desired heading or bearing as input, and outputs control commands to the rudder.

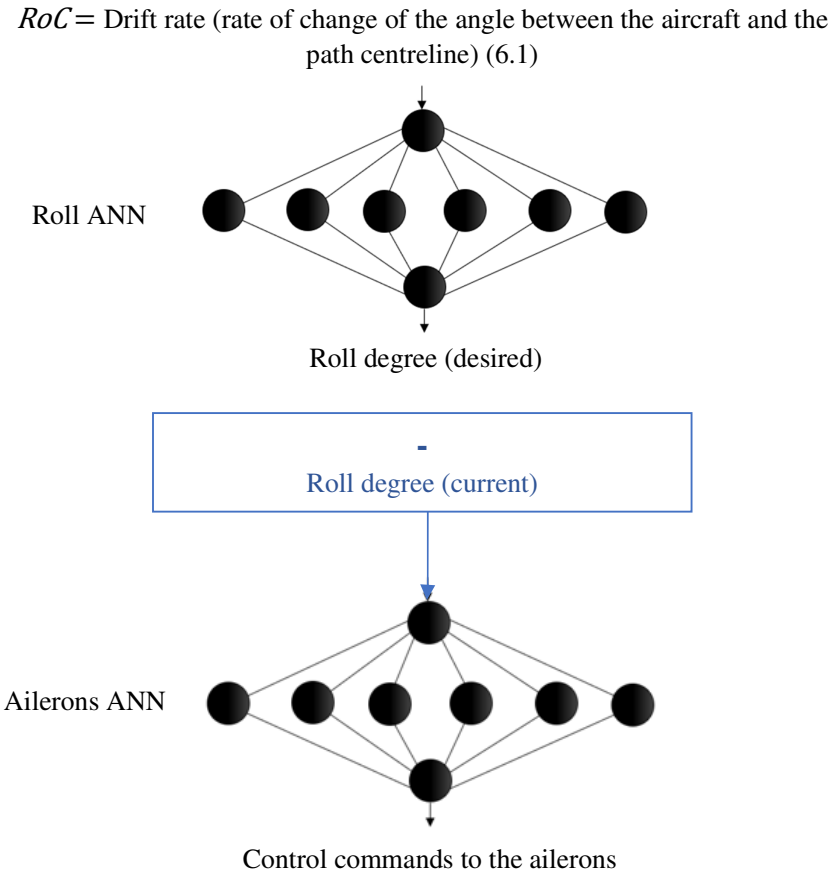

Fig. 7.3. The Roll ANN which takes the angle rate of change as input, and outputs the desired roll degree, and the Ailerons ANN which takes the output of the Roll ANN (desired roll) minus the current roll degree of the aircraft, and outputs control commands to the ailerons.

TABLE 7.1
THE ARTIFICIAL NEURAL NETWORKS DEVELOPED FOR THIS WORK, THE FLIGHT PHASE IN WHICH THEY ARE USED, AND THEIR DESCRIPTION.

Artificial Neural Network	Flight Phase	Description
Pitch Rate of Change ANN (new)	Takeoff	Takes the difference between the aircraft's pitch and the desired pitch as input, and predicts the appropriate rate of change of pitch degrees that is required to reach the desired pitch.
Elevators ANN (enhanced)	Takeoff	Takes the difference between the current rate of change of pitch degrees and the desired rate of change (predicted by the Pitch Rate of Change ANN) as input, and predicts the appropriate command to be sent to the elevators.
Altitude Rate of Change ANN (new)	Cruise	Takes the difference between the aircraft's altitude and the desired altitude as input, and predicts the desired rate of change (climb/sink rate).
Elevators Trim ANN (new)	Cruise	Takes the difference between the current rate of change and the desired rate of change (predicted by the Altitude Rate of Change ANN) as input, and predicts the appropriate command to be sent to the elevators' trim.
Speed Rate of Change ANN (new)	All	Takes the difference between the aircraft's speed and the desired speed as input, and predicts the desired rate of change of speed.
Throttle ANN (new)	All	Takes the difference between the current rate of change of speed and the desired rate of change (predicted by the Speed Rate of Change ANN) as input, and predicts the appropriate command to be sent to the throttle.
Flaps ANN (new)	Takeoff, Approach, and Final Approach	Takes the aircraft's altitude and the flight phase as inputs, and predicts the appropriate command to be sent to the flaps.
Roll ANN (new)	All	Takes the rate of change of the angle between the aircraft and the centreline as input, and predicts the desired roll degree to bank the aircraft towards the path-line.
		Takes the difference between the current roll and the desired roll

Ailerons ANN (enhanced)	All	(predicted by the Roll ANN) as input, and predicts the appropriate command to be sent to the ailerons to bank.
Heading ANN (new)	Landing	Used on the runway to align the aircraft with the centreline of the runway. It takes the rate of change of the angle between the aircraft and the centreline as input, and predicts the desired heading degree that the aircraft should follow on tarmac to be aligned with the centreline of the runway.
Rudder ANN (enhanced)	Final Approach, and Landing	Takes the difference between the current heading and the desired heading (predicted by the Heading ANN), and predicts the appropriate command to be sent to the rudder.
Glideslope Rate of Change ANN (new)	Approach, and Final Approach	Takes the difference between the aircraft's glideslope degree and the desired glideslope degree as input, and predicts the desired rate of change of the glideslope angle that is necessary to align the aircraft with the desired glideslope angle.
Glideslope Elevators Trim ANN (new)	Approach, and Final Approach	Takes the difference between the current rate of change of the glideslope angle and the desired rate of change (predicted by the Glideslope Rate of Change ANN) as input, and predicts the appropriate command to be sent to the elevators' trim.

Fig. 7.4 illustrates how the Flight Manager (from Prototype 2) was enhanced to manage the flight phases as Fig. 7.5 shows by continuously examining the speed and altitude of the aircraft, and the distance to the next waypoint to detect the transition points between the different flight phases according to the demonstration provided by the experienced human pilot. The Flight Manager can now detect the Top of Descent (TOD) point where the aircraft starts the descent towards the destination airport by applying (7.1) which is the standard method for calculating TOD in airliners [149] compared to the previous method introduced in chapter 5. Although the previous method is valid as well, it is not the best practice according to Captain Al Hashmi.

$$TOD = \frac{(\ Altitude\partial_{(cruise)} - Altitude\partial_{(fix)}) \ / \ 100}{Descent_{(angle)}} \qquad (7.1)$$

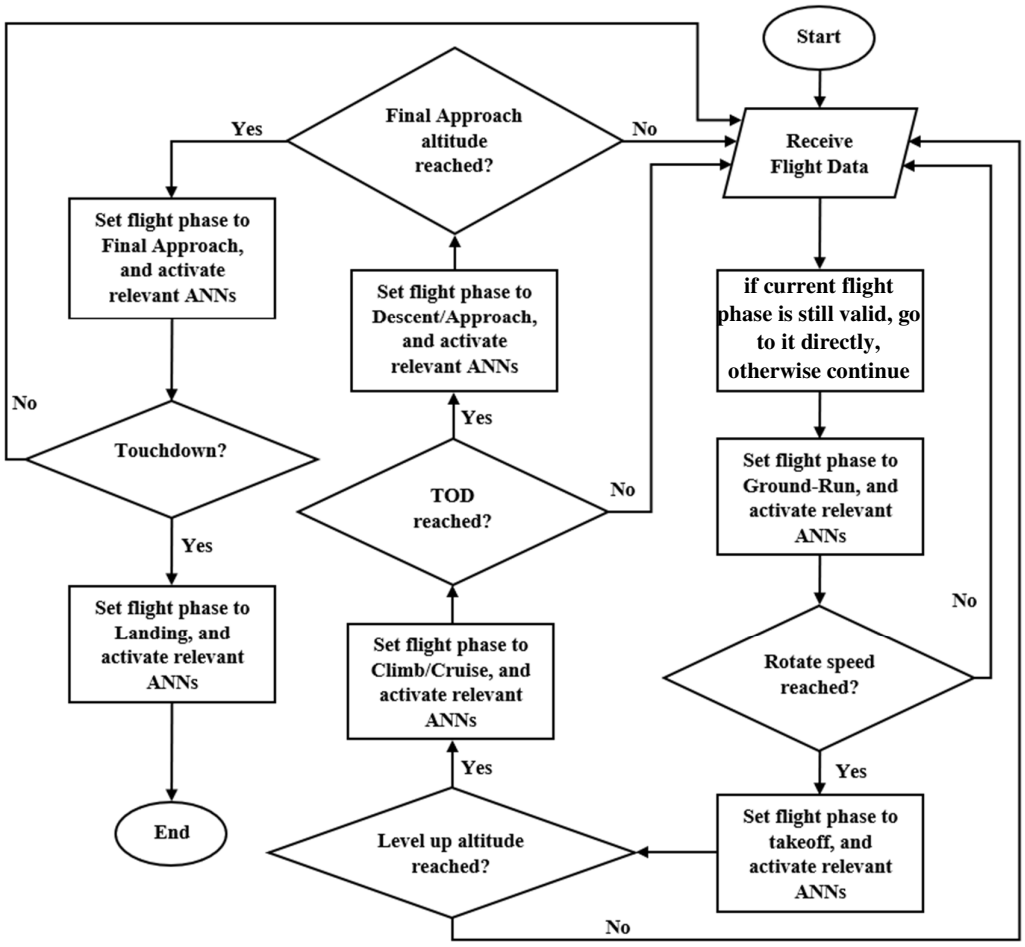

Fig. 7.4. A Flowchart illustrating the process which the Flight Manager program follows to handle the transmission between the different flight phases.

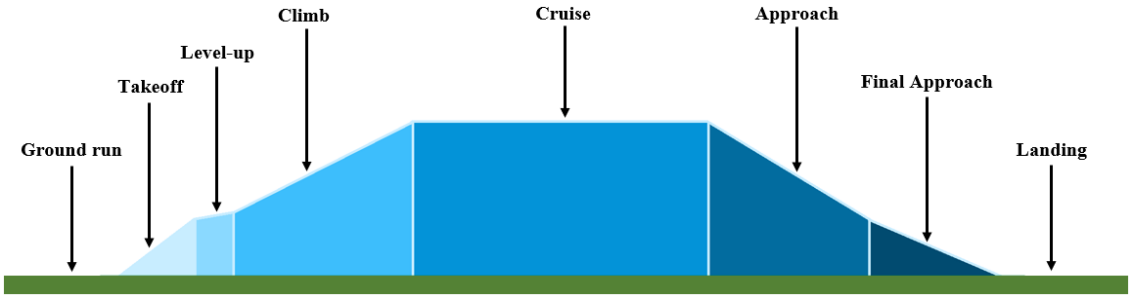

Fig. 7.5. The different flight phases followed and managed by the Flight Manager.

7.1. Experiments on Prototype 5

This section discusses the experiments conducted on the newly designed and altered ANNs that handle the different piloting tasks by controlling the appropriate control surfaces and other control interfaces as learned from the experienced human teacher. This section also discusses the experiments conducted on the newly designed and altered ANNs that are used during the final approach and landing phases to handle the interception of the runway centreline in extreme weather conditions. The experiments were conducted on the Elevators ANN to test the ability of maintaining the desired takeoff pitch angle, the new Elevators Trim ANNs to test the ability of maintaining different altitudes, climb rates, and the glideslope during approach and final approach, the new Throttle ANN to test the ability of maintaining different desired speeds, and the modified Flaps ANN to test the ability of extending the flaps correctly. The latter capabilities were not available in the previous prototypes of the IAS. Furthermore, additional experiments were conducted on the enhanced Ailerons, Rudder, and Roll ANN to handle runway centreline maintenance during the final approach and landing flight phases in extreme weather conditions beyond the capability of the previous prototypes of the IAS and the capabilities of modern autopilots and even human pilots, as well as the Glideslope Elevators Trim ANN to test its ability to maintain the desired 3 degrees glideslope in the same extreme weather conditions.

The first set of experiments that test the ability of mimicking the behaviour of experienced pilots of airliners were conducted under calm weather conditions, while the remaining set of experiments that test the ability to handle the landing runway centreline interception were conducted under extreme weather conditions,

To assess the effectiveness of the proposed approach, the Intelligent Autopilot System (IAS) was tested in eight experiments:

1. Takeoff Pitch Maintenance
2. Altitude Maintenance
3. Climb Rate Maintenance
4. Speed Maintenance
5. Flaps Setting
6. Final Approach Glideslope Maintenance

7. Runway Centreline Maintenance

7.1.1 Takeoff Pitch Maintenance

The purpose of this experiment is to assess the behaviour of the IAS when maintaining the 15 degrees pitch angle during the takeoff phase, and compare it to the demonstration provided by the human pilot. Since no standard modern autopilot is capable of performing autonomous takeoff, no comparison with the standard autopilot is provided.

7.1.1.1 Training

For this experiment, the Elevators ANN and the Pitch Rate of Change ANN were trained until a low Mean Squared Error (MSE) value was achieved (below 0. 01). Since single-hidden-layer ANNs are used for training, and since the training datasets are small given that a single demonstration was provided, training requires a short time to be completed (under 10 minutes).

7.1.1.2 Autonomous Control

For this experiment, the aircraft was reset to the runway in the flight simulator, and the IAS was engaged to test the ability of maintaining the standard takeoff pitch angle of 15 degrees. After the IAS completed the ground-run flight phase on the runway, the output of the Elevators ANN and the Pitch Rate of Change ANN were used to hold and maintain the desired pitch angle.

7.1.2 Altitude Maintenance

The purpose of this experiment is to assess the behaviour of the IAS compared with the standard autopilot of the model aircraft when maintaining a given altitude since the human pilot used the standard autopilot to handle this task.

7.1.2.1 Training

For this experiment, the Elevators Trim ANN and the Climb Rate ANN were trained until a low Mean Squared Error (MSE) value was achieved (below 0. 01). Since single-hidden-layer ANNs are used for training, and since the training datasets are small given that a single demonstration was provided, training requires a short time to be completed (under 10 minutes).

7.1.2.2 Autonomous Control

After training the ANNs, the aircraft was reset to the runway in the flight simulator, and the IAS was engaged to test the ability of maintaining different altitudes selected manually by the user. After the IAS took the aircraft airborne and reached the cruise flight phase, the output of the Altitude Rate of Change ANN and the Elevators Trim ANN were used to hold and maintain

three different altitudes at three different speeds, and maintain three different altitudes while speed is increasing from one speed to another and decreasing from one speed to another.

7.1.3 Climb Rate Maintenance

The purpose of this experiment is to assess the behaviour of the IAS compared with the standard autopilot of the model aircraft when maintaining a given climb or sink rate while changing altitude since the human pilot used the standard autopilot to handle this task.

7.1.3.1 Training

For this experiment, the same models generated after training the Elevators Trim ANN and the Climb Rate ANN in the previous experiments (A. Altitude Maintenance) were used without having to provide additional training.

7.1.3.2 Autonomous Control

For this experiment, the aircraft was reset to the runway in the flight simulator, and the IAS was engaged to test the ability of maintaining different climb rates selected manually by the user. After the IAS took the aircraft airborne and reached the cruise flight phase, the output of the Altitude Rate of Change ANN and the Elevators Trim ANN were used to hold and maintain six different climb or sink rates.

7.1.4 Speed Maintenance

The purpose of this experiment is to assess the behaviour of the IAS compared with the standard autopilot of the model aircraft when maintaining a given speed since the human pilot used the standard autopilot to handle this task.

7.1.4.1 Training

For this experiment, the Throttle ANN and the Speed Rate of Change ANN were trained until a low Mean Squared Error (MSE) value was achieved (below 0. 01). Since single-hidden-layer ANNs are used for training, and since the training datasets are small given that a single demonstration was provided, training requires a short time to be completed (under 10 minutes).

7.1.4.2 Autonomous Control

After training the ANNs, the aircraft was reset to the runway in the flight simulator, and the IAS was engaged to test the ability of maintaining different speeds selected manually by the user. After the IAS took the aircraft airborne and reached the cruise flight phase, the output of the Throttle ANN and the Speed Rate of Change ANN were used to hold and maintain three different speeds at three different altitudes.

7.1.5 Flaps Setting

The purpose of this experiment is to assess the behaviour of the IAS compared with the human pilot when extending and retracting the flaps given the altitude during the different flight phases.

7.1.5.1 Training

For this experiment, the Flaps ANN was trained until a low Mean Squared Error (MSE) value was achieved (below 0. 01). Since single-hidden-layer ANNs are used for training, and since the training datasets are small given that a single demonstration was provided, training requires a short time to be completed (under 10 minutes).

7.1.5.2 Autonomous Control

After training the ANN, the aircraft was reset to the runway in the flight simulator, and the IAS was engaged to test the ability of correctly deploying and retracting the flaps using different settings during the ground-run phase, takeoff, approach, and final approach. The output of the Flaps ANN was used to select the different flaps settings.

7.1.6 Final Approach Glideslope Maintenance

The purpose of this experiment is to assess the behaviour of the IAS compared with the standard autopilot of the model aircraft and the human pilot as well (during the last moments of final approach after disengaging the standard autopilot and taking full control) when maintaining the standard 3 degrees glideslope during the approach and the final approach flight phases in calm weather. In addition, this experiment assesses the behaviour of the IAS compared with the standard autopilot (Autoland) when maintaining the standard 3 degrees glideslope during the approach and the final approach flight phases in extreme weather conditions.

7.1.6.1 Training

For this experiment, the Glideslope Rate of Change ANN and the Glideslope Elevators Trim ANN were trained until a low Mean Squared Error (MSE) value was achieved (below 0. 01). Since single-hidden-layer ANNs are used for training, and since the training datasets are small given that a single demonstration was provided, training requires a short time to be completed (under 10 minutes).

7.1.6.2 Autonomous Control

After training the ANNs, the aircraft was reset to the runway in the flight simulator, and the IAS was engaged to test the ability of maintaining the standard 3 degrees glideslope during

approach and final approach in calm and extreme weather conditions. After the IAS took the aircraft airborne reached the approach flight phase, the output of the Glideslope Rate of Change ANN and the Glideslope Elevators Trim ANN were used to maintain the desired glideslope. The extreme weather conditions provided strong crosswind, gust, shear, and turbulence.

7.1.7 Runway Centerline Maintenance

The purpose of this experiment is to assess the behaviour of the IAS compared with the standard autopilot of the model aircraft and the human pilot as well (during the last moments of final approach after disengaging the standard autopilot and taking full control) when maintaining the centreline of the runway during the approach, final approach, and landing flight phases in calm weather. In addition, this experiment assesses the behaviour of the IAS compared with the standard autopilot (Autoland) when maintaining the centreline of the runway during the approach, final approach, and landing flight phases in extreme weather conditions.

7.1.7.1 Training

For this experiment, the Roll ANN and the Ailerons ANN were trained until a low Mean Squared Error (MSE) value was achieved (below 0. 01). Since single-hidden-layer ANNs are used for training, and since the training datasets are small given that a single demonstration was provided, training requires a short time to be completed (under 10 minutes).

7.1.7.2 Autonomous Control

After training the ANNs, the aircraft was reset to the runway in the flight simulator, and the IAS was engaged to test the ability of maintaining the centreline of the landing runway in calm and extreme weather conditions. After the IAS took the aircraft airborne and reached the approach flight phase, the output of the Roll ANN, the Ailerons ANN, and the Rudder ANNs were used to maintain the centreline of the landing runway. The extreme weather conditions provided strong wind including crosswind, gust, shear, and turbulence.

7.2 Results of Experiments on Prototype 5

The following section describes the results of the conducted tests.

7.2.1 Experiment 1 (Takeoff Pitch Maintenance)

Two models were generated for the Elevators ANN and the Pitch Rate of Change ANN with Mean Squared Error (MSE) values of 0.004 and 0.001 consecutively. Fig. 7.6 shows the pitch degree over time during ten different takeoffs where the IAS is controlling the elevators to maintain the standard fifteen degrees pitch angle (the lines in different shades of blue)

compared with the demonstration of the human pilot (the green line). Since the standard autopilot is not capable of performing takeoff autonomously, no comparison is provided. The results show that the means are equivalent according to the Two One-Sided Test (TOST) [150] when comparing the performance of the IAS and the experienced captain while maintaining the 15 degrees pitch during takeoff (see Table A.1 in Appendix A).

7.2.2 Experiment 2 (Altitude Maintenance)

Two models were generated for the Elevators Trim ANN and the Climb Rate ANN with MSE values of 0.01 and 0.0003 consecutively. Fig. 7.7, 7.8, and 7.9 illustrate a comparison between the IAS and the standard autopilot when maintaining three different altitudes over time. Since the human pilot used the standard autopilot to maintain the altitude, the comparison is done between the IAS and the standard autopilot. Fig. 7.10 illustrates a comparison between prototype 5 of the IAS and prototype 2 when holding an altitude. Prototype 2 used the throttle to maintain a given altitude, while prototype 5 uses the correct flight control surface (elevators trim) to maintain a given altitude. The results show that the means are equivalent according to the TOST when comparing the performance of the IAS and the standard autopilot while maintaining three different altitudes (see Tables A.2, A.3, and A.4 in Appendix A).

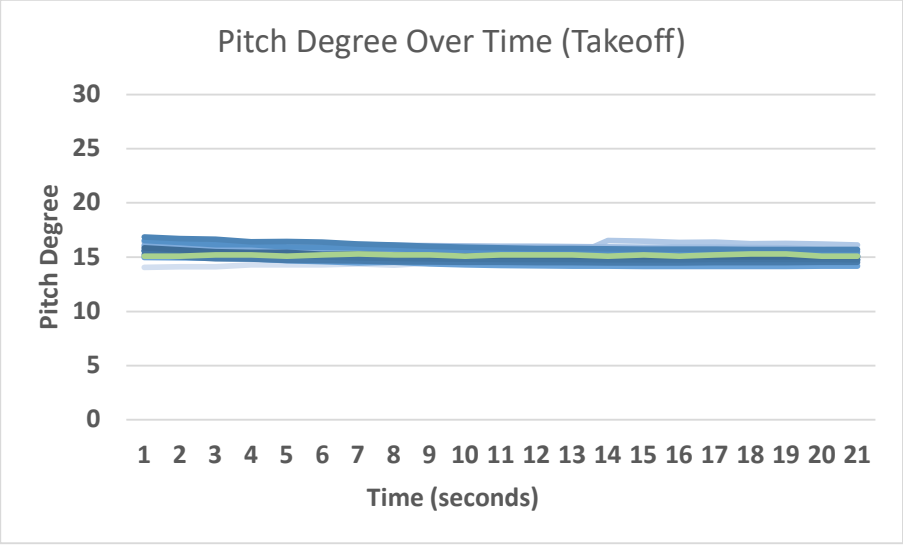

Fig. 7.6. The pitch degrees held by the IAS over time during fifteen different takeoffs (15 different attempts represented by the lines in different shades of blue) compared with the single demonstration of the human pilot (the green line) when maintaining a 15 degrees pitch.

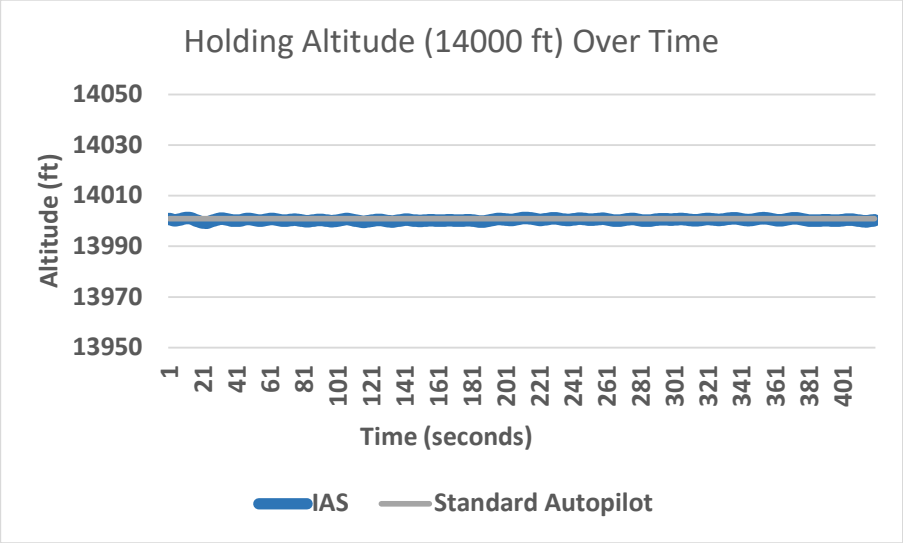

Fig. 7.7. A comparison between the IAS and the standard autopilot when maintaining an altitude of 14000 ft (speed is 250 knots).

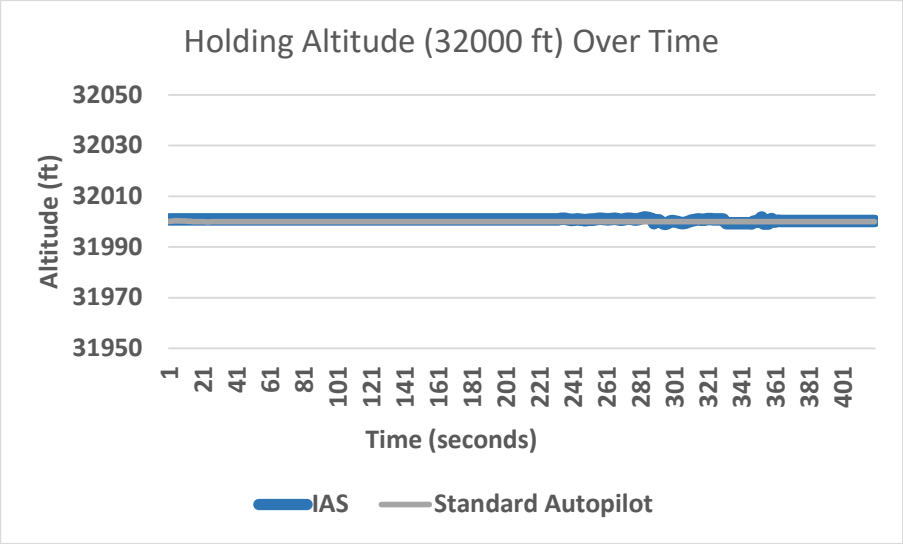

Fig. 7.8. A comparison between the IAS and the standard autopilot when maintaining an altitude of 32000 ft (speed is 340 knots).

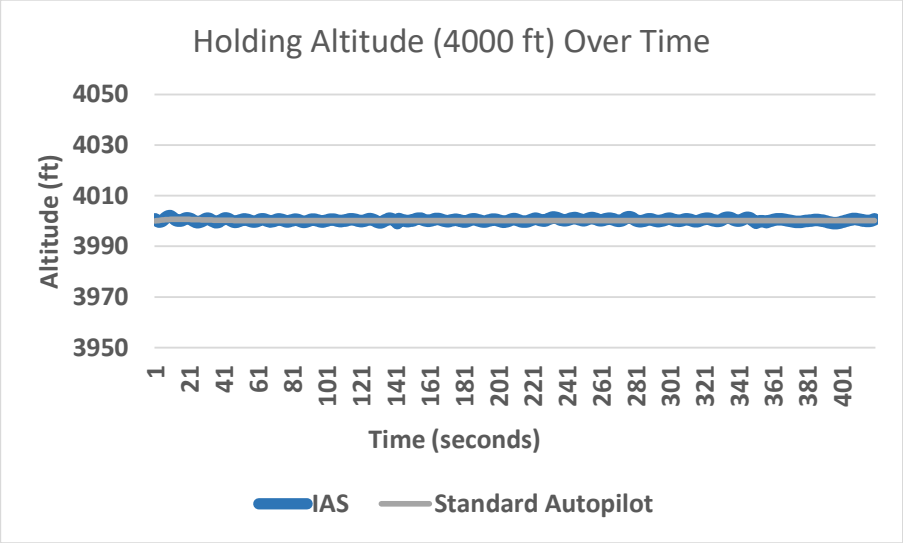

Fig. 7.9. A comparison between the IAS and the standard autopilot when maintaining an altitude of 4000 ft (speed is 220 knots).

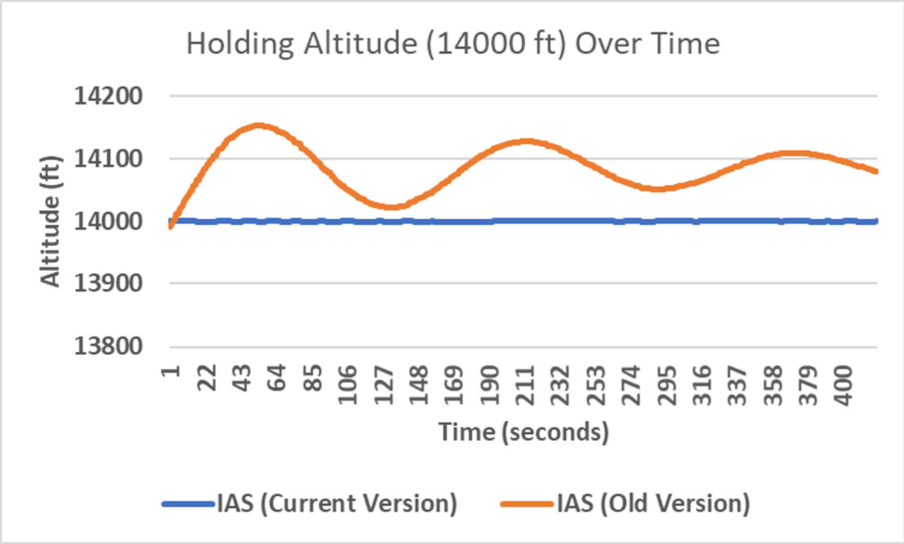

Fig. 7.10. A comparison between prototype 5 (blue line) and prototype 2 (orange line) of the IAS when maintaining an altitude of 14,000 ft. Prototype 2used the throttle, while prototype 5 uses the elevators trim to maintain a given altitude.

7.2.3 Experiment 3 (Climb Rate Maintenance)

The same models generated for altitude maintenance (Experiment 2 Altitude Maintenance) were used to maintain a given climb rate without having to retrain the models. Fig. 7.11, 7.12, 7.13, 7.14, 7.15, and 7.16 illustrate a comparison between the IAS and the standard autopilot when maintaining six different climb rates over time. Since the human pilot used the standard autopilot to maintain the climb rates, the comparison is done between the IAS and the standard autopilot. No comparison with the previous prototypes of the IAS is presented since the previous prototypes did not have the ability to maintain climb rates. The results show that the means are not equivalent (except for when maintaining a climb rate of -2000) according to the TOST when comparing the performance of the IAS and the standard autopilot while maintaining six different altitudes (see Tables A.5, A.6, A.7, A.8, A.9 and A.10 in Appendix A).

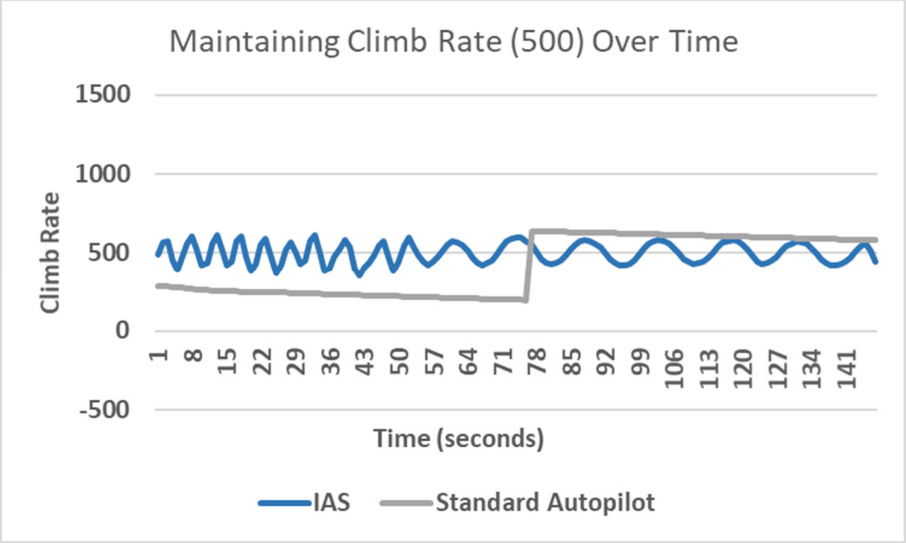

Fig. 7.11. A comparison between the IAS and the standard autopilot when maintaining a climb rate of 500 ft/min (speed is 250 knots).

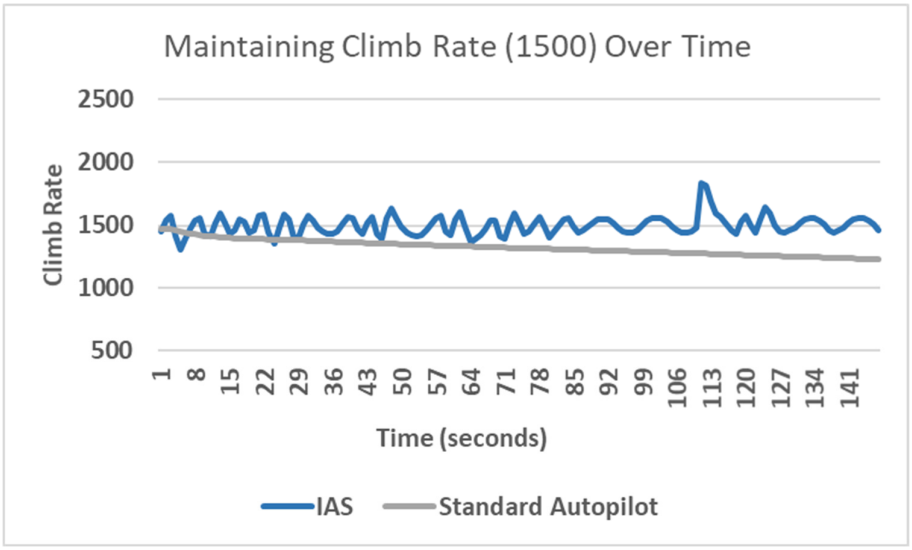

Fig. 7.12. A comparison between the IAS and the standard autopilot when maintaining a climb rate of 1500 ft/min (speed is 280 knots).

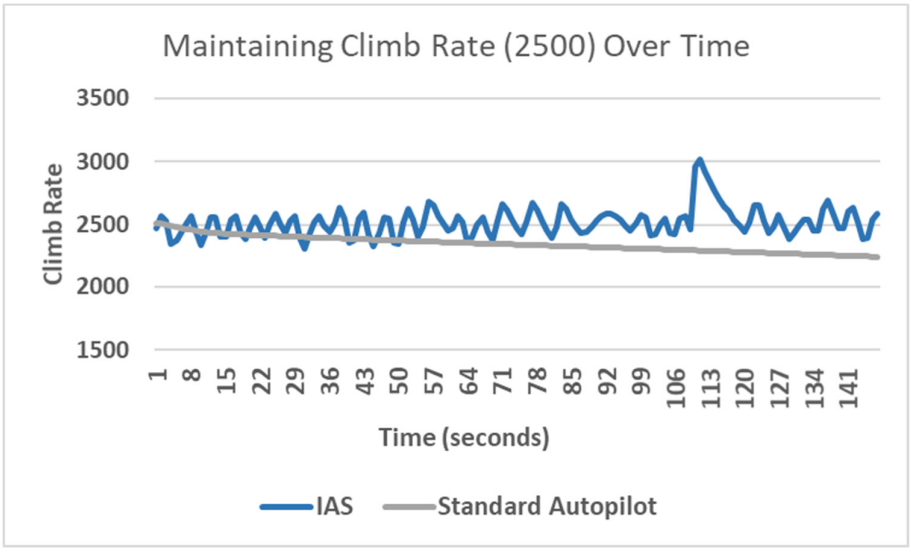

Fig. 7.13. A comparison between the IAS and the standard autopilot when maintaining a climb rate of 2500 ft/min (speed is 310 knots).

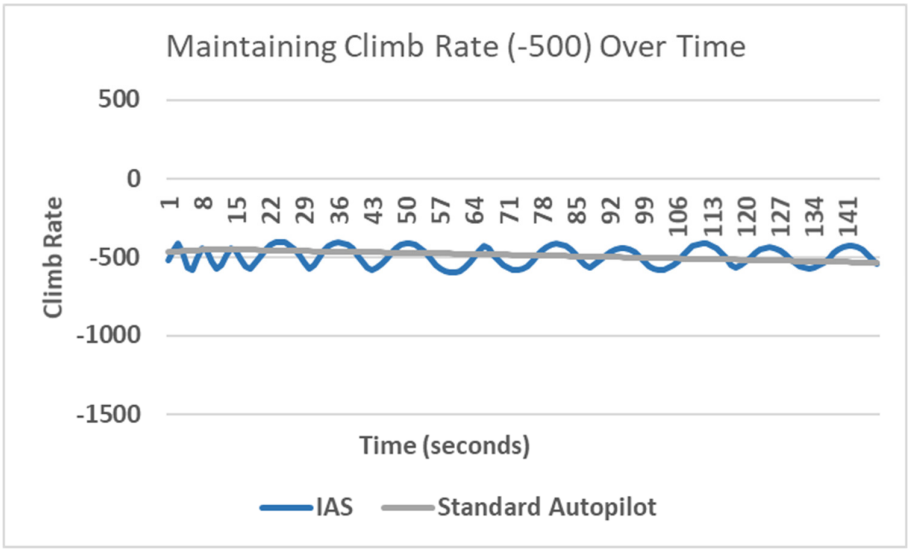

Fig. 7.14. A comparison between the IAS and the standard autopilot when maintaining a climb (sink) rate of -500 ft/min (speed is 230 knots).

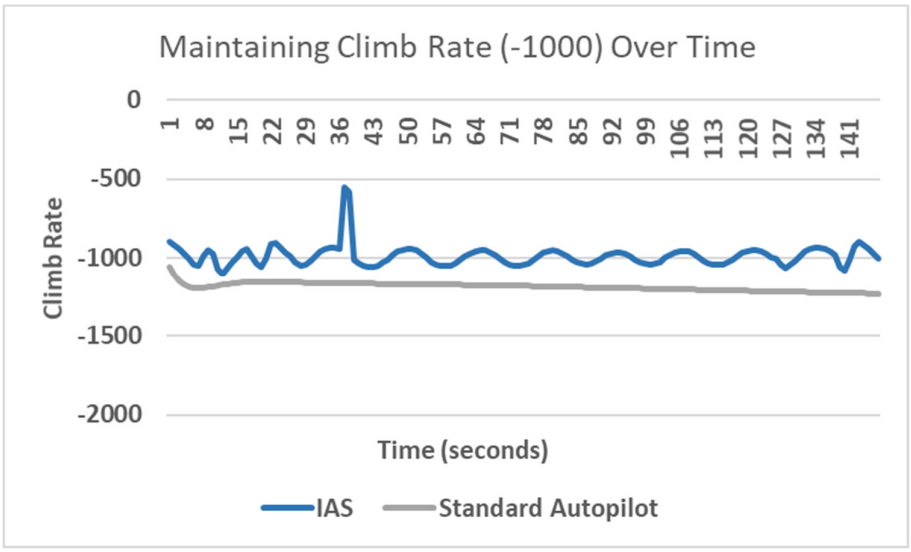

Fig. 7.15. A comparison between the IAS and the standard autopilot when maintaining a climb (sink) rate of -1000 ft/min (speed is 240 knots).

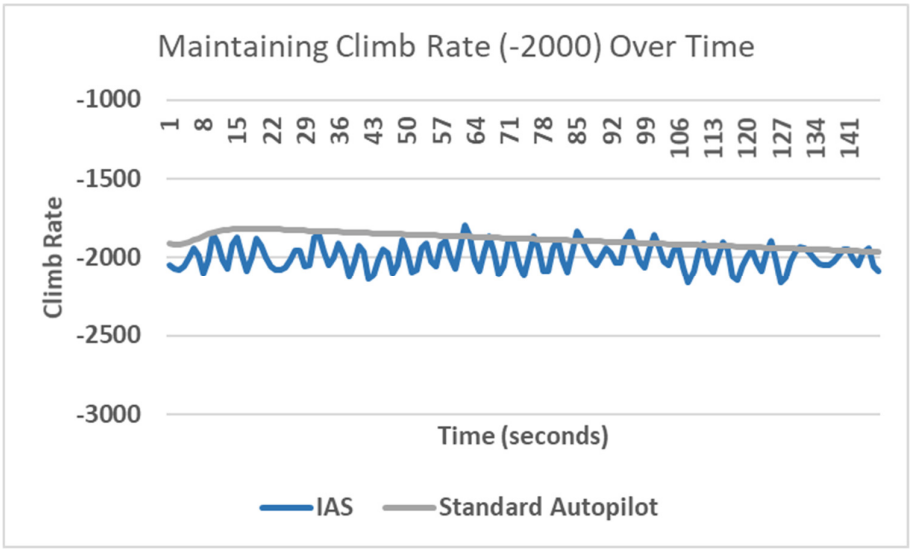

Fig. 7.16. A comparison between the IAS and the standard autopilot when maintaining a climb (sink) rate of -2000 ft/min (speed is 270 knots).

7.2.4 Experiment 4 (Speed Maintenance)

Two models were generated for the Throttle ANN and the Speed Rate of Change ANN with MSE values of 0.0009 and 0.0006 consecutively. Fig. 7.17, 7.18, and 7.19 illustrate a comparison between the IAS and the standard autopilot when maintaining three different speeds over time. Since the human pilot used the standard autopilot to maintain speed, the comparison is done between the IAS and the standard autopilot, however, Fig. 7.20 illustrates a comparison between the IAS and the human pilot when managing the different speeds throughout the complete flight from takeoff to landing. No comparison with the previous prototypes of the IAS is presented since the previous prototypes did not have the ability to maintain a given speed. The results show that the means are equivalent according to the TOST when comparing the performance of the IAS and the standard autopilot while maintaining three different speeds (see Tables A.11, A.12, and A.13 in Appendix A).

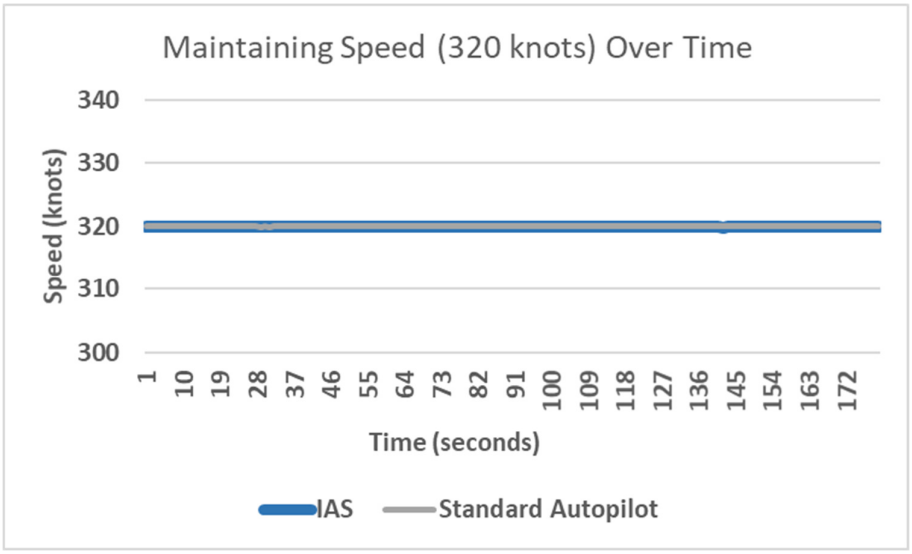

Fig. 7.17. A comparison between the IAS and the standard autopilot when maintaining a speed of 320 knots (altitude is 22000 ft.).

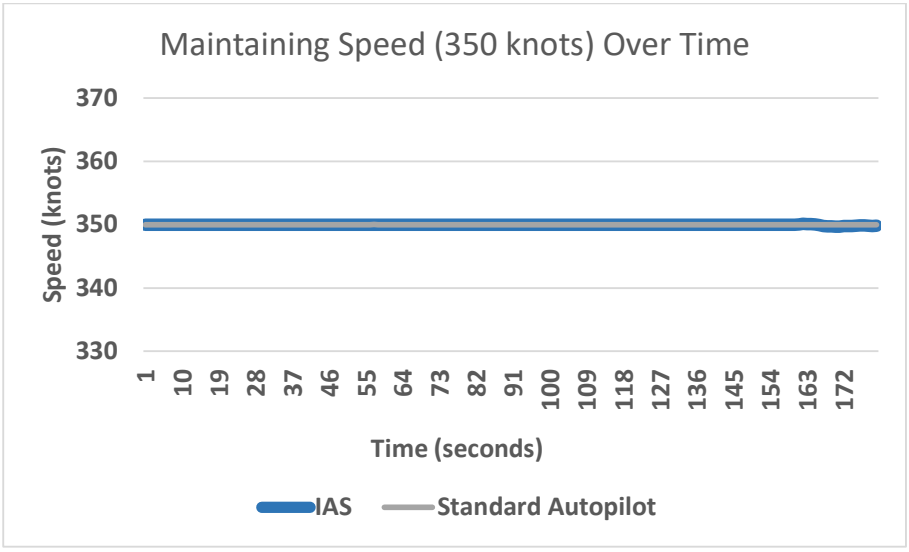

Fig. 7.18. A comparison between the IAS and the standard autopilot when maintaining a speed of 350 knots (altitude is 30000 ft.).

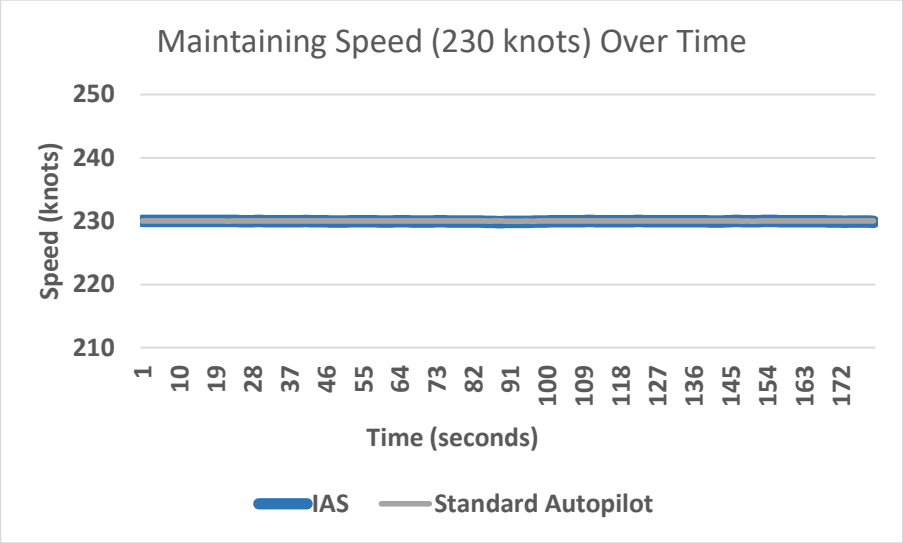

Fig. 7.19. A comparison between the IAS and the standard autopilot when maintaining a speed of 230 knots (altitude is 10000 ft.).

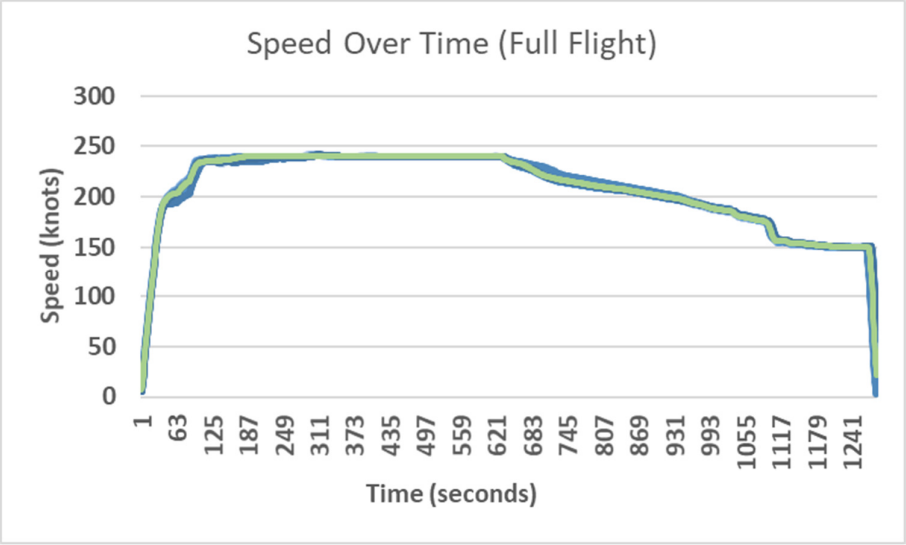

Fig. 7.20. A comparison between the IAS (10 flights represented by the overlapping lines in different blue shades) and the human pilot (1 demonstration represented by the green line) when managing the different speeds over time throughout the complete flight from takeoff to landing (London Heathrow to Birmingham). As can be seen, both the IAS and the human pilot accelerated sharply until the cruise speed of 240 knots was achieved, then, decelerated gradually until the landing speed of 150 knots was achieved before coming to a full stop on the landing runway.

7.2.5 Experiment 5 (Flaps Setting)

One model was generated for the Flaps ANN with an MSE value of 0.006. Fig. 7.21 and 7.22 show the flaps setting over altitude where Fig. 7.21 shows the flaps setting during the ground-run, takeoff, level-up, climb, and cruise flight phases, while Fig. 7.22 shows the flaps setting during the cruise, approach, final approach and landing flight phases. Since the standard autopilot is not capable of controlling the flaps autonomously, the provided comparison is between the IAS and the human pilot. Table 7.2 shows the corresponding flaps settings given the deflection value. Table 7.3 shows the mean, minimum, and maximum altitudes that correlate to each flaps setting in addition to the standard deviation.

TABLE 7.2
THE APPLIED FLAPS DEFLECTION VALUES AND THEIR CORRESPONDING FLAPS SETTINGS.

Flaps Deflection Value	**Flaps Setting**
0	Flaps Zero
0.166	Flaps One
0.332	Flaps Five
0.664	Flaps Twenty
1	Flaps Full

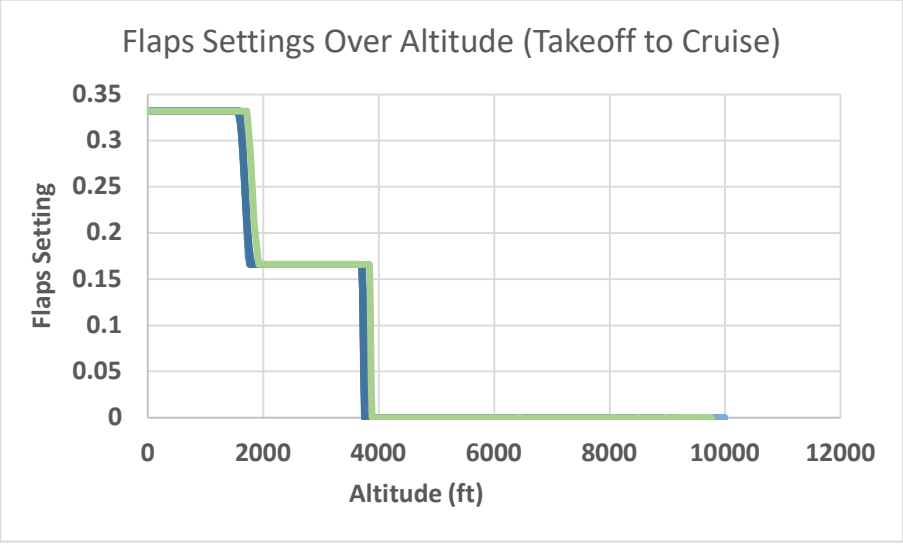

Fig. 7.21. A comparison between the IAS (10 flights represented by the overlapping lines in different blue shades) and the human pilot (1 demonstration represented by the green line) when managing the different flaps settings over altitude from takeoff to cruise.

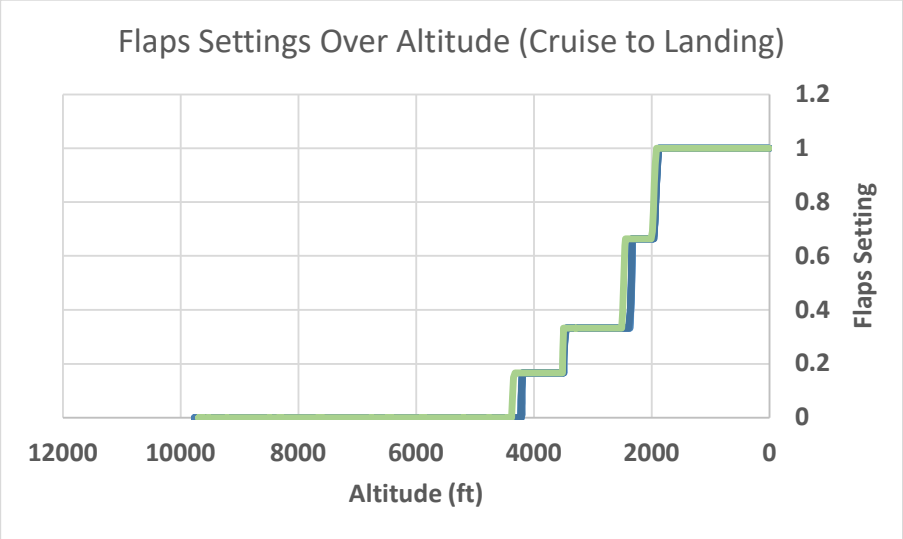

Fig. 7.22. A comparison between the IAS (10 flights represented by the overlapping lines in different blue shades) and the human pilot (1 demonstration represented by the green line) when managing the different flaps settings over altitude from cruise to landing.

TABLE 7.3
A COMPARISON BETWEEN THE HUMAN PILOT AND THE IAS WHEN MANAGING THE CORRELATION BETWEEN THE ALTITUDE (FT) AND FLAPS SETTING INCLUDING MEAN, MINIMUM, AND MAXIMUM ALTITUDES BY THE IAS THAT CORRELATE TO EACH FLAPS SETTING DURING THE DIFFERENT FLIGHT PHASES IN ADDITION TO THE STANDARD DEVIATION.

	Takeoff to Cruise		**Cruise to Landing**			
	Flaps 1	**Flaps 0**	**Flaps 1**	**Flaps 5**	**Flaps 20**	**Flaps Full**
Altitude (Human Pilot)	1800	3800	4150	3450	2330	1890
MIN Altitude (IAS)	1754	3753	4186	3440	2324	1871
MAX Altitude (IAS)	1821	3801	4198	3463	2334	1894
MEAN Altitude (IAS)	1792	3775	4192	3452	2329	1886
STD (IAS)	20	18	4	7	3	7

7.2.6 Experiment 6 (Final Approach Glideslope Maintenance)

Two models were generated for the Glideslope Rate of Change ANN and the Glideslope Elevators Trim ANN with MSE values of 0.0006 and 0.0008 consecutively. Fig. 7.23 illustrates a comparison between the IAS, the standard autopilot, and the human pilot (the final moments of final approach after the human pilot disengaged the autopilot and took full control of the aircraft) when attempting to maintain the standard 3 degrees glideslope during final approach in calm weather. Fig. 7.24 and 7.25 illustrate a comparison between the IAS and the standard autopilot (Autoland) when attempting to maintain the standard 3 degrees glideslope during final approach in extreme weather conditions with the presence of strong wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees (around 360 degrees), and turbulence. The results show that the means are equivalent according to the TOST when comparing the performance of the IAS, the experienced human pilot, and the standard autopilot while maintaining the 3 degrees glideslope during final approach (see Tables A.14 and A.15 in Appendix A).

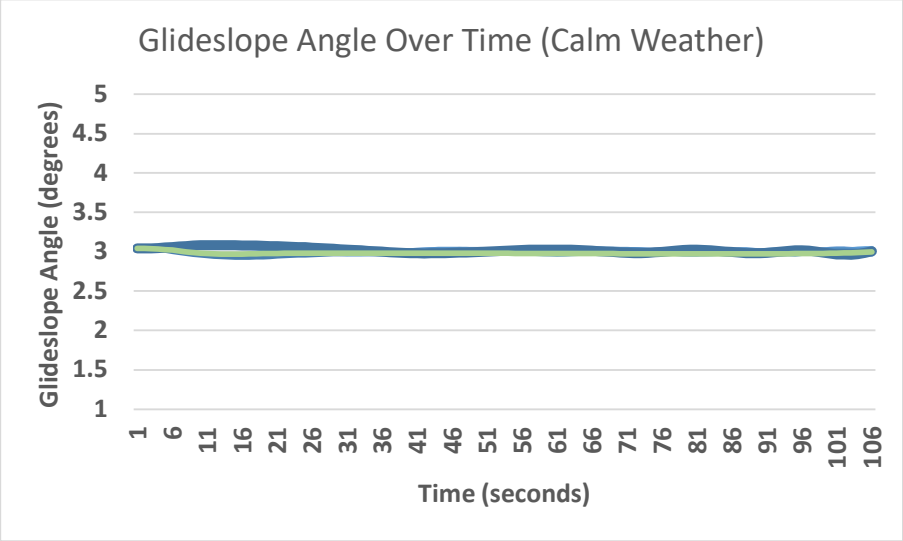

Fig. 7.23. A comparison between the IAS (10 flights represented by the overlapping lines in different blue shades), the standard autopilot, and the human pilot after he took full control of the aircraft during the last moments of final approach (1 demonstration represented by the green line) when maintaining the 3 degrees glideslope angle from final approach to landing in calm weather.

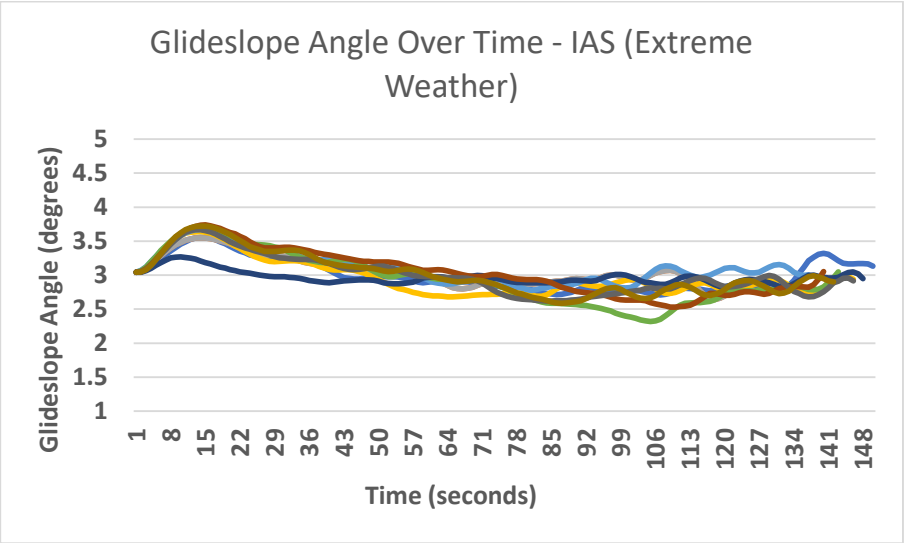

Fig. 7.24. 10 different attempts showing the glideslope angle of the aircraft (flown by the IAS) from final approach to landing. The goal is to try to maintain the standard 3 degrees glideslope. The weather conditions include 360 degrees wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and minor turbulence.

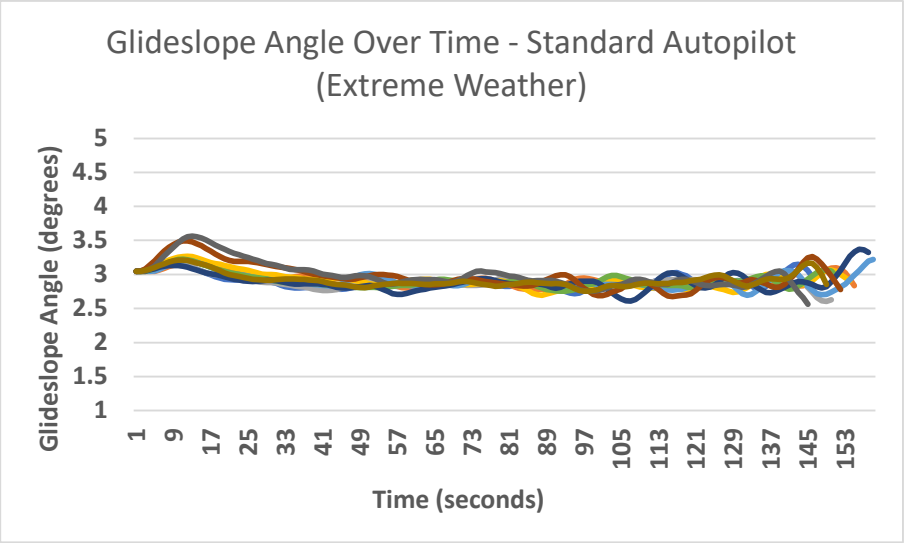

Fig. 7.25. 10 different attempts showing the glideslope angle of the aircraft (flown by the standard autopilot) from final approach to landing (10 different attempts). The goal is to try to maintain the standard 3 degrees glideslope. The weather conditions include 360 degrees wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and minor turbulence.

7.2.7 Experiment 7 (Runway Centerline Maintenance)

Four models were generated for the Roll ANN, the Ailerons ANN, the Heading ANN, and the Rudder ANN with MSE values of 0.0002, 0.001, 0.003, and 0.002 consecutively. Fig. 7.26 illustrates a comparison between the IAS, the standard autopilot of the aircraft model, and the human pilot (the final moments of final approach after the human pilot disengaged the autopilot and took full control of the aircraft) when attempting to maintain the centreline of the landing runway in calm weather. The results show that the means are equivalent according to the TOST when comparing the performance of the IAS, the experienced human pilot, and the standard autopilot while maintaining the 0 degrees angle which represents the centreline of the landing runway in calm weather (see Table A.16 in Appendix A). Fig. 7.27 shows the angle between the aircraft's location and the centreline of the landing runway before landing in extreme weather conditions with the presence of 90 degrees crosswind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and strong turbulence. In the latter weather conditions, the standard autopilot kept disengaging every time, therefore, the comparison is given between prototype 5 and prototype 4 of the IAS. Prototype 4 of the IAS was able to handle severe weather conditions with wind speed up to 50 knots, and a maximum wind shear of around 22 degrees. However, to perform a comparison between the IAS and the Autoland feature of the standard autopilot without facing the disengagement issue, the weather conditions were slightly modified by replacing the 90 degrees crosswind direction with 360 degrees, and lowering the intensity of turbulence. Fig. 7.28 and 7.29 illustrate a comparison between the IAS and the standard autopilot when attempting to intercept the centreline of the landing runway (airborne) in the slightly modified weather conditions. Table 7.4 shows the number of successful and unsuccessful attempts of prototype 4 and 5 of the IAS to keep the aircraft within the safe zone (angle between 0.05 and -0.05 degrees) during final approach while airborne.

Fig. 7.30 and 7.31 illustrate a comparison between the IAS and the standard autopilot when attempting to intercept the centreline of the landing runway after touchdown in extreme weather conditions with the presence of strong wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees (around 0 degrees), turbulence, and high precipitation (wet runway). Table 7.5 shows the number of successful and unsuccessful attempts of the IAS and the standard autopilot to keep the aircraft within the safe zone of the runway (angle between 0.05 and -0.05 degrees) after touchdown while attempting to decrease the speed to taxi speed.

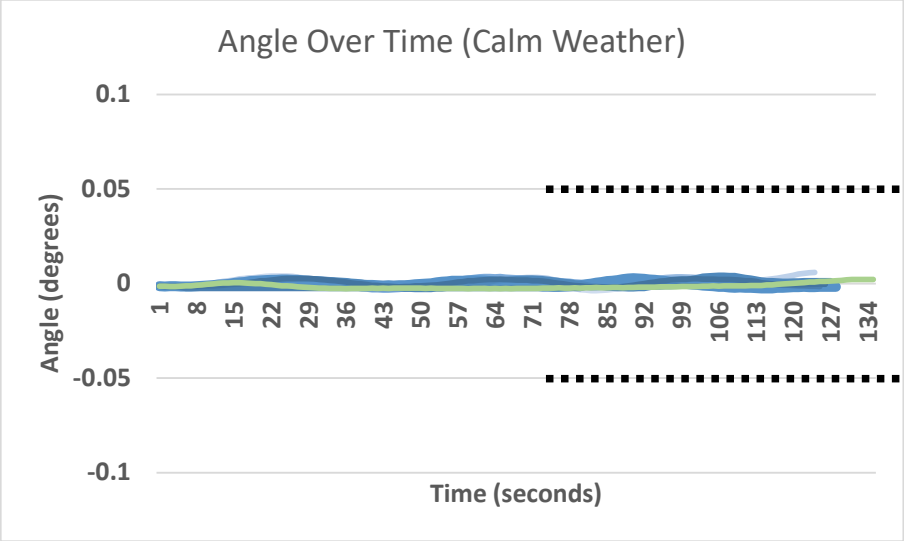

Fig. 7.26. A comparison between the IAS (10 flights represented by the overlapping lines in different blue shades), the standard autopilot, and the human pilot after he took full control of the aircraft during the last moments of final approach (1 demonstration represented by the green line) when maintaining the centreline of the landing runway (0 degrees) during final approach (airborne) and landing (on the ground after touchdown) in calm weather. The angle must be between 0.05 and -0.05 degrees especially during the last moments of the final approach to ensure landing within the safe touchdown zone of the landing runway as the two dashed black lines show (right part of the chart).

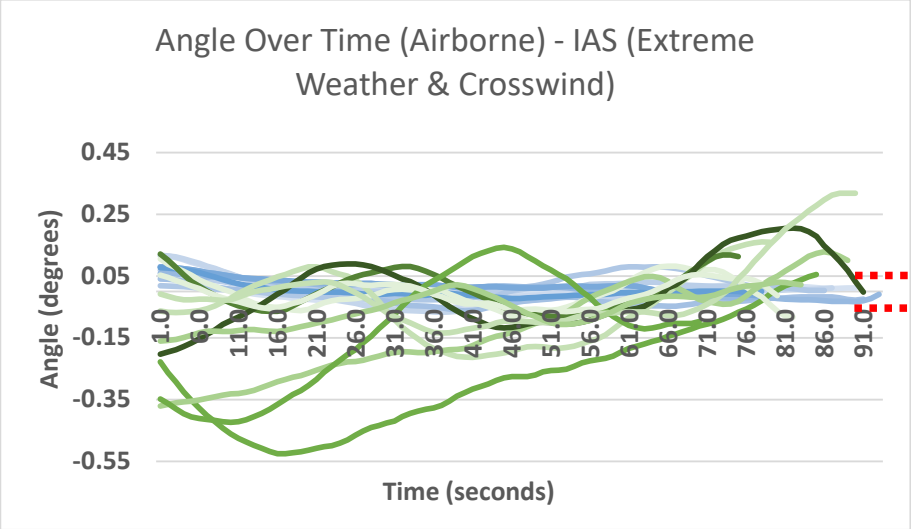

Fig. 7.27. A comparison between prototype 5 of the IAS (represented by the overlapping lines in different blue shades) and prototype 4 of the IAS (represented by the lines in different green shades) during 10 flights each when maintaining the centreline of the landing runway (0 degrees) during final approach (airborne). The angle must be between 0.05 and -0.05 degrees especially during the last moments of the final approach to ensure landing within the safe touchdown zone of the landing runway as the two dashed red lines show on the right side. The extreme weather conditions include 90 degrees crosswind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and strong turbulence.

TABLE 7.4

RESULTS OF COMPARING PROTOTYPE 5 AND 4 OF THE IAS WHEN ATTEMPTING TO MAINTAIN THE CENTRELINE OF THE RUNWAY DURING THE FINAL MOMENTS OF FINAL APPROACH (AIRBORNE) IN EXTREME WEATHER CONDITIONS INCLUDING 90 DEGREES CROSSWIND AT A SPEED OF 50 KNOTS WITH GUST UP TO 70 KNOTS, WIND SHEAR DIRECTION OF 70 DEGREES, AND STRONG TURBULENCE. SUCCESSFUL ATTEMPTS ARE WITHIN THE ANGLE THRESHOLD BETWEEN 0.05 AND -0.05 AND VICE VERSA.

	Runway centreline maintenance (airborne)
Pilot	**Successful**
The IAS (prototype 5)	10 out of 10
The IAS (prototype 4)	4 out of 10

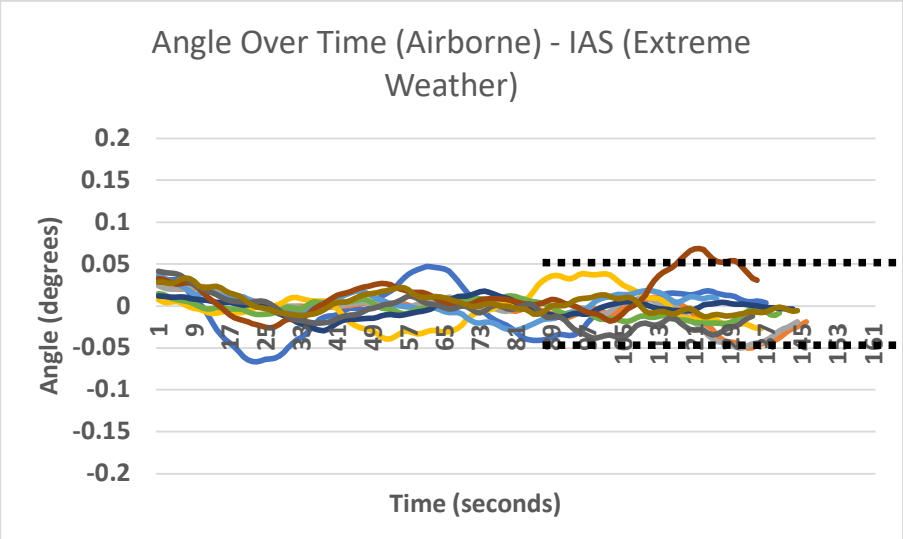

Fig. 7.28. The angle between the aircraft (flown by the IAS) and the centreline of the runway (0) during ten different final approach attempts (airborne). The angle must be between 0.05 and -0.05 degrees especially during the last moments of the final approach to ensure landing within the safe touchdown zone of the landing runway as the two dashed black lines show (right part of the chart). The weather conditions include 360 degrees wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and minor turbulence.

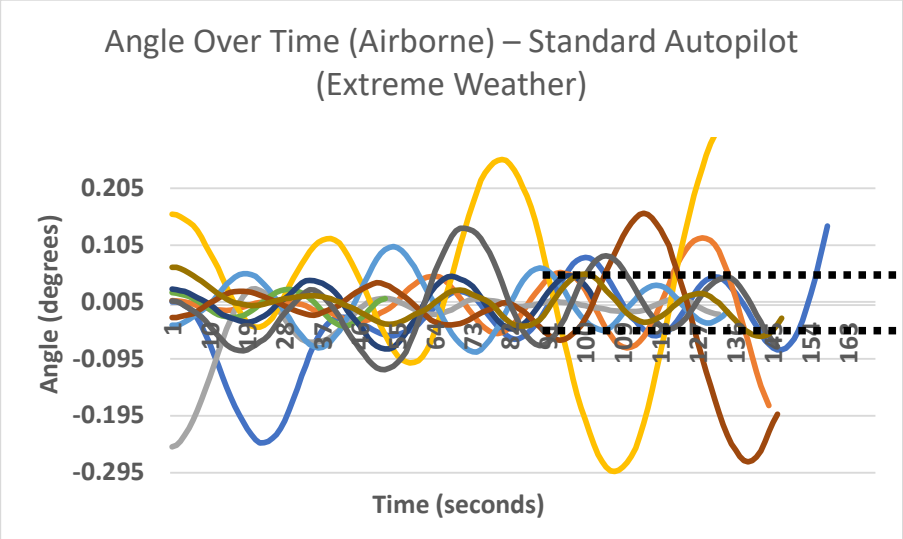

Fig. 7.29. The angle between the aircraft (flown by the standard autopilot) and the centreline of the runway (0) during ten different final approach attempts (airborne). The angle must be between 0.05 and -0.05 degrees especially during the last moments of the final approach to ensure landing within the safe touchdown zone of the landing runway as the two dashed black lines show (right part of the chart). The weather conditions include 360 degrees wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and minor turbulence.

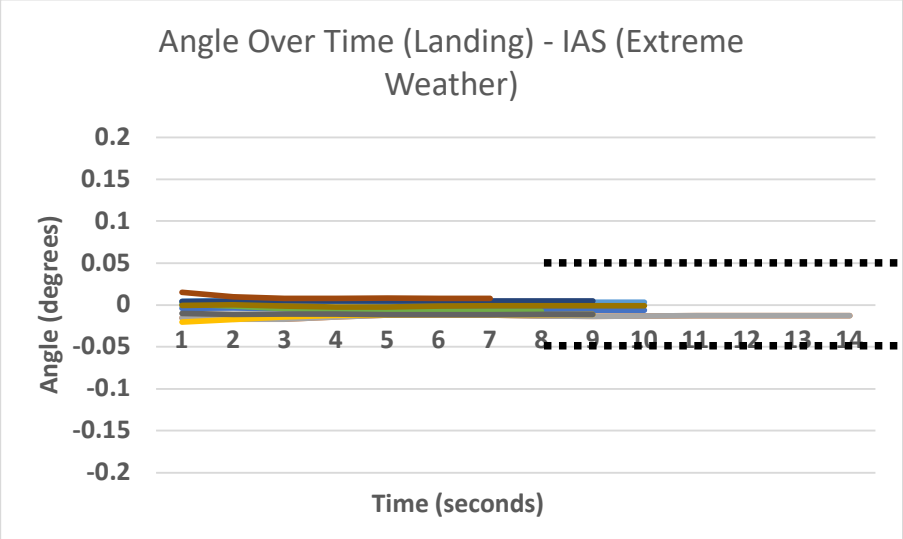

Fig. 7.30. The angle between the aircraft (flown by the IAS) and the centreline of the runway (0 degrees) during ten different landing attempts. The angle must be between 0.05 and -0.05 degrees during touchdown to ensure landing within the safe touchdown zone of the landing runway as the two dashed black lines show (right part of the chart), and during the attempt to decrease the aircraft's speed on the runway to taxi speed. The weather conditions include 360 degrees wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and minor turbulence.

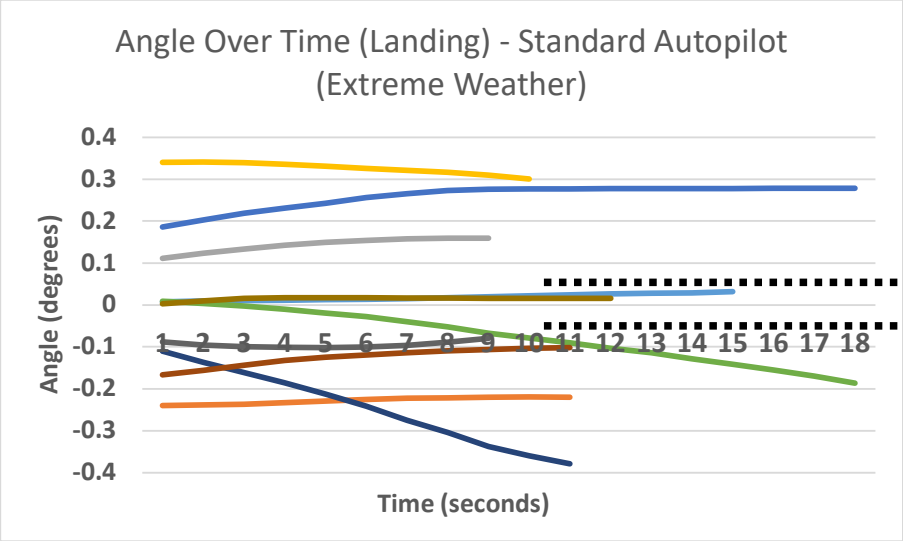

Fig. 7.31. The angle between the aircraft (flown by the standard autopilot) and the centreline of the runway (0 degrees) during ten different landing attempts. The angle must be between 0.05 and -0.05 degrees during touchdown to ensure landing within the safe touchdown zone of the landing runway as the two dashed black lines show (right part of the chart), and during the attempt to decrease the aircraft's speed to taxi speed. The weather conditions include 360 degrees wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and minor turbulence.

TABLE 7.5
RESULTS OF COMPARING THE IAS WITH THE STANDARD AUTOPILOT WHEN ATTEMPTING TO MAINTAIN THE CENTRELINE OF THE RUNWAY DURING THE FINAL MOMENTS OF FINAL APPROACH (AIRBORNE) AND AFTER TOUCHDOWN WHILE DECREASING THE SPEED OF THE AIRCRAFT TO TAXI SPEED ON THE LANDING RUNWAY IN EXTREME WEATHER CONDITIONS. SUCCESSFUL ATTEMPTS ARE WITHIN THE ANGLE THRESHOLD BETWEEN 0.05 AND -0.05 AND VICE VERSA.

	Runway centreline maintenance (airborne)	**Runway centreline maintenance (ground)**
Pilot	**Successful**	**Successful**
The IAS	20 out of 20	10 out of 10
Standard Autopilot	5 out of 20	2 out of 10

7.3 Analysis

As can be seen in Fig. 7.5 (A. Takeoff Pitch Maintenance), the IAS was able to maintain the standard pitch angle of 15 degrees during the takeoff phase. Table A.1 (Appendix A) shows that the IAS was able to maintain a pitch angle mean of 15.24 degrees which is equivalent to the 15.17 degrees mean maintained by the human pilot as the equivalence test shows.

Fig. 7.7, 7.8, and 7.9 (B. Altitude Maintenance) show that the IAS was able to maintain three different altitudes at three different speeds as did the standard autopilot. Tables B.2, B.3, and A.4 (Appendix A) show that the performance of the IAS when maintaining a given altitude at a given speed is equivalent to the performance of the standard autopilot. Fig. 7.10 shows the improvement in the ability of maintaining a given altitude by comparing prototype 2 of the IAS which was not able to accurately maintain altitudes with prototype 5 which now have the ability to handle this task precisely. However, Fig. 7.7, 7.8 and 7.9 show that the performance of the IAS showed oscillations.

Fig. 7.11, 7.12, 7.13, 7.14, 7.15, and 7.16 (C. Climb Rate Maintenance) show that although the performance of the IAS showed oscillations when maintaining climb rates, it performed better than the standard autopilot when maintaining six different climb/sink rates at six different speeds. The TOST results confirm this, which shows the closer means to the desired climb/sink rate values compared with the standard autopilot as Tables A.5, A.6, A.7, A.8, A.9, and A.10 (Appendix A) show. Although the performance of the standard autopilot is steadier (no oscillations), for an unknown reason, when the standard autopilot attempts to reach the selected climb/sink rate, it will continue to drift away from it slowly in most cases as Figs. 7.12, 7.13, 7.14, and 7.15 show.

It is clear that the oscillations are recurrent in both experiments (B. Altitude Maintenance and C. Climb Rate Maintenance), and they can only be seen when controlling the elevators trim control surfaces using the Elevators Trim ANN. This ANN was trained to handle two different tasks which are maintaining climb/sink rates and maintaining altitudes. The reason for having just one ANN for both tasks is because these tasks are handled using the same control surfaces of the aircraft (the elevators trim). However, it is clear that following the general approach in this work of designing and training dedicated ANNs that handle specific tasks showed excellent results in all the previous chapters and most of the work in this chapter, therefore, based on the results of using dedicated ANNs for specific tasks, it is likely that segregating the tasks could improve or eliminate the oscillations, which can be explored in future work.

Fig. 7.17, 7.18, and 7.19 (D. Speed Maintenance) illustrate the equivalent performances of the IAS and the standard autopilot when maintaining three different speeds at three different altitudes. Tables A.11, A.12, and A.13 (Appendix A) confirm the equivalence between the performances of the IAS and the standard autopilot when handling this task. Fig. 7.20 shows

that the IAS was able to manage and maintain the different speeds in the different flight phases from takeoff to landing in a manner that is identical to the human pilot throughout the same flight.

Fig. 7.21 and 7.22 (E. Flaps Setting) illustrate the consistent behaviour of the IAS when extending and retracting the flaps given the flight phase and altitude, which is similar to the behaviour of the human pilot when handling this task. The minor differences shown in table 7.3 are due to the terrain variation below the aircraft since the applied altitude here is feet above ground level instead of sea level.

Fig. 7.23 (F. Final Approach Glideslope Maintenance) shows the similar performance of the IAS, the standard autopilot, and the human pilot when maintaining the standard 3 degrees glideslope angle during final approach and landing in calm weather. Table A.14 (Appendix A) confirms the equivalence between the performance of the IAS, the standard autopilot, and the human pilot when handling this task. Fig. 7.24 and 7.25 show the similar performance of the IAS and the standard autopilot (Autoland) while maintaining the standard 3 degrees glideslope angle in extreme weather conditions including 360 degrees wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and minor turbulence. Table A.15 (Appendix A) shows that the means of the glideslope angle maintained by the IAS and the standard autopilot are equivalent, however, the IAS performed better since the glideslope mean is 3.01 which is closer to the desired 3 degrees glideslope compared with the 2.93 mean achieved by the standard autopilot. The oscillations are not present in this experiment although the same control surfaces which generated them in the previous experiments (B. Altitude Maintenance and C. Climb Rate Maintenance) are used here as well, however, a dedicated ANN (Glideslope Elevators Trim ANN) is used here to control the elevators trim to maintain the desired glideslope degree. This is an indicator that supports the suggestion above which shows the better approach of designing and training dedicated ANNs for each specific task.

As can be seen in Fig. 7.26 (G. Runway Centreline Maintenance), the IAS was able to maintain the centreline of the landing runway as did the human pilot and the standard autopilot in calm weather. Table A.16 (Appendix A) confirms the equivalence between the performance of the IAS, the human pilot, and the standard autopilot when handling this task. Although the angle went briefly beyond the safe touchdown zone (between 0.05 and -0.05 degrees from the centreline of the runway) as Fig. 7.27 and 7.28 show, this happened due to a sudden strong gust while airborne which is normal in such conditions. It is obviously important however to

touchdown and land within the safe touchdown zone, which was achieved every time as Fig. 7.27 and 7.28 show, while the standard autopilot kept disengaging every time in the latter weather conditions. The extreme weather conditions included 90 degrees crosswind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and strong turbulence. Compared with prototype 4 of the IAS which achieved a success rate of 40% (4 successful attempts out of 10 trials), prototype 5 achieved a success rate of 100% (10 successful attempts out of 10 trials) as table 7.4 and Fig. 7.27 show when intercepting the centreline of the landing runway in such extreme weather conditions, which proves the effectiveness of the method introduced in this chapter, which focuses on altering the roll degree instead of the bearing degree. This method performed well because as the external force represented by the extreme wind increases, the IAS will keep increasing the roll degree to counter the effect. After altering the weather conditions by replacing the 90 degrees crosswind with 360 degrees wind and lowering the intensity of turbulence, the standard autopilot was able to land, however, as Fig. 7.29 shows, the standard autopilot struggled to keep the aircraft within the safe zone (between 0.05 and -0.05 degrees from the centreline of the runway). Table 7.5 shows that the IAS was able to achieve a success rate of 100% (20 successful attempts out of 20 trials), while the standard autopilot achieved a success rate of 25% (5 successful attempts out of 20 trials) which confirms that the IAS can perform beyond the capabilities of modern standard autopilots in extreme weather conditions. In addition, Fig. 7.30 illustrates the performance of the IAS while trying to keep the aircraft within the safe zone of the landing runway (between 0.05 and -0.05 degrees from the centreline of the runway) after touchdown while decreasing the speed of the aircraft on the runway to taxi speed, while Fig. 7.31 illustrates the poor performance of the standard autopilot when attempting to handle the same task in the same extreme weather conditions, which is an additional proof of the superiority of Artificial Neural Networks when handling difficult control problems in extreme conditions compared with the PID controllers used in modern autopilots. In addition, table 7.5 shows that the IAS was able to achieve a success rate of 100% (10 successful attempts out of 10 trials), while the standard autopilot achieved a success rate of 20% (2 successful attempts out of 10 trials) while maintaining the centreline of the landing runway after landing, which further confirms the superior performance of the IAS which is beyond the capabilities of standard autopilots.

Overall, the distinct performance of the IAS presented natural and dynamic behaviour when handling the different tasks by manipulating the different control surfaces especially in extreme weather conditions. This proves its superiority compared with the mechanical-precision-like

performance of the standard autopilot, which according to the literature, suffers from robustness issues when facing uncertainty. The latter hinders the reaction time of modern autopilots, and the ability to cope with such extreme and sometimes sudden conditions.

7.4 Evaluating the IAS by Oman Air

To involve the aviation industry in evaluating the performance of the IAS, and in addition to providing training data for the IAS, Captain Khalid Al Hashmi, Oman Air, provided his feedback after being presented with complete (airport to airport) flight demonstrations of the IAS, and landings in calm and extreme weather conditions as the experiments above show. We asked him the following questions, and he answered as follows:

1. Compared with the standard modern autopilot, what is your impression on the performance of the IAS when executing complete flights in calm and severe weather conditions? "*Good. I Wish we can try the IAS in a 6-axis full motion flight simulator to evaluate it further.*"
2. Although flying in such conditions is probably against regulations, but for testing purposes, is the IAS capable of preforming crosswind landings beyond the current limits and capabilities of modern autopilots? What about experienced human pilots? "*Yes. It is always a challenge, human pilots are allowed to land in crosswind conditions up to the demonstrated limit such as 38 knots, whereas the autopilots limit is less. I Hope that the IAS can help in increasing the crosswind limit which is sometimes limited due to flight controllability rather than pure capability.*"
3. Is the current performance of the IAS in general comparable with human pilots? If yes, as an experienced captain and instructor, how would you rate its performance if it were human? novice, intermediate, or experienced? "*Yes, I would say intermediate although I suggest comparing it more with other autopilot.*"
4. Do you agree that the IAS has the potential to introduce new advantages to the aviation industry such as enhancing safety as a dependable autopilot compared with the modern ones? "*Yes, it does. It just needs to be trained more on scenarios and various conditions and malfunctions.*"

7.5 Summary

To summarize, the objective of teaching Artificial Neural Networks how to pilot an aircraft like experienced human pilots of airliners was achieved by learning from a demonstration provided by an experienced human pilot. The latter learning enabled the Intelligent Autopilot

System (IAS) to accurately mimic the behaviour of experienced human pilots by manipulating the appropriate control surfaces and other control interfaces to perform the different piloting tasks including maintaining the desired pitch during takeoff, the desired climb or sink rates, the desired speeds, and the desired glideslope during final approach.

In addition, introducing the new Roll ANN which replaced the Bearing Adjustment ANN (from prototype 4), altering the Ailerons ANN (from prototype 3), introducing the new Heading ANN, and altering the Rudder ANN equipped the IAS with the ability to handle landings in extreme weather conditions beyond the current limits and capabilities of modern autopilots and experienced human pilots as well.

This provides additional evidence to support the hypothesis of this work aimed towards proving the possibility to teach a flight control system piloting skills.

8. CONCLUSION

In this work, a novel and robust approach is proposed to "teach" autopilots how to perform complete flights starting with takeoff, navigating, and ending with safe landing while being able to handle uncertainties and emergencies such as extreme weather conditions, engine(s) failure and fire, rejected takeoff, emergency landing, and go-around, with minimum effort by exploiting Learning from Demonstration, and depending on Artificial Neural Networks as the sole controllers for this difficult control approximation problem.

The results show that the hypothesis of the possibility to use Learning from Demonstration with Artificial Neural Networks to transfer the skills and abilities of experienced human pilots to a flight control system, to give it the capability to perform autonomous and complete flights in normal and uncertain conditions as well, has been proved.

The successful achievement of the five prototypes and their objectives (chapters 3, 4, 5, 6, and 7) provide evidence to support the hypothesis. The successfully achieved objectives are:

1. Proving the ability of Artificial Neural Networks and the Learning from Demonstration concept to learn piloting tasks by generating learning models from training datasets containing demonstrations performed by a human teacher in a flight simulator. The learning models capture low-level and high-level skills and abilities that enabled the IAS to perform basic flights under calm and severe weather conditions.
2. Teaching ANNs complex tasks representing the ability to handle multiple emergency situations including Rejected Takeoff (RTO), engine failure and fire, and emergency landing, in addition to maintaining a desired altitude.
3. Teaching ANNs complex flying including the ability to takeoff from airport A, navigate to airport B, and land safely.
4. Handling severe weather landing, and go-around (aborting landing, then, reattempting).
5. Proving the ability of Artificial Neural Networks and the Learning from Demonstration concept to not only learn how to fly an aircraft, but to learn how to fly and execute the necessary piloting tasks like experienced human pilots of airliners, and to handle landings in extreme weather conditions that are beyond the current limits and abilities.

The results confirm that Supervised Learning is a straightforward and robust approach in the context of Learning from Demonstration since performing offline training on labelled datasets has been efficient in this work. Relying completely on Artificial Neural Networks to

capture low-level and highly dynamic piloting tasks such as the rapid corrections of heading and roll deviations in stormy weather conditions, confirms that ANNs are suitable for handling such highly dynamic data. In addition, the ANNs in this work achieved the ability to capture high-level tasks that are represented by sequences of actions aimed towards achieving a strategic goal such as managing the different flight phases. This confirms the suitability of ANNs for learning various piloting tasks from demonstrations, as well as from synthetic training data.

Enhancing the performance of the ANNs through pre-training techniques such as scaling the values to achieve a closer magnitude of the inputs and outputs resulted in better models. In addition, the technique of altering how the ANNs are stimulated proved to be an efficient and robust technique which can be applied to change the behaviour of the ANNs without having to generate additional models that can achieve the desired behaviour.

Breaking down the piloting tasks, and adding more Artificial Neural Networks allowed overcoming the black-box problem by having multiple small ANNs with single-hidden-layers that learn from small labelled datasets which have clear patterns. Using Supervised Learning on these small and multiple ANNs provides the possibility to trace the complete learning and operation processes. The black-box problem associated with some Artificial Intelligence methods such as Deep Learning, has been the main obstacle of introducing AI to the cockpit. In addition, this approach enhanced performance and accuracy, and allowed the coverage of a wider spectrum of tasks.

The novelties presented in this work which are dedicated to introducing intelligent autonomy to large jets such as airliners, are robust solutions that could enhance flight safety in the civil aviation domain by enabling autonomous behaviour that was not possible before.

However, to achieve comprehensive cockpit autonomy capabilities that can be fully relied on, the IAS must be trained to handle any scenario that human pilots are trained to handle, in other words, the IAS must undergo the same complete and comprehensive training that human pilots undergo.

The aviation industry is currently working on solutions which should lead to decreasing the dependence on crew members. The reason behind this is to increase safety by lowering workload, human error, and stress faced by crew members, and to lower costs associated with having more than one pilot in the cockpit, by developing autopilots capable of handling

multiple scenarios without human intervention. It is anticipated that future Autopilot systems which make of methods proposed here could improve safety, save lives, and lower costs.

9. FUTURE WORK

This work has already caught the attention of the aerospace industry in Europe and North America. Preliminary discussions are already being held to identify the roadmap of adopting the Intelligent Autopilot System (IAS), and Non-Disclosure Agreements have been signed with three industry leaders in the aerospace domain. The first expected step is to expose the IAS to a thorough V&V (Verification and Validation) process that is aimed towards confirming the absence of a black-box, and confirming the clarity and traceability of the different components including the generated models (being based on clear patterns that can even be visualised) since the proposed methods in this work allow the possibility to trace, investigate, and analyse the complete architecture of the IAS. After that, it is expected to integrate the IAS with a 6-axis certified flight simulator, and teach the IAS by allowing it to observe comprehensive demonstrations by experienced pilots representing the different scenarios that all human pilots go through during their training. Furthermore, additional comparisons with different autopilot should be conducted as suggested by Captain Al Hashmi.

In addition, future work should include enhancing the current capabilities of the IAS where needed especially the oscillations issue seen when maintaining climb/sink rates and altitudes. As explained in the analysis section of chapter 7, these two tasks should be segregated, and two dedicated ANNs should be designed and trained to handle each tasks separately to enhance accuracy and achieve smooth control command outputs.

There are other challenges that must be investigated and tackled before the IAS or any other comprehensive cockpit autonomy solution can be fully trusted to replace human pilots. As mentioned above, the IAS must first be trained on all the scenarios, and acquire all the skills and abilities that human pilots acquire during their training. Then, the system must be able to detect and handle faulty sensors that either provide false data or no data at all, which is a vast field of research that investigates this problem to identify potential solutions. Relying on other functional sensors to substitute is a plausible solution such as using GPS to calculate the speed of the aircraft if the speed sensors are faulty. The latter example is quite straightforward, however, it becomes much more challenging when other sensors that cannot be easily substituted become faulty. For instance, what if the roll, pitch, or Angle of Attack (AoA) sensors become faulty such as the case of the Boeing 737 MAX? In this case, a plausible solution could again be relying on other functional sensors but not as straightforward as using GPS to calculate speed. If we take the Boeing's AoA issue as an example, additional Artificial

Neural Networks (ANN) could be added to the IAS for the purpose of continuously monitoring the overall condition of the aircraft including the relationship between the sensor reading and the current setting. For example, an ANN could be designed and trained to detect that although the throttle setting is at 60% (engines power at 60%) during a flight phase that has a relatively low pitch (cruise for example), it is not proportional to the rate of change of speed, meaning, since the engines power is at such high setting, however, the speed is not increasing as it should be or even decreasing, then, this could mean that the aircraft's AoA is too high which makes it difficult for the engines to compensate and provide the necessary increase of speed, and vice versa. In this case, the IAS should gradually push the yoke (or stick) or lower the elevators trim to push the aircraft's nose down until the ANN detects the expected proportion between throttle setting, flight phase, and the rate of change of speed.

Furthermore, the problem of control surfaces faults is another field of research that has been focusing on tackling this issue for future autonomous flying by developing intelligent fault-tolerant control systems as mentioned in the literature review above. For this, the IAS can instead be trained on how to handle faults of control surfaces just like how human pilots are trained to do so. For example, human pilots are trained to handle a faulty aileron that hinders or eliminates the ability of turning or banking right or left by applying more thrust to one of the engines compared with the other to help turning. The IAS can be trained to execute the latter technique by following the same method presented in this work of providing minimal demonstrations of how to perform this particular task or any other by an experienced pilot.

In addition, to handle navigation issues that arise due to GPS denial for example since the IAS fully relies on GPS for navigation and landing as well, additional methods should be incorporated such as Inertial Guidance Systems and even computer-vision especially during the final approach flight phase where the safety risk is higher due to the attempt of landing the aircraft. Computer-vision is also necessary in the case of choosing a suitable flat surface such as a field to crash-land on. The latter addition would enhance the emergency or crash-landing ability of the IAS (chapter 4 - prototype 2) by not only being able to perform a smooth crash-landing that keeps the fuselage of the aircraft intact, and hopefully lower the chances of death or serious injuries significantly, but also by choosing a suitable flat surface on the ground.

In the short run however, and before delving into the effort and challenges mentioned above, the IAS will be tested in real-life scenarios outside the simulation environments by integrating it with a fixed-wing Unmanned Aircraft System (UAS). It would be a great moment to witness

the work and effort presented and discussed in this thesis takeoff into the freedom of the blue sky.

REFERENCES

[1] "Airplane Flying Handbook". (2016). [Online]. The Federal Aviation Administration (FAA). https://www.faa.gov/regulations_policies/handbooks_manuals/aviation/airplane_handbook/media/airplane_flying_handbook.pdf. [accessed 2017]

[2] "Automated Flight Control, Chapter Four". [Online]. The Federal Aviation Administration (FAA). Available: https://www.faa.gov/regulations_policies/handbooks_manuals/aviation/advanced_avionics_handbook/media/aah_ch04.pdf. [accessed 2017]

[3] Baumbach, J., Marais, A. and Gonçalves, D. (2015). "Losing the Boxes: Fragmentation as a Source of System Complexity". *The 11th SA INCOSE Conference*, Cape Town, SA.

[4] Martins, E., and Soares, M. (2012) 'Automation Under Suspicion – Case Flight AF-447 Air France'. 222 – 224.

[5] Final report on the accident on 1st June 2009 to the Airbus A330-203 registered F-GZCP operated by Air France Flight AF447 Rio de Janeiro – Paris. (2012). [Online]. *Bureau d'Enquêtes et d'Analyses pour la sécurité de l'aviation civile*. Available: https://www.bea.aero/docspa/2009/f-cp090601.en/pdf/f-cp090601.en.pdf. [accessed 2015]

[6] Salmon, Paul & Walker, Guy & Stanton, Neville. (2015). Pilot Error Versus Sociotechnical Systems Failure: Distributed Situation Awareness Analysis of Air France 447. Theoretical Issues in Ergonomics Science. 17. 1-16.

[7] Khosravani, M. (2012). "Application of Neural Network on Flight Control". *International Journal Of Machine Learning And Computing*, pp.2.

[8] Abbeel P., Ng A. (2004). "Apprenticeship learning via inverse reinforcement learning". *The twenty-first international conference on Machine learning*, pp.1. Alberta, Canada

[9] Qiu J., Delshad I. S., Zhu Q., Nibouche M. and Yao Y. (2017). "A U-model based controller design for non-minimum phase systems: Application to Boeing 747 altitude-hold autopilot". *The 9th International Conference on Modelling, Identification and Control (ICMIC)*, Kunming, China, pp. 122-127.

[10] Nelson, R. C. (1998). "Flight stability and automatic control", Vol. 2. *WCB/McGraw Hill*.

[11] Chowdhary G., Wu T., Cutler, M. and How J. P. (2013). "Rapid transfer of controllers between UAVs using learning-based adaptive control". *The 2013 IEEE International Conference on Robotics and Automation (ICRA)*, Karlsruhe, Germany, pp. 5409-5416.

[12] Zhao Q., Jiang J. (1998) "Reliable state feedback control system design against actuator failures". *Automatica 34*, pp.1267–1272.

[13] Snell S.A., Enns D.F., and Garrard W.L. (1989). "Nonlinear control of a super-maneuverable aircraft". *AIAA Guidance Navigation and Control Conference*. Washington, DC, USA, pp.AIAA-89-3486.

[14] Grzonka S., Grisetti G., and Burgard W. (2012). "A Fully Autonomous Indoor Quadrotor". *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 90-100.

[15] Flynn, A.M. (1988). "Combining Sonar and Infrared Sensors For Mobile Robot Navigation". *The International Journal of Robotics Research*, pp. 5-14.

[16] Roberts J., Stirling T., Zufferey J.C., and Floreano D. (2007). "Quadrotor Using Minimal Sensing For Autonomous Indoor Flight". *The European Micro Air Vehicle Conference and Flight Competition (EMAV2007)*, Toulouse, France.

[17] Green, W., Oh, P., Sevcik, K., and Barrows, G. (2003). “Autonomous Landing for Indoor Flying Robots Using Optic Flow”. *ASME International Mechanical Engineering Congress*, Washington, DC, USA, pp. 1347-1352.

[18] Bills, C., Chen, J., and Saxena, A. (2011). “Autonomous MAV Flight in Indoor Environments using Single Image Perspective Cues”. *The 2011 IEEE International Conference on Robotics and Automation*, Shanghai, China, pp. 5776-5783.

[19] Zufferey J. C. and D. Floreano. (2006). “Fly-inspired visual steering of an ultralight indoor aircraft". *The IEEE Transactions on Robotics*, vol. 22, no. 1, pp. 137-146.

[20] Mejias L., Roberts J., Usher K., Corke P., and P. Campoy. (2006). “Two seconds to touchdown vision-based controlled forced landing”. *The 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, pp. 3527-3532.

[21] Moore R. J. D., Thurrowgood S., Bland D. P., Soccol D., and Srinivasan M. (2009). “A stereo vision system for uav guidance”. *The 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, St. Louis, MO, USA, pp. 3386-3391.

[22] Johnson N. (2008). “Vision-assisted control of a hovering air vehicle in an indoor setting”. *Ph.D. dissertation*, Bringham Young University, Utah, USA.

[23] Kendoul F. and Nonami K. (2009). “A visual navigation system for autonomous flight of micro air vehicles”. *The 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, St. Louis, MO, USA, pp. 3888-3893.

[24] Cherian A., Andersh J., Morellas V., Papanikolopoulos N., and Mettler B. (2009). “Autonomous altitude estimation of a uav using a single onboard camera”. *The 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, St. Louis, MO, USA, pp. 3900-3905.

[25] Fan C., Baoquan S., Cai X., and Liu Y. (2009). “Dynamic visual servoing of a small scale autonomous helicopter in uncalibrated environments”. *The 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, St. Louis, MO, USA, pp. 5301-5306.

[26] Ross S., Barkhudarov N., Shankar K., Wendel A., Dey D., Bagnell J., and Hebert M. (2013). “Learning monocular reactive UAV control in cluttered natural environments”. *The 2013 IEEE International Conference on Robotics and Automation (ICRA)*, Karlsruhe, Germany, pp. 1765-1772.

[27] Safadinho, D.; Ramos, J.; Ribeiro, R.; Filipe, V.; Barroso, J.; Pereira, A. UAV Landing Using Computer Vision Techniques for Human Detection. Sensors 2020, 20, 613.

[28] Roggi, G.; Niccolai, A.; Grimaccia, F.; Lovera, M. A Computer Vision Line-Tracking Algorithm for Automatic UAV Photovoltaic Plants Monitoring Applications. Energies 2020, 13, 838.

[29] T. Khuc, T. Nguyen, H. Dao and F. Catbas, "Swaying displacement measurement for structural monitoring using computer vision and an unmanned aerial vehicle", Measurement, vol. 159, p. 107769, 2020. Available: 10.1016/j.measurement.2020.107769.

[30] F. Patrona, P. Nousi, I. Mademlis, A. Tefas and I. Pitas, "Visual Object Detection For Autonomous UAV Cinematography", Proceedings of the Northern Lights Deep Learning Workshop, vol. 1, p. 6, 2020. Available: 10.7557/18.5099.

[31] M. Rizk, A. Mroue, M. Farran and J. Charara, "Real-Time SLAM Based on Image Stitching for Autonomous Navigation of UAVs in GNSS-Denied Regions," 2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS), Genova, Italy, 2020, pp. 301-304.

[32] F. L. Junior, L. A. S. Moreira, E. M. Moreira, T. J. M. Baldivieso, M. S. Brunaes and P. F. F. Rosa, "UAV path automation using visual waypoints acquired from the ground," 2020 IEEE 29th International Symposium on Industrial Electronics (ISIE), Delft, Netherlands, 2020, pp. 579-585.

[33] F. Baldini, A. Anandkumar and R. M. Murray, "Learning Pose Estimation for UAV Autonomous Navigation and Landing Using Visual-Inertial Sensor Data," 2020 American Control Conference (ACC), Denver, CO, USA, 2020, pp. 2961-2966.

[34] Manocha A. and Sharma A. (2009). "Three Axis Aircraft Autopilot Control Using Genetic Algorithms : An Experimental Study". *The 2009 IEEE International Advance Computing Conference*, Patiala, India, pp. 171-174.

[35] Tsourdos A. and White B. (2005) "Adaptive flight control design for nonlinear missile". *Control Engineering Practice*, vol. 13, no. 3, pp. 373-382. Available: 10.1016/j.conengprac.2004.04.023.

[36] We D., Xiong H., and Fu J. (2015). "Aircraft Autopilot Pitch Control Based on Fuzzy Active Disturbance Rejection Control". *The 2015 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration*, Wuhan, China, pp. 144-147.

[37] Beygi N., Beigy M., and Siahi, M. (2015). "Design of Fuzzy self-tuning PID controller for pitch control system of aircraft autopilot". *Cornell University*.

[38] Jeevan H. L., Narahari H. K. and Sriram A. T. (2018). "Development of pitch control subsystem of autopilot for a fixed wing unmanned aerial vehicle". *The 2nd International Conference on Inventive Systems and Control (ICISC)*, Coimbatore, India, pp. 1233-1238.

[39] Motea Z., Wahid H., Zahid J., Lwin S., and Hassan A. (2018). "A Comparative Analysis of Intelligent and PID Controllers for an Aircraft Pitch Control System". *Applications of Modelling and Simulation (AMS)*, 2(1), pp.17-25.

[40] Hecht-Nielsen R. (1990). "Neurocomputing". *Addison-Wesley Pub. Co*.

[41] Young-Keun P., Gyungho L. (1995). "Applications of neural networks in high-speed communication networks". *The IEEE Communications Magazine*, vol.33, no.10, pp.68-74.

[42] Tino, P., Schittenkopf, C. and Dorffner, G. (2001). "Financial volatility trading using recurrent neural networks". *The IEEE Transactions on Neural Networks*, vol. 12, no. 4, pp. 865-874.

[43] Burr, G., Shelby, R., Sidler, S., di Nolfo, C., Jang, J., Boybat, I., Shenoy, R., Narayanan, P., Virwani, K., Giacometti, E., Kurdi, B. and Hwang, H. (2015). "Experimental Demonstration and Tolerancing of a Large-Scale Neural Network (165 000 Synapses) Using Phase-Change Memory as the Synaptic Weight Element". *The IEEE Transactions on Electron Devices*, vol. 62, no. 11, pp. 3498-3507

[44] Miller W.T. (1989). "Real-time application of neural networks for sensor-based control of robots with vision". *The IEEE Transactions on Systems, Man, and Cybernetics*, vol. 19, no. 4, pp. 825-831.

[45] Pashilkar, A., Sundararajan, N., and Saratchandran, P. (2006). "A fault-tolerant neural aided controller for aircraft auto-landing". *Aerospace Science And Technology*, 10(1), 49-61.

[46] Soares F. and Burken J. (2006). "A Flight Test Demonstration of On-line Neural Network Applications in Advanced Aircraft Flight Control System". *The 2006 International Conference on Computational Intelligence for Modelling Control and Automation and International Conference on Intelligent Agents Web Technologies and International Commerce (CIMCA'06)*, Sydney, Australia, pp. 136-136.

[47] Collotta M., Pau G., and Caponetto R. (2014). "A real-time system based on a neural network model to control hexacopter trajectories". *International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM)*, Ischia, Italy, pp. 222-227.

[48] Lin Q., Cai Z., Wang Y., Yang J. and Chen L. (2013). "Adaptive Flight Control Design for Quadrotor UAV Based on Dynamic Inversion and Neural Networks". *The Third International Conference on Instrumentation, Measurement, Computer, Communication and Control (IMCCC)*, Shenyang, China, pp. 1461-1466.

[49] Jongho S., Kim, H., Youdan K., & Dixon W. (2012). "Autonomous Flight of the Rotorcraft-Based UAV Using RISE Feedback and NN Feedforward Terms". *The IEEE Transactions On Control Systems Technology*, 20(5), 1392-1399.

[50] Suresh, S., & Kannan, N. (2008). “Direct adaptive neural flight control system for an unstable unmanned aircraft”. *Applied Soft Computing*, 8(2), 937-948.

[51] Frye M. T, and Provence R. S. (2014). “Direct Inverse Control using an Artificial Neural Network for the Autonomous Hover of a Helicopter”. *The 2014 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, San Diego, CA, USA, pp. 4121-4122.

[52] Xu, B., Wang, D., Sun, F., & Shi, Z. (2012). “Direct neural discrete control of hypersonic flight vehicle”. *Nonlinear Dynamics*, 70(1), 269-278.

[53] Savran, A., Tasaltin, R., & Becerikli, Y. (2006). “Intelligent adaptive nonlinear flight control for a high performance aircraft with neural networks”. *ISA Transactions*, 45(2), 225-247.

[54] Matassini T., Shin H., Tsourdos A. and Innocenti M. (2016) "Adaptive Control with Neural Networks-based Disturbance Observer for a Spherical UAV", *IFAC-PapersOnLine*, vol. 49, no. 17, pp. 308-313. Available: 10.1016/j.ifacol.2016.09.053.

[55] Rajagopalan A., Faruqi F., Nandagopal D. (2015). “Intelligent Missile using Artificial Neural Networks”. *Artificial Intelligence Research*, Vol 4, No 1.

[56] DeMarse, T., Dockendorf, K. (2005). “Adaptive flight control with living neuronal networks on microelectrode arrays”. *The 2005 IEEE International Joint Conference on Neural Networks*, pp. 1548-1551 vol. 3.

[57] Soares, F., Burken, J., & Marwala, T. (2006). “Neural Network Applications in Advanced Aircraft Flight Control System, a Hybrid System, a Flight Test Demonstration”. *ICONIP*, 684–691.

[58] Samal, M., Anavatti, S., & Garratt, M. (2008). “Neural Network Based System Identification for Autonomous Flight of an Eagle Helicopter”. *17Th World Congress, The International Federation Of Automatic Control*, Volume 41, Issue 2, pp. 7421-7426.

[59] Calise, A. (1996). “Neural networks in nonlinear aircraft flight control”. *The IEEE Aerospace And Electronic Systems Magazine*, 11(7), pp. 5-10.

[60] Shin, D., & Kim, Y. (2004). “Reconfigurable Flight Control System Design Using Adaptive Neural Networks”. *IEEE Transactions On Control Systems Technology, 12*(1), pp. 87-100.

[61] Hatamleh, K., Al-Shabi, M., Al-Ghasem, A., & Asad, A. (2015). “Unmanned Aerial Vehicles parameter estimation using Artificial Neural Networks and Iterative Bi-Section Shooting method”. *Applied Soft Computing, 36*, pp. 457-467.

[62] Shin, Y., & Calise, A. (2005). “Application of Adaptive Autopilot Designs for an Unmanned Aerial Vehicle”. *American Institute Of Aeronautics And Astronautics*, 10.2514/6.2005-6166.

[63] Tang, W.; Wang, L.; Gu, J.; Gu, Y. Single Neural Adaptive PID Control for Small UAV Micro-Turbojet Engine. Sensors 2020, 20, 345.

[64] Z. Cheng, H. Pei and S. Li, "Neural-Networks Control for Hover to High-Speed-Level-Flight Transition of Ducted Fan UAV With Provable Stability," in IEEE Access, vol. 8, pp. 100135-100151, 2020.

[65] Xu, H.; Fang, G.; Fan, Y.; Xu, B.; Yan, J. Universal Adaptive Neural Network Predictive Algorithm for Remotely Piloted Unmanned Combat Aerial Vehicle in Wireless Sensor Network. Sensors 2020.

[66] Yañez-Badillo, H.; Tapia-Olvera, R.; Beltran-Carbajal, F. Adaptive Neural Motion Control of a Quadrotor UAV. Vehicles 2020.

[67] Zhang, Q.; Chen, X.; Xu, D. Adaptive Neural Fault-Tolerant Control for the Yaw Control of UAV Helicopters with Input Saturation and Full-State Constraints. Appl. Sci. 2020.

[68] L. Guettal, A. Chelihi, M. M. Touba, H. Glida and R. Ajgou, "Neural Network-based Adaptive Backstepping Controller for UAV Quadrotor system," 020 1st International Conference on Communications, Control Systems and Signal Processing (CCSSP), EL OUED, Algeria, 2020, pp. 388-393.

[69] J. Cervantes-Rojas, F. Muñoz, I. Chairez, I. González-Hernández and S. Salazar, "Adaptive tracking control of an unmanned aerial system based on a dynamic neural-fuzzy disturbance estimator", ISA Transactions, vol. 101, pp. 309-326, 2020. Available: 10.1016/j.isatra.2020.02.012.

[70] Gomez-Avila, J.; Villaseñor, C.; Hernandez-Barragan, J.; Arana-Daniel, N.; Alanis, A.Y.; Lopez-Franco, C. Neural PD Controller for an Unmanned Aerial Vehicle Trained with Extended Kalman Filter. Algorithms 2020, 13, 40.

[71] Brooks, R. (1986). "A robust layered control system for a mobile robot". *IEEE Journal on Robotics and Automation*, 2(1), pp.14-23.

[72] Steingrube, S., Timme, M., Wörgötter, F. et al. (2010) "Self-organized adaptation of a simple neural circuit enables complex robot behaviour". *Nature Phys* 6, pp. 224–230.

[73] Shinzato P. Y., Grassi V., Osorio F. S. and Wolf D. F. "Fast visual road recognition and horizon detection using multiple artificial neural networks," *Intelligent Vehicles Symposium (IV), 2012 IEEE*, Alcala de Henares, Spain, pp. 1090-1095.

[74] Oliveira M. B. W. D. and Neto A. D. A. "Optimization of Traffic Lights Timing Based on Multiple Neural Networks," *2013 IEEE 25th International Conference on Tools with Artificial Intelligence*, Herndon, VA, USA, pp. 825-832.

[75] Huang L., Wu Q., Chen Y., Hong S. and Huang X. "Gesture Recognition Based on Fusion Features from Multiple Spiking Neural Networks". *The Fifth International Conference on Communication Systems and Network Technologies (CSNT)*, Gwalior, India, pp. 1167-1171.

[76] Phan K. T., Maul T. H., and Vu T. T. (2015) "A parallel circuit approach for improving the speed and generalization properties of neural networks". *The 11th International Conference on Natural Computation (ICNC)*, Zhangjiajie, China, pp. 1-7.

[77] Dennis L. A., Fisher M., Aitken J. M., Veres S. M., Gao Y., Shaukat A. and Burroughes G. (2014) "Reconfigurable Autonomy", KI - Künstliche Intelligenz, vol. 28, no. 3, pp. 199-207. Available: 10.1007/s13218-014-0308-1.

[78] Argall, B., Chernova, S., Veloso, M., and Browning, B. (2009). "A survey of robot learning from demonstration". *Robotics And Autonomous Systems*, 57(5), pp. 469-483.

[79] Michie, D., Bain, M., and Hayes-Michie, J.E. (1990). "Cognitive Models from Subcognitive Skills". *Knowledge-base Systems in Industrial Control*.

[80] Iiguni, Y., Akiyoshi, H., & Adachi, N. (1998). "An intelligent landing system based on a human skill model". *IEEE Trans. Aerosp. Electron. Syst*., 34(3), pp. 877-882.

[81] Matsumoto, T., Vismari, L., Camargo, J. (2014). "A method to implement and to evaluate a learning-based Piloting Autonomous System for UAS". *International Conference on Unmanned Aircraft Systems (ICUAS)*, Florida, USA, pp.195-199.

[82] Sammut, C., Hurst, S., Kedzier, D., & Michie, D. (1992). "Learning to Fly".

[83] Sammut, C. (1992). "Automatically constructing control systems by observing human behaviour". *Proc. of the Internat. Workshop on Inductive Logic Programming.*

[84] Sheh, R., Hengst, B., & Sammut, C., (2011). "Behavioural cloning for driving robots over rough terrain". *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Francisco, CA, USA, pp. 732-737.

[85] Neu, G., and Szepesvari, C. (2012). "Apprenticeship Learning using Inverse Reinforcement Learning and Gradient Methods". *The Twenty-Third Conference on Uncertainty in Artificial Intelligence*, Vancouver, Canada, pp. 295-302.

[86] Asta, S., and Ozcan, E. (2014). "An Apprenticeship Learning Hyper-Heuristic for Vehicle Routing in HyFlex". *The 2014 IEEE Symposium on Evolving and Autonomous Learning Systems (EALS)*, Orlando, FL, USA, pp. 65-72.

[87] Messer, O., & Ranchod, P. (2014). "The Use of Apprenticeship Learning Via Inverse Reinforcement Learning for Generating Melodies". *ICMC|SMC|2014*, Athens, Greece, pp. 1781-1787

[88] Abbeel, P., Coates, A., and Ng, A. Y. (2010). "Autonomous Helicopter Aerobatics through Apprenticeship Learning". *The International Journal of Robotics Research*, 29(13), 1608–1639.

[89] Bloem, M., & Bambos, N. (2015). "Ground Delay Program Analytics with Behavioral Cloning and Inverse Reinforcement Learning". *Journal of Aerospace Information Systems* vol. 12, iss. 3, pp. 299.

[90] Michini, B., Walsh, T., Agha-Mohammadi, A., and How, J. (2015). "Bayesian Nonparametric Reward Learning From Demonstration". *IEEE Trans. Robot., 31*(2), 369-386.

[91] Viswanathan, R. and Lakshmi, P. (2017). "A Novel Method for Controlling the Roll Angle of Aircraft using Sliding Mode Control Methodology". *Journal of The Institution of Engineers (India)*, Series C, 99(4), pp.369-372.

[92] Ali S. U., Samar R. and Shah M. Z. (2017). "UAV lateral path following: Nonlinear sliding manifold for limited actuation," *The 36th Chinese Control Conference (CCC)*, Dalian, China, pp. 1348-1353.

[93] Botkin N., and Turova V. (2015). "Aircraft Runway Acceleration in the Presence of Severe Wind Gusts". *The 27th IFIP Conference on System Modeling and Optimization (CSMO)*, Sophia Antipolis, France. pp.147-158.

[94] Lungu, R. and Lungu, M. (2017). "Automatic landing system using neural networks and radio-technical subsystems". *Chinese Journal of Aeronautics*, 30(1), pp.399-411.

[95] Evdokimenkov V., Kim R., Krasilshchikov M., and Sebrjakov G. (2016). "Individually Adapted Neural Network for Pilot's Final Approach Actions Modeling". *In: Cheng L., Liu Q., Ronzhin A. (eds) Advances in Neural Networks* – ISNN 2016. ISNN 2016. Lecture Notes in Computer Science, vol 9719. Springer, Cham.

[96] Muliadi J., and Kusumoputro B. (2018). "Neural Network Control System of UAV Altitude Dynamics and Its Comparison with the PID Control System". *Journal of Advanced Transportation*, pp.1-18.

[97] Mumm N. C. and Holzapfel F. (2017). "Vertical speed command performance improvement of a load factor command based autopilot for automatic landing by shaping the desired command during flare". *The 2017 IEEE Conference on Control Technology and Applications (CCTA)*, Mauna Lani, HI, USA, pp. 1117-1122.

[98] Stukenborg P., and Luckner R. (2018). "Evaluating the influence of continuous flap settings on the take-off performance of an airliner using flight simulation". *CEAS Aeronautical Journal*, 9(4), pp.671-681.

[99] Dong J., Zhou D., Shao C., and Wu S. (2018). "Nonlinear system controllability analysis and autopilot design for bank-to-turn aircraft with two flaps". *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 233(5), pp.1772-1783.

[100] Sadeghzadeh, I. & Zhang, Y. (2011). "A Review on Fault-Tolerant Control for Unmanned Aerial Vehicles (UAVs)". *AIAA Infotech at Aerospace Conference and Exhibit*. St. Louis, Missouri, USA. 10.2514/6.2011-1472.

[101] Faheem M., Rao & Aziz, Sumair & Khalid, Adnan & Bashir, Mudassar & Yasin, Amanullah. (2015). "UAV Emergency Landing Site Selection System using Machine Vision- Journal of Machine Intelligence". *Journal of Machine Intelligence*, 1. 13-20. 10.21174/jomi.v1i1.24.

[102] Kugler M. E., and Holzapfel F. (2015) "Enhancing the auto flight system of the SAGITTA Demonstrator UAV by fault detection and diagnosis". *The IEEE International Conference on Aerospace Electronics and Remote Sensing Technology (ICARES)*, Bali, Indonesia, pp. 1-7.

[103] Li P., Chen X., and Li C. (2014) "Emergency landing control technology for UAV". *The 2014 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC),* Yantai, China, pp. 2359-2362.

[104] Braga J. R. G., Velho H. F. C., Conte G., Doherty P. and Shiguemori É. H. (2016). "An image matching system for autonomous UAV navigation based on neural network". *The 14th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, Phuket, Thailand, pp. 1-6.

[105] Oettershagen P., Stastny T., Mantel T., Melzer A., Rudin K., Gohl P., Agamennoni G., Alexis K., and Siegwart R. (2016). "Long-Endurance Sensing and Mapping using a Hand-Launchable Solar-Powered UAV". *Springer Tracts in Advanced Robotics*, Volume 113, pp. 441-454.

[106] Lin S., and Kernighan BW. (1973) "An effective heuristic algorithm for the travelingsalesman problem". *Operations research*, 21(2), pp. 498–516.

[107] Karaman S., and Frazzoli E. (2010) "Incremental sampling-based algorithms for optimal motion planning". *CoRR* abs/1005.0416.

[108] Dong Z., Li W. and Zhou Y. (2016). "An autonomous navigation scheme for UAV in approach phase". *IEEE Chinese Guidance, Navigation and Control Conference (CGNCC)*, Nanjing, China, pp. 982-987.

[109] Ospina L., Abdullah N., Jahdali O. A. and Oteniya O. (2017). "Design of a cruise control system prototype for the GT500's airplane". *2017 Systems and Information Engineering Design Symposium (SIEDS)*, Charlottesville, VA, USA, 2017, pp. 79-83.

[110] Baker B. (2017). "The Global Positioning System". *Physics Capstone Project.* Paper 44. available: http://digitalcommons.usu.edu/phys_capstoneproject/44. [accessed 2017]

[111] "GNSS Frequently Asked Questions - GPS." *FAA seal*. N.p., 20 Dec. 2016. Available: https://www.faa.gov/about/office_org/headquarters_offices/ato/service_units/techops/navservices/gnss/faq/gps/. [accessed 2017]

[112] Duanzhang L., Peng C. and Nong C. (2016). "A generic trajectory prediction and smoothing algorithm for flight management system". *2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC)*, Nanjing, China, pp. 1969-1974.

[113] Khan M. S. I., Tiasha M., and Barman S. (2017). "Auto landing sequence for an unmanned aerial vehicle at a fixed point". *2017 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, Cox's Bazar, Bangladesh, pp. 175-180.

[114] Isaacs J. T., Ezal K. O. and Hespanha J. P. (2016). "Local carrier-based precision approach and landing system". *2016 IEEE 55th Conference on Decision and Control (CDC)*, Las Vegas, NV, USA, pp. 6284-6290.

[115] Kim E. and Choi D. (2016). "A UWB positioning network enabling unmanned aircraft systems auto land". *Aerospace Science and Technology*, vol. 58, pp. 418-426.

[116] Tang D., Jiao Y., and Chen J. (2016). "On Automatic Landing System for carrier plane based on integration of INS, GPS and vision". *2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC)*, Nanjing, China, pp. 2260-2264.

[117] Vladimir T., Jeon D., Kim D. H., Chang C. H., and Kim J. (2012). "Experimental Feasibility Analysis of ROI-Based Hough Transform for Real-Time Line Tracking in Auto-Landing of UAV". *2012 IEEE 15th International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops*, Shenzhen, China, pp. 130-135.

[118] Zhang Y., Wang Y. and Han Z. (2016). "Vision-aided navigation for fixed-wing UAV's autonomous landing" *2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC)*, Nanjing, China, pp. 988-992.

[119] Oszust, M., Kapuscinski, T., Warchol, D., Wysocki, M., Rogalski, T., Pieniazek, J., Kopecki, G., Ciecinski, P. and Rzucidlo, P. (2018). "A vision-based method for supporting autonomous aircraft landing". *Aircraft Engineering and Aerospace Technology*, 90(6), pp. 973-982

[120] Theis, J., Ossmann, D., Thielecke, F. and Pfifer, H. (2018). "Robust autopilot design for landing a large civil aircraft in crosswind". *Control Engineering Practice*, 76, pp.54-64.

[121] de Bruin, A. and Jones, T. (2016). "Accurate Autonomous Landing of a Fixed-Wing Unmanned Aircraft under Crosswind Conditions". *IFAC-PapersOnLine*, 49(17), pp.170-175.

[122] Zhang, L., Zhai, Z., He, L. and Niu, W. (2019). "Infrared-Based Autonomous Navigation for Civil Aircraft Precision Approach and Landing". *IEEE Access*, 7, pp.28684-28695.

[123] Bian, Q., Nener, B. and Wang, X. (2018). "Control parameter tuning for aircraft crosswind landing via multi-solution particle swarm optimization". *Engineering Optimization*, 50(11), pp.1914-1925.

[124] Xiong H., Yuan R., Yi J., Fan G., and Jing F. (2011). "Disturbance Rejection in UAV's velocity and attitude control: Problems and solutions". *Proceedings of the 30th Chinese Control Conference*, Yantai, China, pp. 6293-6298.

[125] Smith J., Jinya S., Cunjia L., and Wen-Hua C. (2017). "Disturbance Observer Based Control with Anti-Windup Applied to a Small Fixed Wing UAV for Disturbance Rejection". *Journal of Intelligent & Robotic Systems*, Volume 88, Issue 2–4, pp. 329–346.

[126] Khaghani M. and Skaloud J. (2016). "Evaluation of Wind Effects on UAV Autonomous Navigation Based on Vehicle Dynamic Model", *The 29th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2016)*, pp. 1432 - 1440, Portland, Oregon, USA.

[127] Park S. (2016). "Autonomous crabbing by estimating wind using only GPS velocity". *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 3, pp. 1399-1407.

[128] Van Es G., Van der Geest P., and Nieuwpoort T. (2001). "Safety aspects of aircraft operations in crosswind". *National Aerospace Laboratory NLR*, Amsterdam, Netherlands.

[129] Van Es, G. et. al. (1998). "Safety Aspects of Aircraft Performance on Wet and Contaminated Runways". *The 10th annual European Aviation Safety Seminar (FSF)*, Amsterdam, Netherlands.

[130] Mori R. and Suzuki S. (2009). "Analysis of pilot landing control in crosswind using neural networks," *2009 IEEE Aerospace conference*, Big Sky, MT, USA, pp. 1-10.

[131] Mori R. (2009). "Analysis of Pilot Landing Control Using Neural Network. 10.1109/AERO.2009.4839640.

[132] Ismail S., Pashilkar A. A., and Ayyagari R. (2015). "Phase compensation and anti-windup design for neural-aided sliding mode fault-tolerant autoland controller". *2015 International Conference on Cognitive Computing and Information Processing(CCIP)*, Noida, India, pp. 1-7.

[133] Alvis E., Bhatt D., Hall B., Driscoll K., Murugesan A. (2018). ". Final Technical Report for NASA Project. Assurance Reasoning for Increasingly Autonomous Systems (ARIAS). [online] Available at: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20180006312.pdf.

[134] Atkins E. (2016). "Safe Autonomous Manned and Unmanned Flight in Off-Nominal Conditions". [Presentation]. *Future Technologies Conference*, San Francisco, USA.

[135] "How X-Plane Works | X-Plane", *X-Plane*, 2019. [Online]. Available: https://www.x-plane.com/desktop/how-x-plane-works/. [accessed 2018].

[136] Wei, F., Amaya-Bower, L., Gates, A., Rose, D. and Vasko, T. (2016). "The Full-Scale Helicopter Flight Simulator Design and Fabrication". *57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*. San Diego, California, USA.

[137] Jirgl, M., Boril, J., Jalovecky, R. (2015). "The identification possibilities of the measured parameters of an aircraft model and pilot behavior model on the flight simulator". *International Conference on Military Technologies (ICMT)*, Czech Republic, pp. 1-5.

[138] Kaviyarasu, A. and Senthil Kumar, K. (2014). "Simulation of Flapping-wing Unmanned Aerial Vehicle using X-plane and Matlab/Simulink". *Defence Science Journal*, 64(4), pp.327-331.

[139] J. M. M. Junior, T. Khamvilai, L. Sutter and E. Feron, "Test platform for autopilot system embedded in a model of multi-core architecture using X-Plane flight simulator," *2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC)*, San Diego, CA, USA, 2019, pp. 1-6.

[140] I. G. A. Agastya T. and B. Kusumoputro, "Data Acquisition for Rocket Model Identification with Aircraft Simulator X-Plane and MATLAB," *2019 16th International Conference on Quality in Research (QIR): International Symposium on Electrical and Computer Engineering*, Padang, Indonesia, 2019, pp. 1-4.

[141] M. G. Michailidis, M. Agha, M. J. Rutherford and K. P. Valavanis, "A Software in the Loop (SIL) Kalman and Complementary Filter Implementation on X-Plane for UAVs," *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, Atlanta, GA, USA, 2019, pp. 1069-1076.

[142] Kaviyarasu, A. Saravanakumar and M. L. Venkatesh, "Hardware in Loop Simulation of a Way Point Navigation Using Matlab/Simulink and X-Plane Simulator," *2019 International Conference on Intelligent Sustainable Systems (ICISS)*, Palladam, Tamilnadu, India, 2019, pp. 332-335.

[143] McCaffrey J. (2014). "Understanding Neural Networks using .NET", [Presentation]. *The Microsoft 2014 Build Conference*, San Francisco, USA.

[144] Heaton J. (2009). "Introduction to neural networks with Java". *Heaton Research Inc*., St. Louis, MO, USA.

[145] LeCun Y., Bottou L., Orr G. B., Müller K. R., Orr G. B., and Müller K. R. (1998). "Efficient backprop in Neural Networks: Tricks of the Trade". *Springer*, pp. 9-50.

[146] Winter K., Hayes I. J., and Colvin R. (2010). "Integrating Requirements: The Behavior Tree Philosophy". *The 8th IEEE International Conference on Software Engineering and Formal Methods*, Pisa, Italy, pp. 41-50.

[147] J. M. (2012). "Calculating a bearing between points in location-aware apps". *Intel Software*, [Online]. Available: https://software.intel.com/en-us/blogs/2012/11/30/calculating-a-bearing-between-points-in-location-aware-apps. [accessed 2017]

[148] Steig J. (2009). "Average Rate of Change Function". *Mesa Community College*, [Online]. Available: http://www.mesacc.edu/~marfv02121/readings/average/. [accessed 2016]

[149] L'hotellier E. and Salzmann J. (2017). "Top of descent calculation". [online] *The International Virtual Aviation Organisation, IVAO.*
Available: https://www.ivao.aero/training/documentation/books/SPP_APC_Top_of_descent.pdf.
[accessed 2018]

[150] Schuirmann D. J. (1987). "A comparison of the Two One-Sided Tests Procedure and the Power Approach for assessing the equivalence of average bioavailability". *Journal of Pharmacokinetics and Biopharmaceutics*. 15 (6): pp. 657–680.

APPENDIX A

TABLE A.1

RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS WHEN MAINTAINING A FIFTEEN DEGREES PITCH DURING TAKEOFF COMPARED WITH THE HUMAN PILOT.

Equivalence Test for Means
Unequal Sample Sizes
$\alpha = 0.05$

	IAS	**Human**
Mean	15.24	15.17
Variance	0.36	0.005
Observations	315	21
Pooled Variance	0.34	
Hypothesized Mean Difference	0.8	
df	334	
t Stat	5.63	-6.55
P(T<=t) one-tail	0.000	0.000
T Critical one-tail	1.65	
P(T<=t) two-tail	0.000	
T Critical Two-tail	1.98	

Means are Equivalent because p1 & p2 < 0.05

TABLE A.2

RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED WITH THE STANDARD AUTOPILOT WHEN MAINTAINING AN ALTITUDE OF 14000 FT.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	**AP**
Mean	14000.49	14000.92
Variance	0.13	0.00
Observations	420	420
Pooled Variance	0.06	
Hypothesized Mean Difference	0.80	
df	838.00	
t Stat	70.87	-20.68
P(T<=t) one-tail	0.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.96	

Means are Equivalent because p1 & p2 < 0.05

TABLE A.3

RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED WITH THE STANDARD AUTOPILOT WHEN MAINTAINING AN ALTITUDE OF 32000 FT.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	**AP**
Mean	32000.72	32000.03
Variance	0.24	0.00
Observations	420	420
Pooled Variance	0.12	
Hypothesized Mean Difference	0.80	
df	838.00	
t Stat	4.36	-61.79
P(T<=t) one-tail	0.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.96	

Means are Equivalent because p1 & p2 < 0.05

TABLE A.4

RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED WITH THE STANDARD AUTOPILOT WHEN MAINTAINING AN ALTITUDE OF 4000 FT.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	**AP**
Mean	4000.38	4000.28
Variance	0.23	0.01
Observations	420	420
Pooled Variance	0.12	
Hypothesized Mean Difference	0.80	
df	838.00	
t Stat	29.38	-37.06
P(T<=t) one-tail	0.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.96	

Means are Equivalent because p1 & p2 < 0.05

TABLE A.5

RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED WITH THE STANDARD AUTOPILOT WHEN MAINTAINING A CLIMB RATE OF 500 FT/MIN.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	**AP**
Mean	498.89	413.88
Variance	4056.47	36247.80
Observations	147	147
Pooled Variance	20152.14	
Hypothesized Mean Difference	0.80	
df	292.00	
t Stat	-5.09	-5.18
P(T<=t) one-tail	1.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.97	

Cannot conclude means are equivalent

TABLE A.6

RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED WITH THE STANDARD AUTOPILOT WHEN MAINTAINING A CLIMB RATE OF 1500 FT/MIN.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	**AP**
Mean	1503.47	1327.31
Variance	5403.78	3514.27
Observations	147	147
Pooled Variance	4459.03	
Hypothesized Mean Difference	0.80	
df	292.00	
t Stat	-22.51	-22.72
P(T<=t) one-tail	1.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.97	

Cannot conclude means are equivalent

TABLE A.7

RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED WITH THE STANDARD AUTOPILOT WHEN MAINTAINING A CLIMB RATE OF 2500 FT/MIN.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	**AP**
Mean	2519.27	2347.46
Variance	12673.10	4014.60
Observations	147	147
Pooled Variance	8343.85	
Hypothesized Mean Difference	0.80	
df	292.00	
t Stat	-16.05	-16.20
P(T<=t) one-tail	1.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.97	

Cannot conclude means are equivalent

TABLE A.8

RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED WITH THE STANDARD AUTOPILOT WHEN MAINTAINING A CLIMB (SINK) RATE OF -500 FT/MIN.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	**AP**
Mean	-491.51	-486.45
Variance	3297.94	640.19
Observations	147	147
Pooled Variance	1969.07	
Hypothesized Mean Difference	0.80	
df	292.00	
t Stat	1.13	0.82
P(T<=t) one-tail	0.13	0.21
T Critical one-tail	1.65	
P(T<=t) two-tail	0.26	
T Critical Two-tail	1.97	

Cannot conclude means are equivalent

TABLE A.9

RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED WITH THE STANDARD AUTOPILOT WHEN MAINTAINING A CLIMB (SINK) RATE OF -1000 FT/MIN.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	**AP**
Mean	-988.85	-1187.84
Variance	4295.21	647.30
Observations	147	147
Pooled Variance	2471.25	
Hypothesized Mean Difference	0.80	
df	292.00	
t Stat	-34.18	-34.46
P(T<=t) one-tail	1.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.97	

Cannot conclude means are equivalent

TABLE A.10

RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED WITH THE STANDARD AUTOPILOT WHEN MAINTAINING A CLIMB (SINK) RATE OF -2000 FT/MIN.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	**AP**
Mean	-1996.08	-1886.80
Variance	6133.68	1901.15
Observations	147	147
Pooled Variance	4017.41	
Hypothesized Mean Difference	0.80	
df	292.00	
t Stat	14.89	14.67
P(T<=t) one-tail	0.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.97	

Means are Equivalent because p1 & p2 < 0.05

TABLE A.11

RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED WITH THE STANDARD AUTOPILOT WHEN MAINTAINING A SPEED OF 320 KNOTS.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	**AP**
Mean	319.98	320.00
Variance	0.00	0.00
Observations	180	180
Pooled Variance	0.00	
Hypothesized Mean Difference	0.80	
df	358.00	
t Stat	337.34	-321.25
P(T<=t) one-tail	0.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.97	

Means are Equivalent because p1 & p2 < 0.05

TABLE A.12

RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED WITH THE STANDARD AUTOPILOT WHEN MAINTAINING A SPEED OF 350 KNOTS.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	**AP**
Mean	349.98	350.00
Variance	0.00	0.00
Observations	180	180
Pooled Variance	0.00	
Hypothesized Mean Difference	0.80	
df	358.00	
t Stat	167.35	-159.95
P(T<=t) one-tail	0.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.97	

Means are Equivalent because p1 & p2 < 0.05

TABLE A.13

RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED WITH THE STANDARD AUTOPILOT WHEN MAINTAINING A SPEED OF 230 KNOTS.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	**AP**
Mean	229.95	230.00
Variance	0.00	0.00
Observations	180	180
Pooled Variance	0.00	
Hypothesized Mean Difference	0.80	
df	358.00	
t Stat	305.61	-268.03
P(T<=t) one-tail	0.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.97	

Means are Equivalent because p1 & p2 < 0.05

TABLE A.14

RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED WITH THE STANDARD AUTOPILOT AND THE HUMAN PILOT WHEN MAINTAINING A 3 DEGREES GLIDESLOPE DURING FINAL APPROACH IN CALM WEATHER.

Equivalence Test for Means
Unequal Sample Sizes
$\alpha = 0.05$

	IAS	**AP/Human**
Mean	3.02	2.99
Variance	0.0009	0.0002
Observations	1059	106
Pooled Variance	0.0009	
Hypothesized Mean Difference	0.8	
df	1163	
t Stat	248.07	-268.5
P(T<=t) one-tail	0	0
T Critical one-tail	1.64	
P(T<=t) two-tail	0	
T Critical Two-tail	1.96	

Means are Equivalent because p1 & p2 < 0.05

TABLE A.15

RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED WITH THE STANDARD AUTOPILOT WHEN MAINTAINING A THREE DEGREES GLIDESLOPE DURING FINAL APPROACH IN EXTREME WEATHER CONDITIONS.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	AP
Mean	3.03	2.93
Variance	0.07	0.02
Observations	1429	1429
Pooled Variance	0.05	
Hypothesized Mean Difference	0.80	
df	2856.00	
t Stat	89.52	-110.65
P(T<=t) one-tail	0.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.96	

Means are Equivalent because p1 & p2 < 0.05

TABLE A.16

RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED WITH THE STANDARD AUTOPILOT AND THE HUMAN PILOT WHEN MAINTAINING THE CENTRELINE OF THE LANDING RUNWAY (0 DEGREES ANGLE) DURING FINAL APPROACH AND LANDING IN CALM WEATHER.

Equivalence Test for Means
Unequal Sample Sizes
$\alpha = 0.05$

	IAS	AP/Human
Mean	0.00004	-0.00002
Variance	0.000	0.000
Observations	1246	135
Pooled Variance	0.000	
Hypothesized Mean Difference	0.8	
df	1379	
t Stat	4926.46	-4944.38
P(T<=t) one-tail	0.000	0.000
T Critical one-tail	1.64	
P(T<=t) two-tail	0.000	
T Critical Two-tail	1.96	

Means are Equivalent because p1 & p2 < 0.05

APPENDIX B

Research papers:

1. H. Baomar and P. J. Bentley, "An Intelligent Autopilot System that learns piloting skills from human pilots by imitation", 2016 International Conference on Unmanned Aircraft Systems (ICUAS), Arlington, VA, USA, 2016, pp. 1023-1031.

2. H. Baomar and P. J. Bentley, "An Intelligent Autopilot System that Learns Flight Emergency Procedures by Imitating Human Pilots", 2016 IEEE Symposium Series on Computational Intelligence (SSCI), Athens, 2016, pp. 1-9.

3. H. Baomar and P. J. Bentley, "Autonomous Navigation and Landing of Airliners Using Artificial Neural Networks and Learning by Imitation", 2017 IEEE Symposium Series on Computational Intelligence (SSCI), Hawaii, USA, 2017.

4. H. Baomar and P. J. Bentley, "Autonomous Landing and Go-around of Airliners Under Severe Weather Conditions Using Artificial Neural Networks", The 2017 International Workshop on Research, Education and Development on Unmanned Aerial Systems (RED-UAS), Linköping, Sweden, 2017.

5. H. Baomar and P. J. Bentley, "Training and Evaluating the Intelligent Autopilot System for Full Flight Cycles with Oman Air and Landing Autonomously Beyond the Current Limits and Capabilities Using Artificial Neural Networks". (to be submitted)

2016 International Conference on
Unmanned Aircraft Systems (ICUAS)
June 7-10, 2016. Arlington, VA USA

An Intelligent Autopilot System that Learns Piloting Skills from Human Pilots by Imitation

Haitham Baomar, Peter J. Bentley

Dept. of Computer Science, University College London, Gower Street, London, WClE 6BT, U.K.
Email: {h.baomar, p.bentley} @ cs.ucl.ac.uk

Abstract— An Intelligent Autopilot System (IAS) that can learn piloting skills by observing and imitating expert human pilots is proposed. IAS is a potential solution to the current problem of Automatic Flight Control Systems of being unable to handle flight uncertainties, and the need to construct control models manually. A robust Learning by Imitation approach is proposed which uses human pilots to demonstrate the task to be learned in a flight simulator while training datasets are captured from these demonstrations. The datasets are then used by Artificial Neural Networks to generate control models automatically. The control models imitate the skills of the human pilot when performing piloting tasks including handling flight uncertainties such as severe weather conditions. Experiments show that IAS performs learned take-off, climb, and slow ascent tasks with high accuracy even after being presented with limited examples, as measured by Mean Absolute Error and Mean Absolute Deviation. The results demonstrate that the IAS is capable of imitating low-level sub-cognitive skills such as rapid and continuous stabilization attempts in stormy weather conditions, and high-level strategic skills such as the sequence of sub-tasks necessary to pilot an aircraft starting from the stationary position on the runway, and ending with a steady cruise.

I. Introduction

Human pilots are trained to handle flight uncertainties or emergency situations such as severe weather conditions or system failure. In contrast, Automatic Flight Control Systems (AFCS/Autopilot) are highly limited, capable of performing minimal piloting tasks in non-emergency conditions. Strong turbulence, for example, can cause the autopilot to disengage or even attempt an undesired action which could jeopardise flight safety. The limitations of autopilots require constant monitoring of the system and the flight status by the flight crew to react quickly to any undesired situation or emergencies. On the other hand, trying to anticipate everything that could go wrong with a flight, and incorporating that into the set of rules or control models "hardcoded" in an AFCS is infeasible. There have been reports either discussing the limitations of current autopilots [1] [2] such as the inability to handle severe weather conditions, or blaming autopilots for a number of aviation catastrophes. One such example was Air France flight AF447 on June 1st 2009 where the aircraft entered a severe turbulence zone forcing it to climb steeply and stall. Shortly after that, the autopilot disengaged causing the aircraft to lose altitude dramatically. Unfortunately, it was too late for the flight crew to rectify the situation [3] [4].

This work aims to address this problem by creating an Intelligent Autopilot System (IAS) that can learn from human pilots by applying the Learning by Imitation concept with Artificial Neural Networks. By using this approach we aim to extend the capabilities of modern autopilots and enable them to autonomously adapt their piloting to suit multiple scenarios ranging from normal to emergency situations.

This paper is structured as follows: part (II) covers the autopilot problem in more details, and related work on utilizing Learning by Imitation in autonomous aviation. Part (III) explains the proposed Intelligent Autopilot System (IAS) prototype. Part (IV) describes the experiments, Part (V) describes the results by comparing the behaviour of the human pilot with the behaviour of the Intelligent Autopilot, and part (VI) provides an analysis of the results. Finally, we provide conclusions and future work.

II. Background

A. Automatic Flight Control Systems

Current operational autopilots fall under the domain of Control Theory. Classic and modern autopilots rely on controllers such as Proportional Integral Derivative controller (PID controller), and Finite-State automation [5]. Many recent research efforts focus on enhancing flight controllers, through the introduction of various methods such as a non-adaptive Backstepping approach [6], Dynamical Inversion flight control approach based on Artificial Neural Network Disturbance Observer to handle the dynamical inversion error factor [7], an L1 adaptive controller which is based on piecewise constant adaptive laws [8], a multi-layered hybrid linear/non-linear controller for biologically inspired Unmanned Aerial Vehicles [9], and a fault-tolerant control based on Gain-Scheduled PID [10]. However, manually designing and developing all the necessary controllers to handle the complete spectrum of flight scenarios and uncertainties ranging from normal to emergency situations might not be the ideal method due to feasibility limitations such as the difficulty in covering all possible eventualities.

978-1-4673-9333-1/16/$31.00 ©2016 IEEE

A. Artificial Neural Networks

Artificial Neural Networks (ANNs) are popular learning methods due to their ability to handle highly dynamic real-time large volumes of data. They are a highly interconnected system capable of processing data through their dynamic state response to external inputs. [11] Although Artificial Neural Networks are sometimes referred to as slow learners, as soon as the learning model is generated, ANNs are very fast classification and regression techniques that are suitable for applications running in dynamic and high-speed environments [12] such as high frequency trading [13], and electrical circuits management and analysis [14]. ANNs are also used in robotics applications due to their capability of handling large amounts of real-time noisy sensor data [15]. The latter resemble the Intelligent Autopilot System (IAS) which should be able to receive real-time flight status data from multiple sensors, process the data, and apply the appropriate command control actions given the current flight state.

B. Learning by Imitation for Autonomous Flight Control

Learning by Imitation can be applied to machines just as it can be applied to humans. Michie et al [16] demonstrated this concept with the attempt to balance a pole by a simulated system. Learning by Imitation is split into two main parts each with its own objectives: 1. learning a policy or a low-level task which could represent a direct mapping between states and relative actions, and 2. learning a reward function or a high-level task which could represent a specific goal to be achieved.

While Behavioural Cloning [17] has been applied to capture the high-level decision making process of a human pilot, Apprenticeship Leaning [18] has been applied to capture low-level highly dynamic tasks. Sammut [17] presented an early attempt to develop an autopilot that can learn by imitation. In [17], the Decision Tree induction program C4.5 was used to capture the set of rules or high-level tasks required to fly an aircraft in a flight simulator. The rules were transformed into a collection of If-Statements that govern the control commands sent by the autopilot. In [17], the main challenge was the need to capture low-level sub-cognitive actions that a human pilot performs rapidly.

Apprenticeship Learning using Inverse Reinforcement Learning, either by considering a Markov decision process [19], or by considering Gradient methods [20] focus on capturing low-level highly dynamic and rapid actions of a human demonstrator. These methods in general do not depend on receiving a Reward Function in advance, which is how classic Reinforcement Learning works, instead, the proposed approach attempts to find a reward function by observing how an expert human demonstrates the task to be learned by the system. Abbeel et al [21] applied Apprenticeship Learning to a dynamic control system performing acrobatic manoeuvres using a helicopter. Applying Apprenticeship Learning proved to be an efficient learning technique to capture the expert demonstrator's skills. In [21], multiple demonstrations by an expert were gathered. The goal was to consider observations as noisy attempts from the expert while performing the desired manoeuvre successfully. The main reported challenge was the difficulty to capture high-level dynamic models present in complex manoeuvres where successful performance of manoeuvres require a careful transition among multiple sub-actions.

Recently, and in the same context, Matsumoto et al [22] proposed a similar learning approach that depends on Learning from Demonstration (LFD) to capture the human pilot's skills and apply them in an autonomous Unmanned Aerial System (UAS) to achieve the same level of safety observed in civil aviation.

III. The Intelligent Autopilot System

The proposed Intelligent Autopilot System (IAS) in this paper can be viewed as an apprentice that observes the demonstration of a new task by the experienced teacher, and then performs the same task autonomously. In the IAS we bridge the gap between Behavioural Cloning and Apprenticeship Learning. A successful generalization of Learning by Imitation should take into consideration the capturing of low-level models and high-level models, which can be viewed as rapid and dynamic sub-actions that occur in fractions of a second, and actions governing the whole process and how it should be performed strategically. It is important to capture and imitate both levels in order to handle flight uncertainties successfully.

The IAS is made of the following components: a flight simulator, an interface, a database, and Artificial Neural Networks. The IAS implementation method has three steps: A. pilot data collection, B. training, and C. autonomous control. In each step, different IAS components are used. The following sections describe each step and the components used in turn.

A. Pilot Data Collection

Fig. 1 illustrates the IAS components used during the pilot data collection step.

1) Flight Simulator

Before the IAS can be trained or can take control, we must collect data from a pilot. This is performed using X-Plane which is an advanced flight simulator that has been used as the simulator of choice in many research papers such as [23] [24] [25].

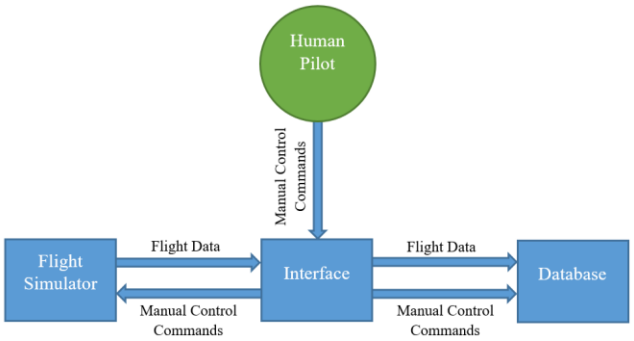

Fig. 1. Block diagram illustrating IAS components used during the pilot data collection step.

X-Plane is used by multiple organizations and industries such as NASA, Boeing, Cirrus, Cessna, Piper, Precession Flight Controls Incorporated, Japan Airlines, and the American Federal Aviation Administration.[1] X-Plane can communicate with external applications by sending and receiving flight status and control commands data over a network through User Datagram Protocol (UDP) packets. For this work, the simulator is set up to send and receive packets comprising desired data every 0.1 second.

2) IAS Interface

The IAS Interface is responsible for data flow between the flight simulator and the system in both directions. The Interface contains control command buttons that provide a simplified yet sufficient aircraft control interface including throttle, brakes, gear, elevator, aileron, and rudder, which can be used to perform basic tasks of piloting an aircraft such as take-off and landing in the simulator. It also displays flight data received from the simulator.

Data collection is started immediately before demonstration, then; the pilot uses the Interface to perform the piloting task to be learned. The Interface collects flight data from X-Plane over the network using UDP packets, and collects the pilot's actions while performing the task, which are also sent back to the simulator as manual control commands. The Interface organizes the collected flight data received from the simulator (inputs), and the pilot's actions (outputs) into vectors of inputs and outputs, which are sent to the database every 1 second.

3) Database

An SQL Server database stores all data captured from the pilot demonstrator and X-Plane, which are received from the Interface. The database contains tables designed to store: 1. continuous flight data as inputs, and 2. pilot's actions as outputs. These tables are then used as training datasets to train the Artificial Neural Networks of IAS.

A. Training

1) Artificial Neural Networks

After the human pilot data collection step is completed, Artificial Neural Networks are used to generate learning models from the captured datasets through offline training. Fig. 2 illustrates the training step.

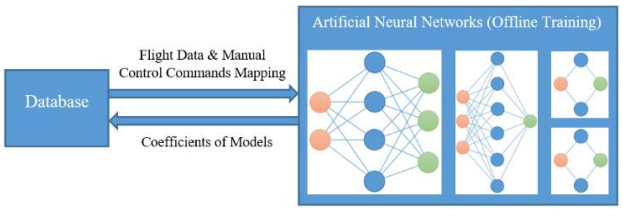

Fig. 2. Block diagram illustrating IAS components used during training.

Four feedforward Artificial Neural Networks represent the core of IAS. Each ANN is designed and trained to handle specific controls. The inputs of ANN 1 are: speed and altitude values, and the outputs are: throttle, gear, and brakes values. The inputs of ANN 2 are: speed, altitude, and pitch values, and the output is: elevator value. The input of ANN 3 is: roll value, and the output is: aileron value. The input of ANN 4 is: heading value, and the output is: rudder value.

The topologies of the four ANNs are illustrated in Fig. 3. The method for choosing ANN topologies in this work is based on a rule-of-thumb [26] which indicates that problems requiring more than one hidden layer are rarely encountered. This rule follows an approach that tries to avoid under-fitting caused by too few neurons in the hidden layer, or over-fitting caused by too many neurons, by having the number of hidden neurons less than or equal to twice the size of the input layer.

During training, the datasets are normalized, and retrieved from the database. Then, the datasets are fed to the ANNs. Next, Sigmoid (1) [26] and Hyperbolic Tangent (Tanh) (2) [26] functions are applied for the neuron activation step, where $f(x)$ is the activation value for each neuron, and x is the relevant target value:

$$f(x) = \frac{1}{1+e^{-x}} \tag{1}$$

$$f(x) = \frac{e^{2x}-1}{e^{2x}+1} \tag{2}$$

The Sigmoid activation function (1) is used by ANN 1 since all input and output values are positive, while Tanh is used by ANN 2, 3, and 4 since the datasets contain few negative values: pitch (ANN 2), rudder (ANN 3), roll, and aileron (ANN 4).

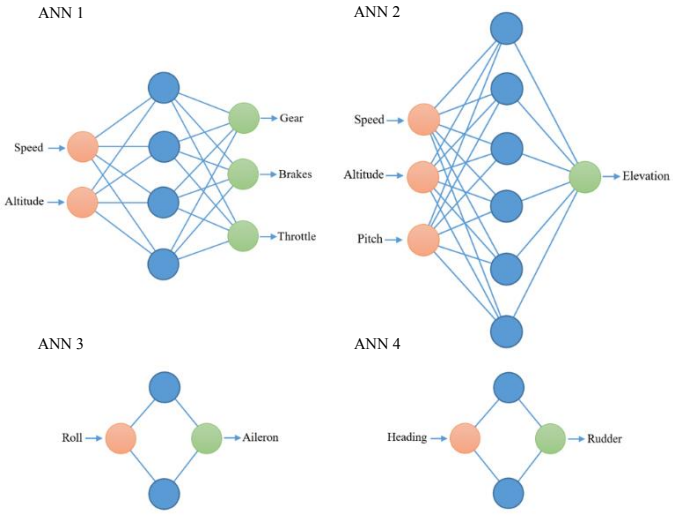

Fig. 3. Topology of ANN 1 trained to handle throttle, gear and brakes (top left), topology of ANN 2 trained to handle elevator control (top right), topology of ANN 3 trained to handle aileron control (bottom left), and topology of ANN 4 trained to handle rudder control (bottom right).

[1] "X-Plane 10 Global
http://www.x-plane.com

Next, Backpropagation is applied. Based on the activation function, (3) [27], or (4) [27] are applied to calculate the error signal (δ) for each neuron where t_n is the desired target value and a_n is the actual activation value:

$$\delta_n = (t_n - a_n)a_n(1 - a_n) \quad (3)$$

$$\delta_n = (t_n - a_n)(1 - a_n)(1 + a_n) \quad (4)$$

Finally, coefficients of models (weights and biases) are updated using (5) [28] where $\delta w_{i,j}$ is the change in the weight between nodes j and k.

$$w_{i,j} = w_{i,j} + \delta w_{i,j} \quad (5)$$

When training is completed, the learning models are generated, and the free parameters or coefficients represented by weights and biases of the models are stored in the database.

B. Autonomous Control

Once trained, the IAS can now be used for autonomous control. Fig. 4 illustrates the components used during the autonomous control step.

1) IAS Interface

Here, the Interface retrieves the coefficients of the models from the database for each trained ANN, and receives flight data from the flight simulator every 0.1 second. The Interface organizes the coefficients into sets of weights and biases, and organizes data received from the simulator into sets of inputs for each ANN. The relevant coefficients, and flight data input sets are then fed to the ANNs of the IAS to produce outputs. The outputs of the ANNs are sent to the Interface which sends them to the flight simulator as autonomous control commands using UDP packets every 0.1 second.

2) Artificial Neural Networks

The relevant set of flight data inputs received through the Interface is used by each ANN input neurons along with the relevant coefficients to predict and output the appropriate control commands given the flight status by applying (1) and (2). The values of the output layer are continuously sent to the Interface which sends them to the flight simulator as autonomous control commands.

IV. Experiments

In order to assess the effectiveness of the proposed approach, the Intelligent Autopilot System was tested in two experiments: A. autonomous flying under calm weather, and B. autonomous flying under stormy weather. Each experiment is composed of 10 attempts by the IAS to fly autonomously under the given weather conditions.

At this point of our work, the scope only covers the ability of the proposed system to imitate the behaviour of the human pilot while performing basic piloting tasks. We do not focus on maintaining a strict velocity and attitude during the flight, which is among the tasks to be taught to the IAS in our next work.

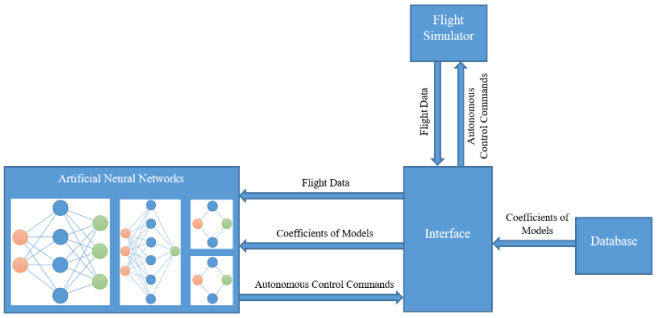

Fig. 4. Block diagram illustrating IAS components used during autonomous control.

Fig. 5 illustrates a break-down of the piloting task to be learned, to four sub-tasks based on time. Each attempt lasted for 182 seconds. The human pilot who provided the demonstrations is the first Author. The simulated aircraft used for the experiments is Cirrus Vision SF50. Since it is a light single- engine jet aircraft, it is relatively simpler to control, and responds quickly to pilot input. The experiments are as follows:

A. Autonomous Flying under Calm Weather

The purpose of this experiment is to assess the behaviour of the IAS compared to the behaviour of the human pilot under calm weather conditions.

1) Data Collection

In this experiment, the human pilot used the IAS Interface to perform the following in the flight simulator: take off, gaining altitude, and maintaining a slower climb rate with a fixed vector, under calm weather with null readings of wind gusts and turbulence. The performed tasks lasted for 182 seconds as Fig. 5 shows. While the pilot performed the demonstration, the Interface collected speed and altitude as simulator inputs, throttle, gear, and brakes as pilot outputs, and elevator control data (speed, altitude, pitch as simulator inputs, and elevator as pilot output). The Interface stored collected data as two training datasets in the database. Only one demonstration was presented to the system under calm weather.

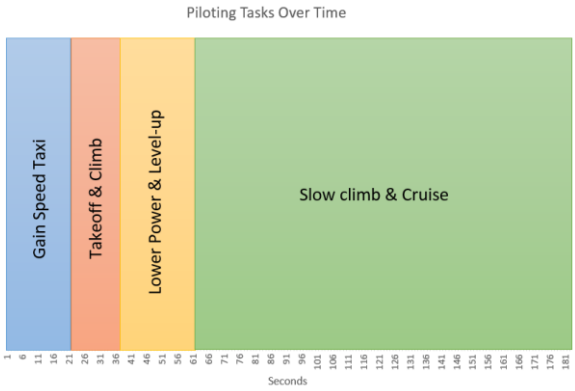

Fig. 5. Piloting tasks over time.

An additional data collection process was initiated to capture and compare the aircraft's Automatic Flight Control (AFC)/Autopilot performance with the IAS under calm weather conditions. Due to the AFC's inability to take-off, it was engaged at an altitude of 1600 ftmsl. The AFC was set to climb to an altitude of 6000 ftmsl at a rate of 1500 ftmsl per minute.

2) Training

For this experiment, ANN 1 (throttle, gear, and brakes control), and ANN 2 (elevator control) were trained until low Mean Squared Error (MSE) values were achieved (below 0.1).

3) Control

After training the ANNs on the relevant training datasets, the aircraft was reset to the runway in the flight simulator, calm weather conditions were chosen, and the IAS was engaged. ANN 1 (throttle, gear, and brakes control), and ANN 2 (elevator control) operate simultaneously to control the aircraft autonomously. Through the Interface, they receive: 1. continuous flight data from the flight simulator as inputs, and 2. coefficients of models from the database (calm weather throttle, gear, brakes, and elevator control models) to predict and output command controls that are sent to the flight simulator. This process allows the IAS to perform learned tasks: take off, gaining altitude, and maintaining a slow climb rate with a fixed vector autonomously. This was repeated 10 times to assess performance consistency.

B. Autonomous Flying under Stormy Weather

The purpose of this experiment is to assess the behaviour of the Intelligent Autopilot compared to the behaviour of the human pilot under stormy weather conditions.

1) Data Collection

In this experiment, the human pilot used the IAS Interface to perform the following in the flight simulator: take off, gaining altitude, and maintaining a slower climb rate with a fixed vector, under stormy weather. The weather conditions included: wind gusts reaching up to 33 knots, wind directions flowing from all directions (0 to 360 degrees clockwise deviation from north), local turbulence up to 0.19, and rain and hail perception up to 68 mm.

While the pilot performed the demonstration, the Interface collected rudder control and aileron control data only, and stored them as two training datasets in the database.

Two demonstrations were required to capture the skill needed to keep the light aircraft on the runway during strong crosswinds using rudders, and only one demonstration of roll stabilization using ailerons was presented to the system. To test the system's ability to generalize well in unseen conditions, no new throttle, gear, brakes, and elevator control data was collected under stormy weather conditions; instead, the data collected for these controls in Experiment 1 were reused. This aims to test the ability of the models generated under calm weather conditions to generalize in the unseen stormy weather conditions.

During taxi speed gain on the runway, the human pilot attempted multiple heading corrections using the rudder to stay on the runway while strong crosswinds pushed the aircraft right and left. After take-off, the human pilot constantly corrected the roll deviation by controlling the ailerons. The collected data was cleaned by removal of outliers. These were caused by noise represented by values that fall within transition phases (e.g. aggressive correction of heading), human error, or signal error.

An additional data collection process was initiated to capture and compare the aircraft's AFC performance with the IAS under stormy weather conditions with the same settings used in experiment 1. It should be mentioned that the AFC disengaged itself multiple times while flying through the storm which made it difficult to capture a complete demonstration, especially when the strong winds affected the aircraft's stability and caused it to stall.

2) Training

For this experiment, ANN 3 (rudder control), and ANN 4 (aileron control) were trained until low Mean Squared Error (MSE) values were achieved (below 0.1).

3) Control

After training the ANNs on the relevant training datasets, the aircraft was reset to the runway in the flight simulator, stormy weather conditions were chosen, and the IAS was engaged. ANN 1 (throttle, gear, and brakes control), ANN 2 (elevator control), ANN 3 (aileron control), and ANN 4 (rudder control) operate simultaneously to control the aircraft autonomously. Through the Interface, they receive: 1. continuous flight data from the flight simulator as inputs, and 2. coefficients of models from the database (calm weather throttle, gear, brakes, and elevator control models, and stormy weather rudder and aileron control models) to predict and output command controls that are sent to the flight simulator. This process allows the IAS to perform learned tasks: take off, gaining altitude, and maintaining a slow climb rate with a fixed vector autonomously, while continuously correcting the aircraft's heading and roll. This was repeated 10 times to assess performance consistency.

V. RESULTS

The following section describes the results of the conducted tests. The 10 attempts by IAS to fly autonomously in each experiment (calm and stormy weather) were averaged and compared with the performance of the human pilot, and the aircraft's AFC using Mean Absolute Error (MAE), Mean Absolute Deviation (MAD), and illustrated by Behaviour Charts.

A. Experiment 1 (Calm Weather Condition)

Two models were generated with the following MSE values as table I shows.

Table II lists the accuracy assessment results by comparing the behaviour of IAS with the behaviour of the human pilot in the calm weather experiment.

Table III lists the accuracy assessment results by comparing the behaviour of IAS with the behaviour of the aircraft's AFC in the calm weather experiment.

Fig. 6, 7, and 8 illustrate the Intelligent Autopilot's control commands compared to the human pilot. Fig. 9 and 10 illustrate altitude and speed over time comparisons between the human pilot, the Intelligent Autopilot System, and the aircraft's AFC.

B. Experiment 2 (Stormy Weather Condition)

Two models were generated with MSE values as table IV shows.

Table V lists the accuracy assessment results by comparing the behaviour of IAS with the behaviour of the human pilot in the stormy weather experiment.

Table VI lists the accuracy assessment results by comparing the behaviour of IAS with the behaviour of the aircraft's AFC in the stormy weather experiment.

TABLE I
MSE VALUES OF THE MODELS GENERATED UNDER CALM WEATHER

ANN Model	MSE	Weather Condition
Throttle, gear, and brakes model	0.03	Calm
Elevation control model	0.09	Calm

TABLE II
IAS ACCURACY ASSESSMENT RESULTS COMPARED WITH THE HUMAN PILOT. ACCURACY IS MEASURED USING MEAN ABSOLUTE ERROR (MAE) AND MEAN ABSOLUTE DEVIATION (MAD) – CALM WEATHER.

Command Control/Flight Status	Weather Condition	(MAE)	(MAD) - Human	(MAD) - AI
Throttle Command	Calm	0.016	0.1	0.1
Gear Command	Calm	0.016	0.3	0.3
Elevation Command	Calm	0.007	0.022	0.022
Altitude	Calm	91.694	1204.779	1251.816
Speed	Calm	6.148	23.031	24.925

TABLE III
IAS COMPARED WITH THE AIRCRAFT'S AFC. ACCURACY IS MEASURED USING MAE AND MAD – CALM WEATHER.

Command Control/Flight Status	Weather Condition	(MAE)	(MAD) - AFC	(MAD) - AI
Altitude	Calm	200.096	814.382	833.118
Speed	Calm	17.308	1.098	4.785

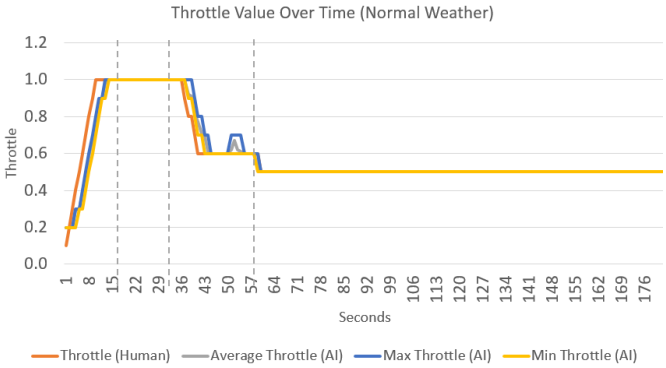

Fig. 6. (Exp. 1) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum throttle commands over time during the four phases –separated by dotted lines- as illustrated in Fig. 4.

Fig. 11, 12, and 13 illustrate the IAS control commands compared to the human pilot in the stormy weather experiment. Fig. 14 and 15 illustrate altitude and speed over time comparisons between the human pilot, the IAS, and the aircraft's AFC in the stormy weather experiment.

Fig. 16 generated from sample heading/rudder data, illustrates a comparison between the human pilot and IAS heading correction attempts using the rudder. Fig. 17 generated from sample roll/aileron data illustrates the comparison between the human pilot and the IAS roll correction attempts using the ailerons.

TABLE IV
MSE VALUES OF THE MODELS GENERATED UNDER STORMY WEATHER

ANN Model	MSE	Weather Condition
Rudder control model	0.009	Storm
Aileron control model	0.07	Storm

TABLE V
IAS ACCURACY ASSESSMENT RESULTS COMPARED WITH THE HUMAN PILOT. ACCURACY IS MEASURED USING MEAN ABSOLUTE ERROR (MAE) AND MEAN ABSOLUTE DEVIATION (MAD) – STORMY WEATHER.

Command Control/Flight Status	Weather Condition	(MAE)	(MAD) - Human	(MAD) - AI
Throttle Command	Stormy	0.080	0.146	0.122
Gear Command	Stormy	0.063	0.349	0.355
Elevation Command	Stormy	0.017	0.034	0.030
Altitude	Stormy	350.963	1297.031	1153.647
Speed	Stormy	15.757	36.520	41.347
Rudder Command	Stormy	0.041	0.093	0.133
Aileron Command	Stormy	0.006	0.086	0.080

TABLE VI
IAS COMPARED WITH THE AIRCRAFT'S AFC. ACCURACY IS MEASURED USING MEAN ABSOLUTE ERROR (MAE) AND MEAN ABSOLUTE DEVIATION (MAD) – STORMY WEATHER.

Command Control/Flight Status	Weather Condition	(MAE)	(MAD) - AFC	(MAD) - AI
Altitude	Stormy	316.952	804.572	783.771
Speed	Stormy	12.322	3.349	4.212

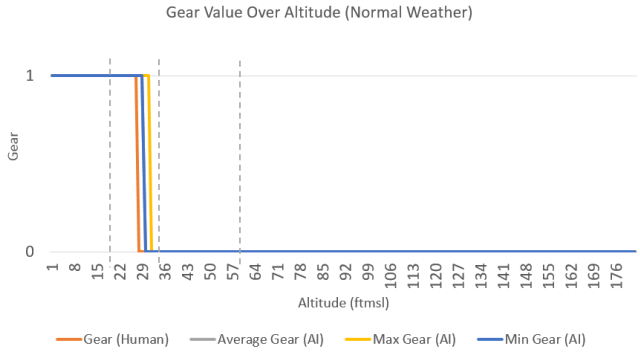

Fig. 7. (Exp. 1) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum gear commands over time.

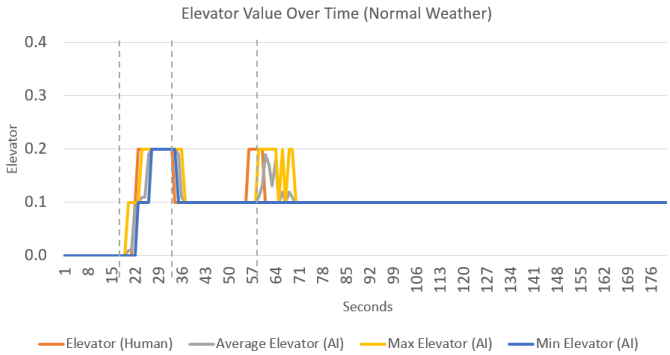

Fig. 8. (Exp. 1) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum elevator commands over time.

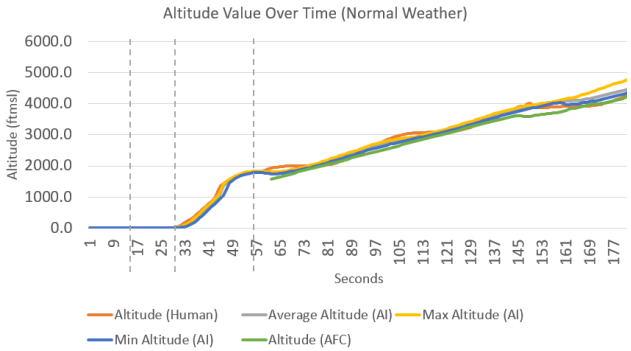

Fig. 9. (Exp. 1) A comparison between the human pilot, the aircraft's AFC/Autopilot, and the Intelligent Autopilot's average, maximum, and minimum altitude over time.

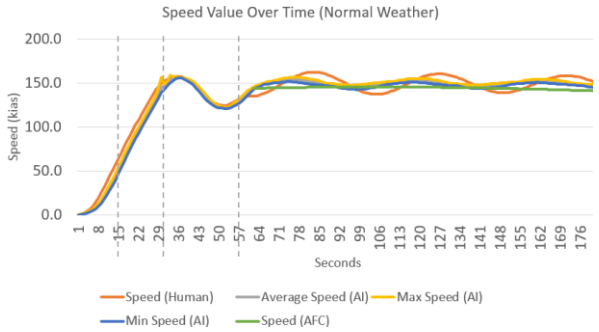

Fig. 10. (Exp. 1) A comparison between the human pilot, the aircraft's AFC/Autopilot, and the Intelligent Autopilot's average, maximum, and minimum speed over time.

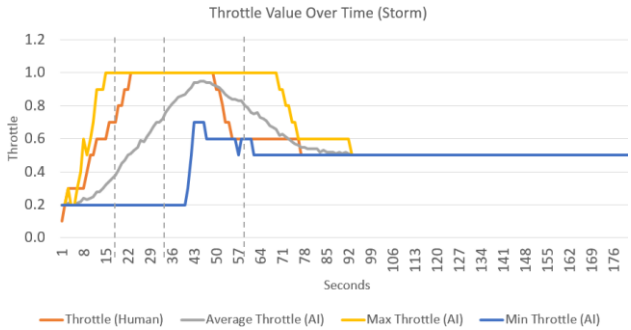

Fig. 11. (Exp. 2) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum throttle commands over time during the four phases –separated by dotted lines- as illustrated in Fig. 4.

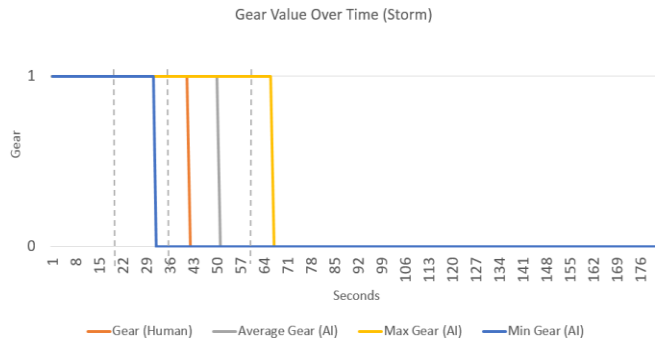

Fig. 12. (Exp. 2) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum gear commands over time.

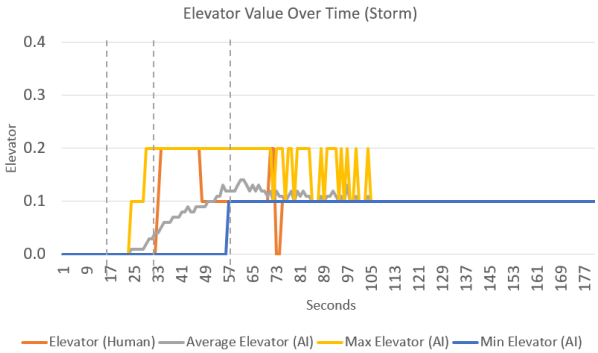

Fig. 13. (Exp. 2) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum elevator commands over time.

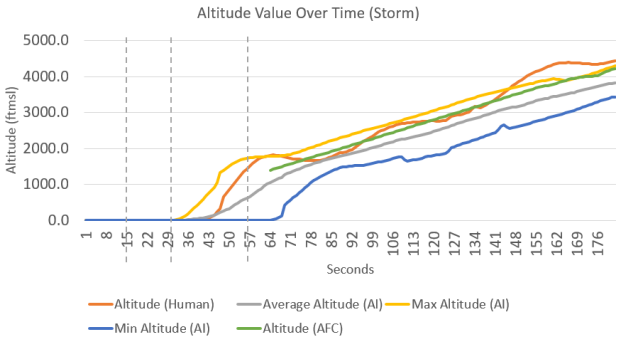

Fig. 14. (Exp. 2) A comparison between the human pilot, the aircraft's AFC/Autopilot, and the Intelligent Autopilot's average, maximum, and minimum altitude over time.

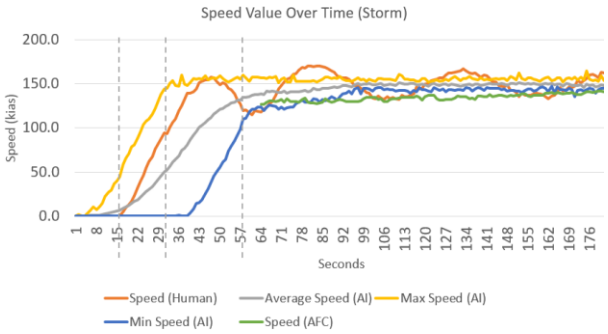

Fig. 15. (Exp. 2) A comparison between the human pilot, the aircraft's AFC/Autopilot, and the Intelligent Autopilot's average, maximum, and minimum speed over time.

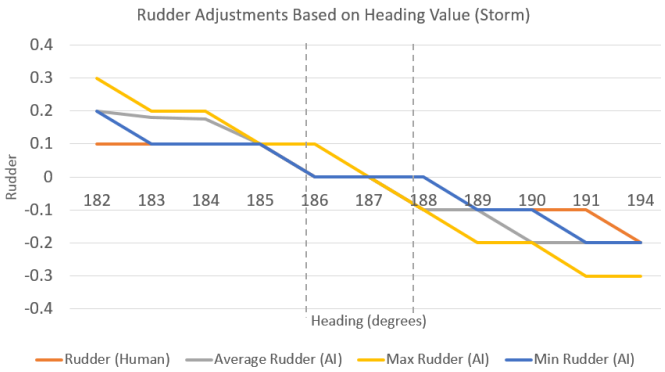

Fig. 16. (Exp. 2) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum heading correction attempts. The middle part between the two dotted lines is the area where no corrections are required (based on a heading of 187 degrees). The right part illustrates a deviation in heading towards the right, while the left part illustrates a deviation in heading towards the left.

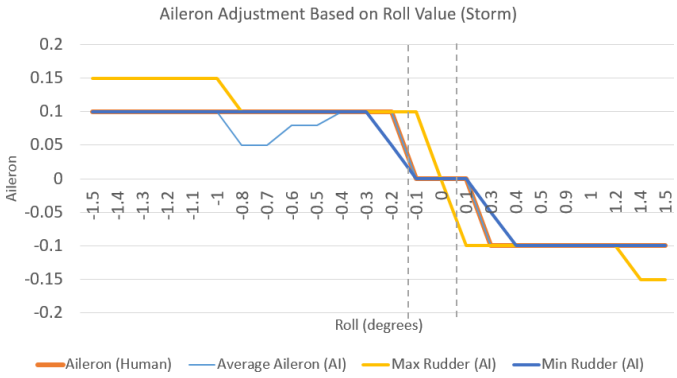

Fig. 17. (Exp. 2) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum roll correction attempts. The middle part between the two dotted lines is the area where no corrections are required. The right part illustrates a deviation in roll towards the right, while the left part illustrates a deviation in roll towards the left.

VI. Analysis

As can be seen in Figs 6 to 10, experiment 1 (calm weather condition) presented very desirable results. The IAS was capable of imitating the human pilot's actions and behaviour with remarkable accuracy, and strong consistency.

As can be seen in Figs 11 to 17, experiment 2 (stormy weather condition) showed the ability of IAS to imitate rapid stabilization actions, and generalize well in unseen conditions. The system used the calm weather models to fly in stormy conditions gracefully.

The system was able to imitate multiple human pilot's skills and behaviour after being presented with very limited examples (1 example for throttle, gear, and brakes, 1 example for elevator control, 1 example for aileron control, and 2 examples for rudder control). The results show that the Intelligent Autopilot continued to stabilize the aircraft in difficult weather condition, while the AFC of the simulated aircraft disengaged itself multiple times.

It should be mentioned that the human pilot found it difficult to regulate the speed of the aircraft as shown by the oscillations, but despite receiving this data as training, the IAS learned to fly smoothly - indeed smoother than the human pilot as can be seen in Figs 10 and 15.

The complete learning process starting from the demonstration of the specific task by the human pilot, and ending with the automatic generation of the learning model takes less than 20 minutes.

Informal trials were also performed with the IAS in which the aircraft was put into a variety of situations that it had not been trained to handle (e.g., a stall, inversion, etc.). In all cases the IAS was able to stabilize the aircraft safely on its own.

VII. Conclusion & Future Work

The aviation industry is currently working on solutions which should lead to decreasing the dependence on crew members. The reason behind this is to lower workload, human error, and stress faced by crew members, by developing autopilots capable of handling multiple scenarios without human intervention. In this work, a robust approach is proposed to "teach" autopilots how to handle uncertainties and emergencies with minimum effort by exploiting Learning by Imitation.

The experiments were strong indicators towards the ability of Supervised Learning with Artificial Neural Networks to capture low-level piloting tasks such as the rapid corrections of heading and roll deviations in stormy weather conditions. The experiments showed the ability of the IAS to capture high-level tasks and rules such as applying elevator only after a certain speed is achieved, retracting gear at a certain altitude, and also levelling the aircraft and shifting from the climb to the smooth ascent and cruise phase at a certain altitude.

Future effort will focus on a further and extended break-down of the piloting tasks. More Artificial Neural Networks should be added to the Intelligent Autopilot System to enhance performance and accuracy, and to cover a wider spectrum of sub-tasks. The learning by Imitation approach in this context should be extended to cover new tasks and scenarios that have not been presented yet to the system. The new tasks and scenarios could cover emergency situations such as handling urgent take-off abortion, engine fire, etc. We anticipate that future Autopilot systems which make of methods proposed here could improve safety and save lives.

REFERENCES

[1] Baumbach, J., Marais, A. and Gonçalves, D. (2015). Losing the Boxes: Fragmentation as a Source of System Complexity. Proc. of the 11th SA INCOSE Conference.

[2] Sherry, L., Mauro, R. (2014). Controlled Flight into Stall (CFIS): Functional complexity failures and automation surprises. Integrated Communications, Navigation and Surveillance Conference (ICNS), vol., no., pp.D1-1-D1-11.

[3] Salmon, P., Walker, G. and Stanton, N. (2015). Pilot error versus sociotechnical systems failure: a distributed situation awareness analysis of Air France 447. Theoretical Issues in Ergonomics Science, 17(1), pp.64-79.

[4] Final report on the accident on 1st June 2009 to the Airbus A330-203 registered F-GZCP operated by Air France Flight AF447 Rio de Janeiro – Paris by Bureau d'Enquêtes et d'Analyses pour la sécurité de l'aviation civile.

[5] Nelson, Robert C. Flight stability and automatic control. Vol. 2. WCB/McGraw Hill, 1998.

[6] Sartori, D., Quagliotti, F., Rutherford, M. and Valavanis, K. (2014). Implementation and Testing of a Backstepping Controller Autopilot for Fixed-wing UAVs. Journal of Intelligent & Robotic Systems, 76(3-4), pp.505-525.

[7] Chen, M., Jiang, C. and Wu, Q. (2011). Disturbance-Observer-Based Robust Flight Control for Hypersonic Vehicles Using Neural Networks. adv sci lett, 4(4), pp.1771-1775.

[8] Capello, E., Guglieri, G., Quagliotti, F. and Sartori, D. (2012). Design and Validation of an ${\mathcal{L}}_{1}$ Adaptive Controller for Mini-UAV Autopilot. Journal of Intelligent & Robotic Systems, 69(1-4), pp.109-118.

[9] Ratti, J. and Vachtsevanos, G. (2011). Inventing a Biologically Inspired, Energy Efficient Micro Aerial Vehicle. Journal of Intelligent & Robotic Systems, 65(1-4), pp.437-455.

[10] Sadeghzadeh, I. and Zhang Y. (2013). Actuator fault-tolerant control based on Gain-Scheduled PID with application to fixed-wing Unmanned Aerial Vehicle. Control and Fault-Tolerant Systems (SysTol), pp.342-346.

[11] Hecht-Nielsen, Robert. Neurocomputing. Reading, Mass.: Addison-Wesley Pub. Co., 1990. Print.

[12] Young-Keun Park, Gyungho Lee, "Applications of neural networks in high-speed communication networks," in Communications Magazine, IEEE , vol.33, no.10, pp.68-74, Oct 1995 doi: 10.1109/35.466222

[13] Tino, P., Schittenkopf, C. and Dorffner, G. (2001). Financial volatility trading using recurrent neural networks. IEEE Trans. Neural Netw., 12(4), pp.865-874.

[14] Burr, G., Shelby, R., Sidler, S., di Nolfo, C., Jang, J., Boybat, I., Shenoy, R., Narayanan, P., Virwani, K., Giacometti, E., Kurdi, B. and Hwang, H. (2015). Experimental Demonstration and Tolerancing of a Large-Scale Neural Network (165 000 Synapses) Using Phase-Change Memory as the Synaptic Weight Element. IEEE Trans. Electron Devices, 62(11), pp.3498-3507.

[15] Miller III, W. Thomas. "Real-time application of neural networks for sensor-based control of robots with vision." Systems, Man and Cybernetics, IEEE Transactions on 19.4 (1989): 825-831.

[16] Michie, D., Bain, M., and Hayes-Michie, J.E. (1990). Cognitive Models from Subcognitive Skills. Knowledge-base Systems in Industrial Control.

[17] Sammut, Claude. (1992). Automatically constructing control systems by observing human behaviour. Proc. of the Internat. Workshop on Inductive Logic Programming.

[18] Ng, A. Y., & Russell, S. J. (2000). Algorithms for inverse reinforcement learning. In Icml (pp. 663-670).

[19] Abbeel, P., & Ng, A. Y. (2004). Apprenticeship learning via inverse reinforcement learning. In proc of the twenty-first international conference on Machine learning (p. 1).

[20] Neu, G., & Szepesvári, C. (2012). Apprenticeship learning using inverse reinforcement learning and gradient methods. arXiv preprint arXiv:1206.5264.

[21] Abbeel, P., Coates, A., & Ng, A. (2010). Autonomous Helicopter Aerobatics through Apprenticeship Learning. International Journal of Robotics Research vol. 29, iss. 13, pp. 1608

[22] Matsumoto, T., Vismari, L., Camargo, J. (2014). A method to implement and to evaluate a learning-based Piloting Autonomous System for UAS. International Conference on Unmanned Aircraft Systems (ICUAS), vol., no., pp.195-199.

[23] Wei, F., Amaya-Bower, L., Gates, A., Rose, D. and Vasko, T. (2016). The Full-Scale Helicopter Flight Simulator Design and Fabrication at CCSU. 57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference.

[24] Jirgl, M., Boril, J., Jalovecky, R. (2015). The identification possibilities of the measured parameters of an aircraft model and pilot behavior model on the flight simulator. International Conference on Military Technologies (ICMT), vol., no., pp.1-5.

[25] Kaviyarasu, A. and Senthil Kumar, K. (2014). Simulation of Flapping-wing Unmanned Aerial Vehicle using X-plane and Matlab/Simulink. Defence Science Journal, 64(4), pp.327-331.

[26] Heaton, J. (2005). Introduction to neural networks with Java. St. Louis: Heaton Research.

[27] McClelland, J. (2015). Explorations in Parallel Distributed Processing: A Handbook of Models, Programs, and Exercises (2nd ed.). Stanford.

[28] Tveter, D. (1995). Chapter 2, The Backprop Algorithm.

An Intelligent Autopilot System that Learns Flight Emergency Procedures by Imitating Human Pilots

Haitham Baomar, Peter J. Bentley

Dept. of Computer Science, University College London, Gower Street, London, WClE 6BT, U.K.
Email: {h.baomar, p.bentley} @ cs.ucl.ac.uk

Abstract— We propose an extension to the capabilities of the Intelligent Autopilot System (IAS) from our previous work, to be able to learn handling emergencies by observing and imitating human pilots. The IAS is a potential solution to the current problem of Automatic Flight Control Systems of being unable to handle flight uncertainties, and the need to construct control models manually. A robust Learning by Imitation approach is proposed which uses human pilots to demonstrate the task to be learned in a flight simulator while training datasets are captured from these demonstrations. The datasets are then used by Artificial Neural Networks to generate control models automatically. The control models imitate the skills of the human pilot when handling flight emergencies including engine(s) failure or fire, Rejected Take Off (RTO), and emergency landing, while a flight manager program decides which ANNs to be fired given the current condition. Experiments show that, even after being presented with limited examples, the IAS is able to handle such flight emergencies with high accuracy.

I. INTRODUCTION

Human pilots are trained to handle flight uncertainties or emergency situations such as severe weather conditions or system failure. For example, pilots are exposed to scenarios of forced or emergency landing which is performed by executing standard emergency procedures. Usually, the main phase of an emergency landing is known as gliding which is the reliance on the aerodynamics of the aircraft to glide for a given distance while altitude is lost gradually. This happens when the aircraft has lost thrust due to full engine failure in relatively high altitudes.

In contrast, Automatic Flight Control Systems (AFCS/Autopilot) are highly limited, capable of performing minimal piloting tasks in non-emergency conditions. Autopilots are not capable of handling flight emergencies such as engine failure, fire, performing a Rejected Take Off, or a forced (emergency) landing. The limitations of autopilots require constant monitoring of the system and the flight status by the flight crew to react quickly to any undesired situation or emergencies. The reason for such limitations of conventional AFCS is that it is not feasible to anticipate everything that could go wrong with a flight, and incorporate all of that into the set of rules or control models "hardcoded" in an AFCS.

This work aims to address this problem by expanding the capabilities of the Intelligent Autopilot System (IAS) [1] to be able to learn flight emergency procedures from human pilots by applying the Learning by Imitation concept with Artificial Neural Networks. By using this approach, we aim to extend the capabilities of modern autopilots and enable them to autonomously adapt their piloting to suit multiple scenarios ranging from normal to emergency situations.

This paper is structured as follows: part (II) reviews related literature on fault/failure tolerant systems, and the application of multiple ANNs or Artificial Neural Circuits. Part (III) explains the Intelligent Autopilot System (IAS). Part (IV) describes the experiments, Part (V) describes the results by comparing the behaviour of the human pilot with the behaviour of the Intelligent Autopilot System, and part (VI) provides an analysis of the results. Finally, we provide conclusions and future work.

II. BACKGROUND

A review of the Autopilot problem, Artificial Neural Networks, and Learning by Imitation for Autonomous Flight Control is presented in our previous work [1].

A. *Fault/Failure Tolerant Systems for Flight Control*

Current operational autopilots fall under the domain of Control Theory. Classic and modern autopilots rely on controllers such as the Proportional Integral Derivative (PID) controller, and Finite-State automation [2]. Many recent research efforts focus on enhancing flight controllers by adding fault/failure tolerant capabilities. With respect to flight control systems, a fault is "an unpermitted deviation of at least one characteristic property of the system from the acceptable, usual, standard condition." [3], while failure is "a permanent interruption of a system's ability to perform a required function under specified operating conditions." [3].

To handle faults and failures, recent research efforts have been focusing on designing Fault Detection and Diagnosis (FDD) systems that can either stream information to ground crew members especially in the case of UAVs, or feed fault tolerant systems that are capable of handling system faults. The first type of such systems are known as the Passive Fault Tolerant Controllers which can handle moderate faults such as parameters deviations by using a robust feedback controller. However, if the faults are beyond the capabilities of such controllers, another type of fault tolerant systems becomes a necessity. This type is known as an Active Fault Tolerant control

system which includes a separate FDD system that adds an extended and enhanced level of fault tolerance capabilities [4].

In case of emergency situations, mainly engine failure, engine fire, flight instruments failure, or control surface damage or failure, continuing to fly becomes either impossible or can poses a serious threat to the safety of the flight. In such circumstances, a forced or emergency landing on a suitable surface such as a flat field becomes a must especially if it is not possible to return safely to the runway [5]. In [6], an emergency landing controller is proposed for an Unmanned Aerial Vehicle by segmenting the emergency landing period into four sub-levels known as slipping guiding, straight line down, exponential pulling up, and shallow sliding. Each level uses different control strategies aimed at insuring the safe execution of the complete emergency landing. For example, during the exponential pulling up level, the system maintains a certain pitch without causing the UAV to stall. Using a simulator, the proposed approach showed its ability to handle emergency landing [6].

B. *Multiple ANNs or Artificial Neural Circuits*

The problem of coordinating multiple sensor-motor architectures found in complex robotic systems is challenging. This is due to the simultaneous and dynamic operation of these motors while insuring rapid and adaptive behaviour, and due to the need to properly handle the fusion of data from disparate sources. In nature, animals manage this problem by the large number of neural circuits in the animals' brains. For example, neural circuits which are responsible for motion are connected to the muscles (motor systems), and operate simultaneously and dynamically while handling changes in the environment [7]. This has inspired the field of complex robotics to develop multiple neural-based controllers and integrate them together to tackle larger problems such as long-endurance locomotion under uncertainties. For example, the problem of coordinating multiple sensor-motor architectures is addressed in the context of walking by developing a neural circuit which generates multiple gaits adaptively, and coordinates the process of walking with different behavioural-based processes in a hexapod robot. The results showed the ability of the biology-inspired system to detect and stabilize multiple instability scenarios, and to determine what needs to be controlled at each moment which allows the system to handle changes in the environment [7].

Multiple Artificial Neural Networks were applied to the problem of detecting roads visually. In [8], different inputs are fed into multiple ANNs to handle multiple segments of the image. The proposed approach allows the system to detect and classify multiple factors of the environment ahead which leads to an enhanced performance compared to other computer-vision solutions [8]. In [9], Multiple ANNs were applied to tackle the limitations problem of traffic light control systems that are based on conventional mathematical methods. In simulation, the results showed that the approach of using multiple ANNs to address this problem presented an improvement in performance compared to other methods [9]. Another proposed system inspired by biology; is presented in [10] which is designed to handle the challenging problem of gesture recognition. The system shares similarities with the human visual system by developing multiple spiking ANNs. The outputs of the spiking ANNs are used to generate a fusion of multiple data from different segments of the gesture. The results proved the system's ability to handle dynamic visual recognition with the presence of complex backgrounds [10].

The approach of segmenting or breaking down the problem, and using multiple ANNs to handle multiple segment shows the potential to enhance the properties of ANNs as explained in [11]. A large ANN is split into parallel circuits that resemble the circuits of the human retina. During training, the Backpropagation algorithm runs in each circuit separately. This approach does not only decrease training time, but it also enhances generalization [11].

III. The Intelligent Autopilot System

The proposed Intelligent Autopilot System (IAS) in this paper can be viewed as an apprentice that observes the demonstration of a new task by the experienced teacher, and then performs the same task autonomously. A successful generalization of Learning by Imitation should take into consideration the capturing of low-level models and high-level models, which can be viewed as rapid and dynamic sub-actions that occur in fractions of a second, and actions governing the whole process and how it should be performed strategically. It is important to capture and imitate both levels in order to handle flight uncertainties successfully.

The IAS is made of the following components: a flight simulator, an interface, a database, a flight manager program, and Artificial Neural Networks. The IAS implementation method has three steps: A. Pilot Data Collection, B. Training, and C. Autonomous Control. In each step, different IAS components are used. The following sections describe each step and the components used in turn.

A. *Pilot Data Collection*

Fig. 1 illustrates the IAS components used during the pilot data collection step.

1) *Flight Simulator*

Before the IAS can be trained or can take control, we must collect data from a pilot. This is performed using X-Plane which is an advanced flight simulator that has been used as the simulator of choice in many research papers such as [12] [13] [14].

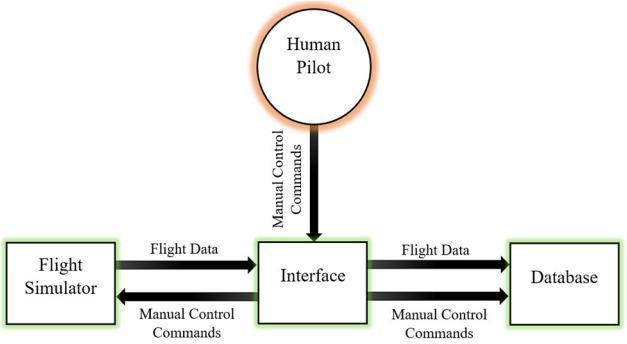

Fig. 1. Block diagram illustrating the IAS components used during the pilot data collection step.

X-Plane is used by multiple organizations and industries such as NASA, Boeing, Cirrus, Cessna, Piper, Precession Flight Controls Incorporated, Japan Airlines, and the American Federal Aviation Administration.[1] X-Plane can communicate with external applications by sending and receiving flight status and control commands data over a network through User Datagram Protocol (UDP) packets. For this work, the simulator is set up to send and receive packets comprising desired data every 0.1 second. In X-Plane, it is possible to simulate a number of flight emergencies for the purpose of training pilots. Emergencies range from severe weather conditions to system failure such as engine failure or fire.

2) The IAS Interface

The IAS Interface is responsible for data flow between the flight simulator and the system in both directions. The Interface contains control command buttons that provide a simplified yet sufficient aircraft control interface which can be used to perform basic tasks of piloting an aircraft such as take-off and landing in the simulator while being able to control other systems such as fuel and fire systems. It also displays flight data received from the simulator.

Data collection is started immediately before demonstration, then; the pilot uses the Interface to perform the piloting task to be learned. The Interface collects flight data from X-Plane over the network using UDP packets, and collects the pilot's actions while performing the task, which are also sent back to the simulator as manual control commands. The Interface organizes the collected flight data received from the simulator (inputs), and the pilot's actions (outputs) into vectors of inputs and outputs, which are sent to the database every 1 second.

3) Database

An SQL Server database stores all data captured from the pilot demonstrator and X-Plane, which are received from the Interface. The database contains tables designed to store: 1. Flight data as inputs, and 2. Pilot's actions as outputs. These tables are then used as training datasets to train the Artificial Neural Networks of the IAS.

B. Training

1) Artificial Neural Networks

After the human pilot data collection step is completed, Artificial Neural Networks are used to generate learning models from the captured datasets through offline training. Fig. 2 illustrates the training step.

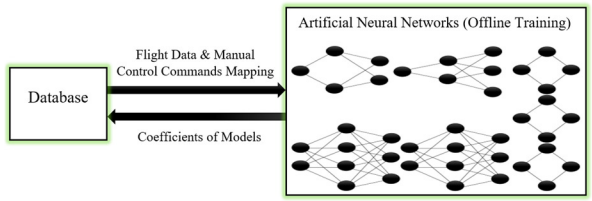

Fig. 2. Block diagram illustrating the IAS components used during training.

Ten feedforward Artificial Neural Networks comprise the core of the IAS. Each ANN is designed and trained to handle specific controls and tasks. The ANNs are: Taxi Speed Gain ANN, Take Off ANN, Rejected Take Off ANN, Aileron ANN, Rudder ANN, Cruise Altitude ANN, Cruise Pitch ANN, Fire Situation ANN, Emergency Landing Pitch ANN, and Emergency Landing Altitude ANN. The inputs and outputs which represent the gathered data and relevant actions, and the topologies of the ten ANNs are illustrated in Fig. 3.

The method for choosing ANN topologies in this work is based on a rule-of-thumb [15] which indicates that problems requiring more than one hidden layer are rarely encountered. This rule follows an approach that tries to avoid under-fitting caused by too few neurons in the hidden layer, or over-fitting caused by too many neurons, by having the number of hidden neurons less than or equal to twice the size of the input layer.

Before training, the datasets are normalized, and retrieved from the database. Then, the datasets are fed to the ANNs. Next, Sigmoid (1) [15] and Hyperbolic Tangent (Tanh) (2) [15] functions are applied for the neuron activation step, where $f(x)$ is the activation function for each neuron, and x is the relevant input value:

$$f(x) = \frac{1}{1+e^{-x}} \tag{1}$$

$$f(x) = \frac{e^{2x}-1}{e^{2x}+1} \tag{2}$$

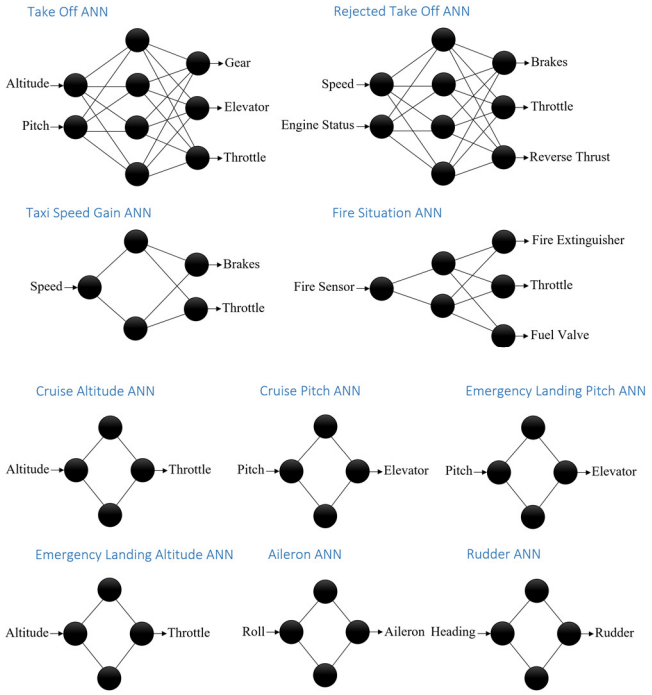

Fig. 3. Inputs, outputs, and the topologies of the ten ANNs representing the core of the Intelligent Autopilot System. Each ANN is designed and trained to handle a specific task.

[1] "X-Plane 10 Global
http://www.x-plane.com

The Sigmoid activation function (1) is used by the Taxi Speed Gain ANN, Take Off ANN, Emergency Landing Altitude ANN, Rejected Take Off ANN, and the Fire Situation ANN, while (2) is used by the rest since their datasets contain negative values.

Next, Backpropagation is applied. Based on the activation function, (3) [16], or (4) [16] are applied to calculate the error signal (δ) where t_n is the desired target value and a_n is the actual activation value:

$$\delta_n = (t_n - a_n)a_n(1 - a_n) \quad (3)$$

$$\delta_n = (t_n - a_n)(1 - a_n)(1 + a_n) \quad (4)$$

Finally, coefficients of models (weights and biases) are updated using (5) [17] where $\delta w_{i,j}$ is the change in the weight between nodes j and k.

$$w_{i,j} = w_{i,j} + \delta w_{i,j} \quad (5)$$

When training is completed, the learning models are generated, and the free parameters or coefficients represented by weights and biases of the models are stored in the database.

C. Autonomous Control

Once trained, the IAS can now be used for autonomous control. Fig. 4 illustrates the components used during the autonomous control step.

1) The IAS Interface

Here, the Interface retrieves the coefficients of the models from the database for each trained ANN, and receives flight data from the flight simulator every 0.1 second. The Interface organizes the coefficients into sets of weights and biases, and organizes data received from the simulator into sets of inputs for each ANN. The relevant coefficients, and flight data input sets are then fed to the Flight Manager and the ANNs of the IAS to produce outputs. The outputs of the ANNs are sent to the Interface which sends them to the flight simulator as autonomous control commands using UDP packets every 0.1 second.

2) The Flight Manager Program

The Flight Manager is a program which resembles a Behaviour Tree [18]. The purpose of the Flight Manager is to manage the ten ANNs of the IAS by deciding which ANNs are to be used simultaneously at each moment. The Flight Manager starts by receiving flight data from the flight simulator through the interface of the IAS, then it detects the flight condition and phase by examining the received flight data, and decides which ANNs are required to be used given the flight condition (normal/emergency/fire situation) and phase (taxi speed gain/take off/cruise/emergency landing). Fig. 5 illustrates the process which the Flight Manager follows.

3) Artificial Neural Networks

The relevant set of flight data inputs received through the Interface is used by the ANNs' input neurons along with the relevant coefficients to predict control commands given the flight status by applying (1) and (2). The values of the output layers are sent to the Interface which sends them to the flight simulator as autonomous control commands. Taxi Speed Gain ANN is used while on the runway just before take off to predict the suitable brakes and throttle command values. Take Off ANN is used after a certain take off speed is achieved to predict gear, elevator, and throttle command values. Rejected Take Off ANN is used to abort take off if necessary by predicting brakes, throttle, and reverse throttle command values. Aileron ANN is used to control the aircraft's roll immediately after take off. Rudder ANN is used to control the aircraft's heading before take off, and yaw when airborne in case one engine fails and creates drag. Cruise Altitude ANN is used to control the aircraft's desired cruising altitude by predicting the throttle command value. Cruise Pitch ANN controls the pitch while cruising by predicting the elevator command value. Fire Situation ANN is used in case of fire by predicting fuel valve and fire extinguishing control commands. Emergency Landing Pitch ANN maintains a certain pitch during emergency landing to lose speed without stalling and to prevent a nose first crash. Emergency Landing Altitude ANN controls the throttle in case of a single engine failure.

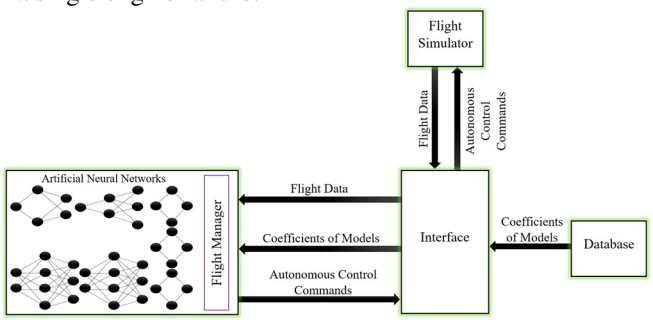

Fig. 4. Block diagram illustrating the IAS components used during autonomous control.

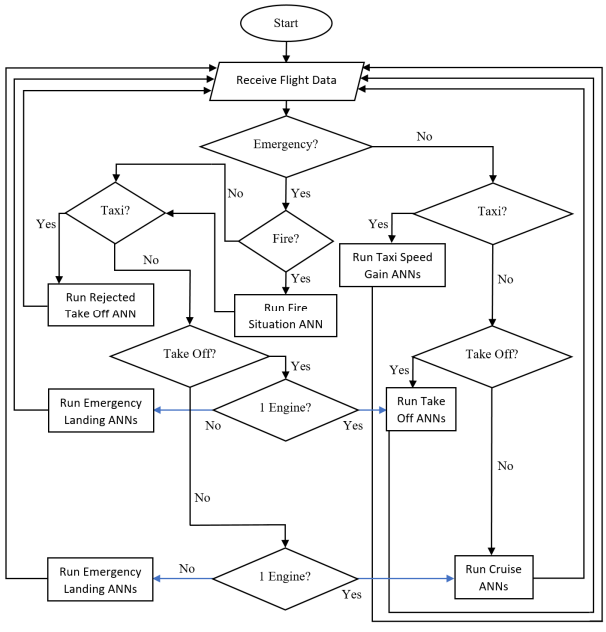

Fig. 5. A Flowchart illustrating the process which the Flight Manager program follows to decided which ANNs are to be used.

IV. EXPERIMENTS

Our previous work [1] provides detailed explanations of the experiments of autonomous taxi speed gain, take off, climb, and applying rudder and aileron to correct heading and roll deviations under normal and severe weather conditions. The new approach in this paper is to segment the training dataset of taxi speed gain, take off, and climb into three different sets that are handled separately by three ANNs (Taxi Speed Gain ANN, Take Off ANN, and Cruise ANN) instead of just one ANN. This work also introduces four new ANNs in order to learn flight emergency procedures for the first time.

In order to assess the effectiveness of the proposed approach in this paper, the Intelligent Autopilot System was tested in four experiments: A. Rejecting take off, B. Emergency landing, C. Maintaining a cruising altitude, and D. Handling single engine failure/fire while airborne. Each experiment is composed of 20 attempts by the IAS to perform autonomously under the given conditions.

The human pilot who provided the demonstrations is the first author. The simulated aircraft used for the experiments is a Boeing 777 as we want to experiment using a more complex model with more than one engine rather than a light single-engine model. The experiments are as follows:

A. *Rejecting Take Off*

The purpose of this experiment is to assess the behaviour of the IAS compared to the behaviour of the human pilot when a Rejected Take Off (RTO) is required.

1) *Data Collection*

In this experiment, the human pilot used the IAS Interface to perform the following in the flight simulator: reject take off when one engine fails or catches fire, and when two engines fail or catch fire (one demonstration for each scenario). The flight simulator was set to simulate the failure or fire conditions for one or two engines immediately after the user presses a hot key on the keyboard. Rejecting take off is performed by going to full reverse thrust and engaging brakes. In case of fire, the human pilot turned off the fuel valve, turned on the fire extinguishing system, and went to full throttle to burn the fuel left in the engine(s). While the pilot performed the demonstration, the Interface collected speed and engine status as inputs, and brakes, throttle, and reverse thrust control data as outputs. The Interface stored the collected data in the database as the training dataset for the Rejected Take Off ANN. The Interface also collected fire sensor readings as input, and fire extinguisher, throttle, and fuel valve control data as outputs. The Interface stored the collected data in the database as the training dataset for the Fire Situation ANN.

2) *Training*

For this experiment, the Rejected Takeoff ANN, and the Fire Situation ANN were trained until low Mean Squared Error (MSE) values were achieved (below 0.001).

3) *Autonomous Control*

After training the ANNs on the relevant training datasets, the aircraft was reset to the runway in the flight simulator to test autonomous RTO multiple times under different scenarios (one and two engine(s) failure and fire), the simulator was set to simulate the desired emergency scenario, and the IAS was engaged. When the flight manager detects the emergency, it stops the Taxi Speed Gain ANN, and runs the Rejected Takeoff ANN and the Fire Situation ANN simultaneously to reject take off and handle fire autonomously. Through the Interface, ANNs receive: 1. Relevant flight data from the flight simulator as inputs, and 2. Coefficients of the relevant models from the database to predict and output command controls that are sent to the flight simulator. This process allows the IAS to autonomously perform the learned task: rejecting take off if necessary. This was repeated 20 times for each scenario to assess performance consistency.

B. *Emergency Landing*

The purpose of this experiment is to assess the behaviour of the IAS compared to the behaviour of the human pilot when a forced or emergency landing is required.

1) *Data Collection*

In this experiment, the human pilot used the IAS Interface to perform the following in the flight simulator: emergency landing when two engines fail or catch fire (one demonstration for each scenario). The flight simulator was set to simulate the failure or fire conditions for two engines immediately after the user presses a hot key on the keyboard. Emergency landing is performed by maintaining a controlled glide using the elevators to insure a gradual loss of speed and altitude without stalling the aircraft, by maintaining a slight positive pitch. If there is any power left in the engines, the throttle is used to aid the gliding phase. In case of fire, the human pilot turned off the fuel valve, and turned on the fire extinguishing system. In this scenario going to full throttle to burn the fuel left in the engines is not possible since both engines do not have sufficient power. While the pilot performed the demonstration, the Interface collected pitch as input, and elevator control data as output. The Interface stored the collected data in the database as the training dataset for the Emergency Landing Pitch ANN. The Interface also collected altitude as input, and throttle control data as output. The Interface stored the collected data in the database as the training dataset for the Emergency Landing Altitude ANN.

2) *Training*

For this experiment, the Emergency Landing Pitch ANN, and the Emergency Landing Altitude ANN were trained until low Mean Squared Error (MSE) values were achieved (below 0.001 for the Emergency Landing Pitch ANN and below 0.2 for the Emergency Landing Altitude ANN).

3) *Control*

After training the ANNs on the relevant training datasets, the aircraft was reset to the runway in the flight simulator to test autonomous emergency landing multiple times under different scenarios (both engines failure or fire), the simulator was set to simulate the desired emergency scenario, and the IAS was engaged. After the IAS took the aircraft airborne, and when the flight manager detects the emergency, it stops the Take Off ANN (during climb), or the cruise ANNs, and runs the Emergency Landing Pitch ANN, and the Emergency Landing Altitude ANN simultaneously to maintain a controlled glide while descending to the ground. Through the Interface, the

ANNs receive: 1. Relevant flight data from the flight simulator as inputs, and 2. Coefficients of the relevant models from the database to predict and output command controls that are sent to the flight simulator. This process allows the IAS to autonomously perform learned task: emergency landing by maintaining a controlled glide. This was repeated 20 times for each scenario to assess performance consistency.

C. Maintaining a Cruising Altitude

The purpose of this experiment is to assess the behaviour of the IAS compared to the behaviour of the human pilot while maintaining a desired cruising altitude.

1) Data Collection

In this experiment, the human pilot used the IAS Interface to maintain a cruising altitude in the flight simulator by increasing and decreasing the throttle, and by using the elevator to maintain a fairly leveled pitch (one demonstration). While the pilot performed the demonstration, the Interface collected altitude as input, and throttle control data as output. The Interface stored the collected data in the database as the training dataset for the Cruise Altitude ANN. The Interface also collected pitch as input, and elevator control data as output. The Interface stored the collected data in the database as the training dataset for the Cruise Pitch ANN.

2) Training

For this experiment, the Cruise Altitude ANN, and the Cruise Pitch ANN were trained until low Mean Squared Error (MSE) values were achieved (below 0.02 and 0.001 respectively).

3) Autonomous Control

After training the ANNs on the relevant training datasets, the aircraft was reset to the runway in the flight simulator to test the ability of maintaining a desired cruise altitude autonomously, and the IAS was engaged. After the IAS took the aircraft airborne, continued to climb, and reached the proximity of the desired altitude, the system's ability to maintain the given altitude was observed. Through the Interface, the ANNs receive: 1. Relevant flight data from the flight simulator as inputs, and 2. Coefficients of the relevant models from the database to predict and output command controls that are sent to the flight simulator. This process allows the IAS to autonomously perform learned task: maintain a desired cruising altitude. This was repeated 20 times for each scenario to assess performance consistency.

D. Handling Single Engine Failure/Fire while Airborne

The purpose of this experiment is to assess the behaviour of the IAS in case of an engine failure or fire while airborne.

1) Data Collection

In this experiment, the human pilot did not provide an explicit demonstration for the single engine failure. Instead, it was intended to test the already trained ANNs, and determine whether their models are able to generalize well in this new scenario where the failed engine creates a drag, and forces the aircraft to descend, and creates a yaw deviation towards the failed engine's side.

2) Training

For this experiment, the previously trained models of the Cruise Altitude ANN, the Cruise Pitch ANN, and the rudder ANN from our previous work [1] were used.

3) Autonomous Control

After setting the simulator to simulate the desired emergency scenario (single engine failure or fire), and after the IAS took the aircraft airborne, when the flight manager detects the emergency, it continues to use the same ANNs (Take Off ANN, or cruise ANNs), and runs the Fire Situation ANN if fire is detected, to fly autonomously using the power left from the engine that operates normally. Through the Interface, the ANNs receive: 1. Relevant flight data from the flight simulator as inputs, and 2. Coefficients of the relevant models from the database to predict and output command controls that are sent to the flight simulator. This was repeated 20 times for each scenario to assess performance consistency.

Throughout all the experiments, the Rudder and Aileron ANNs from our previous work [1] are used normally during the different phases.

V. Results

The following section describes the results of the conducted tests. The 20 attempts by the IAS to handle each scenario autonomously were averaged and compared with the performance of the human pilot when applicable.

A. Rejecting Take Off

Two models were generated with the MSE values as table I shows. Fig. 6 illustrates the behaviour of the IAS when controlling the transition of flight modes under normal conditions, while Fig. 7 illustrates the behaviour of the IAS when engine(s) failure or fire is detected and a Rejected Take Off (RTO) is performed. The results of the 20 experiments showed strong consistency by following the correct procedure in each experiment with a 100% accuracy rate.

B. Emergency Landing

Two models were generated with the MSE values as table I shows. Fig. 8 and 9 illustrate a comparison between the human pilot and the IAS while maintaining a positive pitch during emergency landing, and their altitude (sink rate). The pitch Mean Absolute Deviation (MAD) results (0.024 for the IAS and 0.196 for the human pilot) show less deviation and a steady behaviour of the IAS due to the good model fit as can be seen in Fig. 8. Fig. 10 illustrates the behaviour of the IAS when both engines failure or fire is detected and a forced or emergency landing is performed. The results of the 20 experiments showed strong consistency by following the correct procedure in each experiment with a 100% accuracy rate.

TABLE I
MSE VALUES OF THE MODELS GENERATED FOR THE REJECTED TAKE OFF AND THR EMERGENCY LANDING EXPERIMENTS.

ANN	MSE
Rejected Takeoff ANN	0.000999
Fire Situation ANN	0.000999
Emergency Landing Pitch ANN	0.000997
Emergency Landing Altitude ANN	0.196117

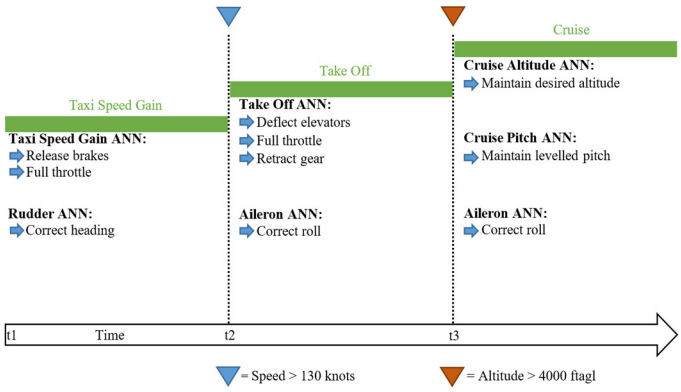

Fig. 6. The behaviour of the IAS when controlling the transition of flight modes under normal conditions. Different ANNs are used in each flight mode.

Fig. 7. (Rejected Take Off experiment) The behaviour of the IAS when engine(s) failure or fire is detected and a Rejected Take Off (RTO) is performed. The Fire Situation ANN is used only when fire is detected.

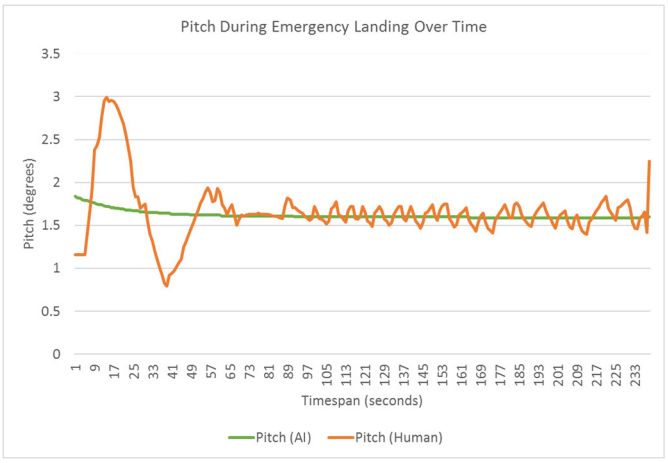

Fig. 8. (Emergency landing experiment) A comparison between the human pilot and the Intelligent Autopilot System's pitch during emergency landing. In this case the human pilot struggled to generate perfect training data so our training approach was designed to prevent overfitting, instead creating a general model (good fit) which provided the desired performance.

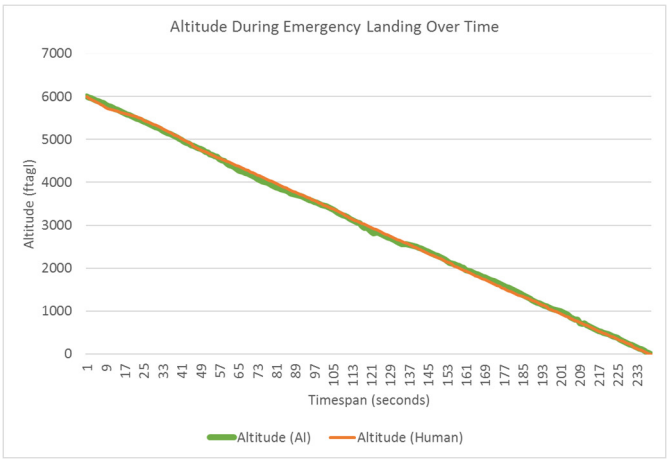

Fig. 9. (Emergency landing experiment) A comparison between the human pilot and the Intelligent Autopilot System's altitude during emergency landing. The results show a significantly close sink rate of about 1500 ftagl per minute.

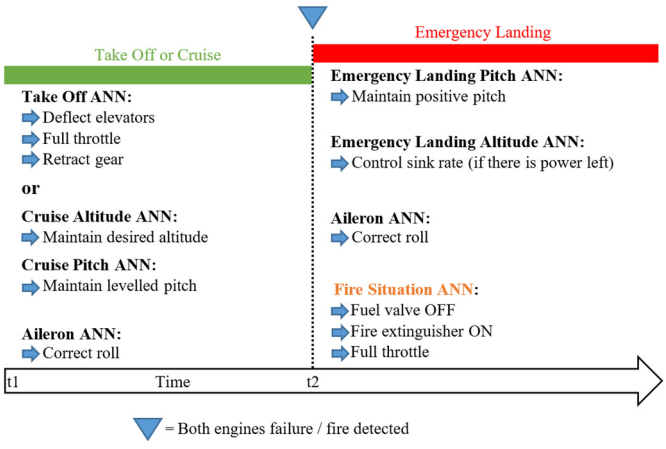

Fig. 10. (Emergency landing experiment) The behaviour of the IAS when both engines failure or fire is detected during either take off or cruise, and an emergency landing is performed. The Fire Situation ANN is used only when fire is detected.

C. *Maintaining a Cruise Altitude*

Two models were generated with the MSE values as table II shows. Fig. 11 and 12 illustrate a comparison between the human pilot and the IAS while maintaining a desired cruising altitude. The altitude Mean Absolute Deviation (MAD) results (85.8 for the IAS and 204.58 for the human pilot) shows less deviation of altitude and a steady behaviour of the IAS due to the good model fit as can be seen in Fig. 11.

TABLE II
MSE VALUES OF THE MODELS GENERATED FOR THE CRUISE EXPERIMENT.

ANN	MSE
Cruise Altitude ANN	0.017574
Cruise Pitch ANN	0.000835

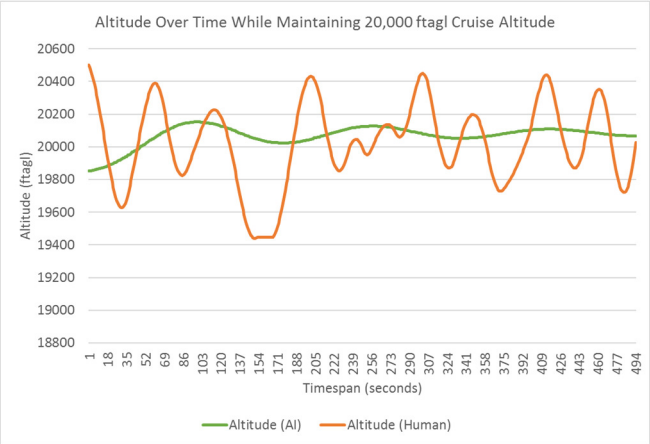

Fig. 11. (Maintaining a cruise altitude experiment) A comparison between the human pilot and the Intelligent Autopilot System's altitude during cruising. While the human pilot demonstrator struggled to maintain a desired cruise altitude of 20,000 ftagl, the IAS performed better due to the good fit of the generated learning model.

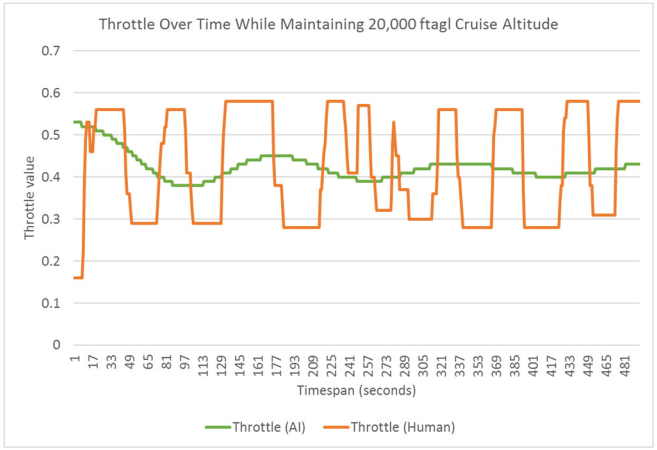

Fig. 12. (Maintaining a cruise altitude experiment) The IAS manipulation of throttle to maintain a desired cruise altitude of 20,000 ftagl compared with the human pilot. The IAS manipulated the throttle smoothly compared to the human pilot due to the good fit of the generated learning model.

D. Handling Single Engine Failure/Fire while Airborne

As mentioned in part (IV) the human pilot did not provide an explicit demonstration for the single engine failure scenario. Instead, it was intended to test the already trained ANNs, and determine whether their models are able to generalize well in this new scenario's experiment. Fig. 13 illustrates the behaviour of the IAS when a single engine fails or catches fire during take off or cruise. The system was intended to carry on flying, apply the rudder ANN from our previous work [1], and run the Fire Situation ANN in case of fire. The results of the 20 experiments showed strong consistency by following the correct procedure in each experiment accurately. Fig. 14 illustrates how the IAS continues to fly while losing altitude gradually compared to the aircraft's autopilot under the same situation.

VI. ANALYSIS

As can be seen in Fig. 7, the rejected take off experiment presented excellent results. The IAS was capable of imitating the human pilot's actions and behaviour with excellent accuracy, and strong consistency by following the correct procedure in each experiment accurately.

Fig. 8 to 10 (the emergency landing experiment) show very desirable results of the ability of the IAS to imitate the human pilot's demonstration of controlling an emergency landing. They show the ability of the IAS to perform the learned sink rate which enabled the aircraft to hit the ground smoothly without being severely wrecked. The flight simulator measures the G force effect on the aircraft's frame, and informs the user in case of an unsurvivable crash. It should be mentioned that selecting a suitable landing surface is not within the scope of this work.

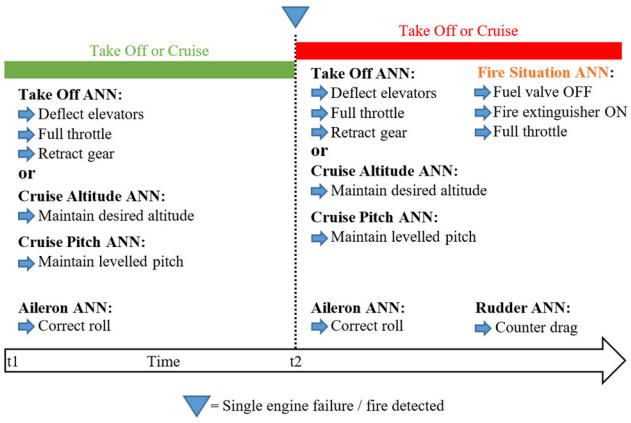

Fig. 13. (Handling single engine failure/fire experiment) The behaviour of the IAS when a single engine failure or fire is detected during either take off or cruise. The Fire Situation ANN is used only when fire is detected. The ANNs used during Take Off or Cruise perform the same tasks as Fig. 6 shows, while the Aileron ANN continues to correct roll.

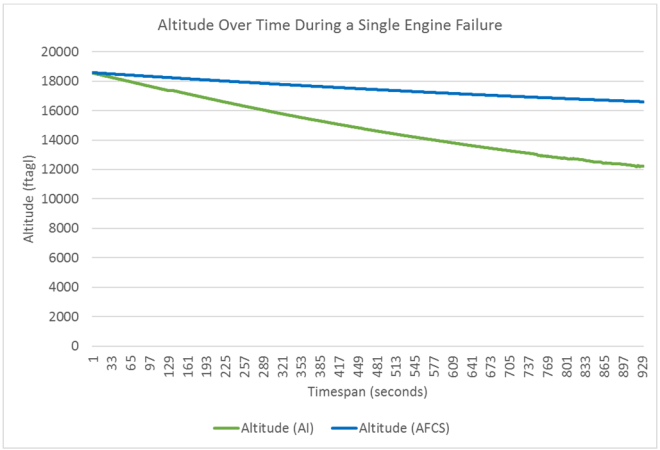

Fig. 14. (Handling single engine failure/fire experiment) Comparing the altitude loss rate of the IAS and the aircraft's AFCS. Since the AFCS is not aware of the single engine failure situation, it compensates by increasing the throttle aggressively, which results in a smaller altitude loss rate, but puts excessive stress on the single operating engine.

Fig. 11 and 12 (maintaining a cruise altitude experiment) show very desirable results of the ability of the IAS to learn how to use throttle and elevator to maintain a given altitude. They illustrate the ability of the IAS to perform better than the human pilot teacher due to the achieved good fit of the learning models. This can also be seen in Fig. 8 (the emergency landing experiment).

As can be seen in Fig. 13 and 14, the single engine failure/fire experiment presented excellent results. The IAS was capable of using the already learned models to continue flying while gradually losing altitude. Although the aircraft's standard autopilot maintained a better altitude in the short term, by aggressively increasing engine thrust it increases the likelihood of engine failure in the remaining engine, with potentially catastrophic results.

The system was able to imitate multiple human pilot's skills and behaviour after being presented with very limited examples. This is due to the approach of segmenting the problem of autonomous piloting while handling uncertainties into small blocks of tasks, and assigning multiple ANNs specially designed and trained for each task, which resulted in the generation of highly accurate models as tables I, and II show.

VII. Conclusion & Future Work

In this work, a robust approach is proposed to "teach" autopilots how to handle uncertainties and emergencies with minimum effort by exploiting Learning by Imitation also known as Learning from Demonstration.

The experiments were strong indicators towards the ability of Supervised Learning with Artificial Neural Networks to capture low-level piloting tasks such as the rapid manipulation of the elevator and throttle to maintain a certain pitch or a given altitude. The experiments showed the ability of the IAS to capture high-level tasks such as coordinating the necessary actions to reject take off and extinguish fire.

Breaking down the piloting tasks, and adding more Artificial Neural Networks enhanced performance and accuracy, and allowed the coverage of a wider spectrum of tasks.

The aviation industry is currently working on solutions which should lead to decreasing the dependence on crew members. The reason behind this is to lower workload, human error, stress, and emergency situations where the captain or the first officer becomes incapable, by developing autopilots capable of handling multiple scenarios without human intervention. We anticipate that future Autopilot systems which make of methods proposed here could improve safety and save lives.

Future effort will focus on giving the IAS the ability to learn how to fly a pre-selected course, and land safely in an airport. The IAS should be capable of avoiding no-fly zones that are either pre-identified, or detected during the flight such as severe weather systems detected by the aircraft's radar.

The Flight Manager program should be redesigned to utilize Artificial Neural Networks to classify the situation (normal or emergency), and predict the suitable flight control law or mode given the situation.

The problem of sensor fault and denial should be investigated to test the feasibility of teaching the IAS how to handle such scenarios.

References

[1] H. Baomar and P. J. Bentley, "An Intelligent Autopilot System that learns piloting skills from human pilots by imitation," *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, Arlington, VA, USA, 2016, pp. 1023-1031. doi: 10.1109/ICUAS.2016.7502578

[2] Nelson, Robert C. Flight stability and automatic control. Vol. 2. WCB/McGraw Hill, 1998.

[3] G. J. J. Ducard, "Fault-Tolerant Flight Control and Guidance Systems: Practical Methods for Unmanned Aerial Vehicles," Springer, 2009.

[4] Sadeghzadeh, I. & Zhang, Y. (2011). A Review on Fault-Tolerant Control for Unmanned Aerial Vehicles (UAVs). St. Louis, Missouri, USA: The American Institute of Aeronautics and Astronautics.

[5] Rao, Faheem Muhammad et al. "UAV Emergency Landing Site Selection System Using Machine Vision". J MACH INTELL 1.1 (2016): n. pag. Web.

[6] P. Li, X. Chen and C. Li, "Emergency landing control technology for UAV," *Guidance, Navigation and Control Conference (CGNCC), 2014 IEEE Chinese*, Yantai, 2014, pp. 2359-2362.

[7] Steingrube, Silke et al. "Self-Organized Adaptation Of A Simple Neural Circuit Enables Complex Robot Behaviour". Nat Phys 6.3 (2010): 224-230. Web.

[8] P. Y. Shinzato, V. Grassi, F. S. Osorio and D. F. Wolf, "Fast visual road recognition and horizon detection using multiple artificial neural networks," *Intelligent Vehicles Symposium (IV), 2012 IEEE*, Alcala de Henares, 2012, pp. 1090-1095.

[9] M. B. W. D. Oliveira and A. D. A. Neto, "Optimization of Traffic Lights Timing Based on Multiple Neural Networks," *2013 IEEE 25th International Conference on Tools with Artificial Intelligence*, Herndon, VA, 2013, pp. 825-832.

[10] L. Huang, Q. Wu, Y. Chen, S. Hong and X. Huang, "Gesture Recognition Based on Fusion Features from Multiple Spiking Neural Networks," *Communication Systems and Network Technologies (CSNT), 2015 Fifth International Conference on*, Gwalior, 2015, pp. 1167-1171.

[11] Kien Tuong Phan, T. H. Maul and Tuong Thuy Vu, "A parallel circuit approach for improving the speed and generalization properties of neural networks," *Natural Computation (ICNC), 2015 11th International Conference on*, Zhangjiajie, 2015, pp. 1-7.

[12] Wei, F., Amaya-Bower, L., Gates, A., Rose, D. and Vasko, T. (2016). The Full-Scale Helicopter Flight Simulator Design and Fabrication at CCSU. 57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference.

[13] Jirgl, M., Boril, J., Jalovecky, R. (2015). The identification possibilities of the measured parameters of an aircraft model and pilot behavior model on the flight simulator. International Conference on Military Technologies (ICMT), vol., no., pp.1-5.

[14] Kaviyarasu, A. and Senthil Kumar, K. (2014). Simulation of Flapping-wing Unmanned Aerial Vehicle using X-plane and Matlab/Simulink. Defence Science Journal, 64(4), pp.327-331.

[15] Heaton, J. (2005). Introduction to neural networks with Java. St. Louis: Heaton Research.

[16] McClelland, J. (2015). Explorations in Parallel Distributed Processing: A Handbook of Models, Programs, and Exercises (2nd ed.). Stanford.

[17] Tveter, D. (1995). Chapter 2, The Backprop Algorithm.

[18] K. Winter, I. J. Hayes and R. Colvin, "Integrating Requirements: The Behavior Tree Philosophy," 2010 8th IEEE International Conference on Software Engineering and Formal Methods, Pisa, 2010, pp. 41-50.

Autonomous Navigation and Landing of Airliners Using Artificial Neural Networks and Learning by Imitation

Haitham Baomar, Peter J. Bentley

Dept. of Computer Science, University College London, Gower Street, London, WClE 6BT, U.K.
Email: {h.baomar, p.bentley} @ cs.ucl.ac.uk

Abstract— We introduce the Intelligent Autopilot System (IAS) which is capable of autonomous navigation and landing of large jets such as airliners by observing and imitating human pilots using Artificial Neural Networks and Learning by Imitation. The IAS is a potential solution to the current problem of Automatic Flight Control Systems of being unable to perform full flights that start with takeoff from a given airport, and end with landing in another. A navigation technique, and a robust Learning by Imitation approach are proposed. Learning by Imitation uses human pilots to demonstrate the task to be learned in a flight simulator while training datasets are captured from these demonstrations. The datasets are then used by Artificial Neural Networks to generate control models automatically. The control models imitate the skills of the human pilot when banking to navigate between waypoints, and when performing final approach and landing, while a flight manager program generates the flight course, and decides which ANNs to be fired given the current flight phase. Experiments show that, even after being presented with limited examples, the IAS can handle such flight tasks with high accuracy. The proposed IAS is a novel approach towards achieving full control autonomy of large jets using ANN models that match the skills and abilities of experienced human pilots.

I. INTRODUCTION

Human pilots are trained to perform piloting tasks that are required during the different phases of the flight. Performing a complete flight cycle starts with a ground-run on the runway to gain speed, rotate after a certain airspeed is achieved, climb, cruise while navigating between waypoints, descend, prepare for final approach while intercepting the landing runway path line, touchdown, flare, and lower airspeed before coming to a full stop [1].

In contrast, Automatic Flight Control Systems (AFCS/Autopilot) are highly limited, capable of performing minimal piloting tasks. Although modern autopilots can maintain or hold a desired heading, speed, altitude, and even perform auto-land, they cannot handle complete flight cycles automatically, and they must be engaged and operated manually by the human pilots to constantly change and update the desired parameters. In addition, modern autopilots cannot handle flight uncertainties such as severe weather conditions, or emergency situations such as system failures. The limitations of autopilots require constant monitoring of the system and the flight status by the flight crew to react quickly to any undesired situation or emergencies. The reason for such limitations of conventional AFCS is that it is not feasible to anticipate everything that could go wrong with a flight, and incorporate all of that into the set of rules or control models "hardcoded" in an AFCS.

This work aims to address this problem by creating an Intelligent Autopilot System (IAS) with the capability to perform autonomous navigation, and to learn how to land from human pilots by applying the Learning by Imitation concept with Artificial Neural Networks. The IAS is a novel approach which introduces the possibility to transfer human intelligence and intuitions required to pilot an aircraft to an autonomous system. By using this approach, we aim to extend the capabilities of modern autopilots and enable them to autonomously adapt their piloting to suit multiple scenarios ranging from normal to emergency situations. This work builds on previous work by the authors [2][3] by adding navigation and landing capabilities to the IAS.

This paper is structured as follows: part (II) reviews related literature on autonomous navigation and landing. Part (III) explains the Intelligent Autopilot System (IAS). Part (IV) describes the experiments, Part (V) describes the results by comparing the behaviour of the human pilot with the behaviour of the Intelligent Autopilot System, and part (VI) provides an analysis of the results. Finally, we provide conclusions and future work.

II. BACKGROUND

A. Autonomous Navigation

Autonomous navigation is the ability of the travelling vehicle to estimate the state of its trajectory automatically [4]. In autonomous aerial systems, such as UAVs or cruise missiles, it is common to estimate the state of trajectory by fusing data from multiple navigation systems such as the Inertial Navigation System (INS) and the Global Navigation Satellite System (GNSS) such as the Global Positioning System (GPS). It is also possible to fuse additional data from different types of systems such as vision-based navigation systems [4].

In [4], an image matching system which uses aerial images acquired during flights in addition to aerial georeferenced images, is proposed to estimate the position of a UAV. The proposed image matching system applies image-edge detection algorithms to the acquired images, and the posterior automatic image registration to estimate the location of the UAV [4]. An

Artificial Neural Network (ANN) with an optimal architecture set by the Multiple Particle Collision Algorithm (MPCA) is used to detect the edges, while the automatic image registration is acquired through a cross-correlation process [4].

Different navigation and path planning approaches are being investigated as well. In [5], an algorithm based on inspection path planning is proposed, which is tailored inherently for structural inspection. The proposed algorithm is designed to compute full coverage and collision–free paths depending on a model of the UAV's nonholonomic constraints [5]. A resampling of the viewpoint technique applies randomized sampling which allows the designed algorithm to achieve continuous enhancements of the path cost without affecting the desired area to be covered [5]. In addition, navigation with a collision avoidance capability is achieved by applying Boundary Value Solver and a motion planner [7] for the used UAV model [5].

Relying on GPS alone for autonomous navigation is proposed in [8], where a cost-efficient cruise control system is designed for a GT-500 recreational aircraft using affordable and off-the-shelve components such as an Arduino system [8].

In [9], a GPS based generic trajectory prediction and smoothing algorithm is proposed. The algorithm is designed to be able to handle both accurate frequency legs, and inaccurate legs that are present in old flight procedures, that have not been updated using advanced Flight Management Systems (FMS) [9]. The estimation of the desired trajectory is calculated using numerical integration of the different states of the aircraft given the flight path [9].

B. Autonomous Landing

Pilots operating Remotely Piloted Aircraft Systems (RPAS) or UAVs do not get to feel the aircraft they are flying as onboard pilots do [10]. Feeling the forces of the surrounding environment such as the wind, and the aircraft itself, such as getting a feel of how the engines are behaving, the vibrations, motions, and so on, is not possible for ground pilots [10]. The lack of this onboard sensing affects the situational awareness which is a crucial factor that pilots depend on especially during the most difficult flight phases such as landing, therefore, most UAV accidents happen during landing [10]. In addition, performing an optimum landing all the time is important for maintenance cost reduction, and durability preservation [10]. So, investigating the possibilities of developing autonomous landing systems (Auto Land) for UAVs has been a significant challenge, and is being covered in recent research efforts [10].

In [10], a landing sequence algorithms is proposed, which can either be initiated by the ground pilot, or automatically during emergency situations such as the loss of connection between the UAV and the ground command and control station. The proposed landing system utilizes the Global Positioning System (GPS) along with geometry to orient the UAV to a desirable point in space from which it can initiate the descend process [10]. The algorithm works by plotting multiple slopes via MATLAB, and are considered as potential descend paths that the UAV can follow, in a fashion like creating a virtual inverted cone, where the circular base of the cone can act as a potential point of descend, and the taper surface can be considered as the glide path [10].

To achieve higher levels of accuracy required for landing on significantly small, or moving landing runways such as aircraft carriers, some recent research efforts are focusing on fusing multiple guidance systems, such as the work presented in [11]. The proposed system works by fusing readings from multiple systems or sensors including GPS, the aircraft's INS, the aircraft carrier's INS, and a vision-based navigation system mounted on the aircraft [11]. The system computes the aircraft-ship relative position, while the acceleration and velocity of the ship are sent to the aircraft via a dedicated data-link [11]. The aircraft-ship relative position, and the relative velocity are added to the state vector, and the relative position information retrieved from GPS, along with the airborne INS, the carrier's INS, and the vision-based navigation system are utilized to build the vector via a Kalman filter [11]. Finally, the relative position information having the same period as the one generated from the INS is calculated [11].

Introducing intelligent autonomy to the aviation industry through developing intelligent control techniques that fit into an overall flight management system capable of making the highest level of decisions, is expected to significantly enhance safety, and lower costs [12].

In addition of having limited capabilities, modern autopilots can contribute to catastrophes since they can only operate under certain conditions that fit their design and programming, otherwise, they cede control to the pilots, and with the lack of proper situational awareness and reaction, the result could be fatal [13]. Although the civil aviation sector that uses medium to large jets equipped with such autopilots, is the largest with the highest risk and costs, the current focus of the relevant and recent research efforts is on investigating and developing autonomous autopilots for Unmanned Aircraft Systems especially small and micro drones by introducing solutions that may not be suitable for Large jets such as airliners. Therefore, we propose a solution that can be applied to multiple aircraft categories including airliners and cargo airplanes. We believe that manned aircraft especially airliners require significant attention to enhance safety by addressing the limitations of modern autopilots and flight management systems, and the human error factor as well.

A review of the Autopilot problem, Artificial Neural Networks, and Learning by Imitation for Autonomous Flight Control is presented in our previous work [2].

III. THE INTELLIGENT AUTOPILOT SYSTEM

The proposed Intelligent Autopilot System (IAS) in this paper can be viewed as an apprentice that observes the demonstration of a new task by the experienced teacher, and then performs the same task autonomously. A successful generalization of Learning by Imitation should take into consideration the capturing of low-level models and high-level models, which can be viewed as rapid and dynamic sub-actions that occur in fractions of a second, and actions governing the whole process and how it should be performed strategically. It is important to

capture and imitate both levels to handle different piloting tasks successfully.

The IAS is made of the following components: a flight simulator, an interface, a database, a flight manager program, and Artificial Neural Networks. The IAS implementation method has three steps: A. Pilot Data Collection, B. Training, and C. Autonomous Control. In each step, different IAS components are used. The following sections describe each step and the components used in turn.

A. *Pilot Data Collection*

Fig. 1 illustrates the IAS components used during the pilot data collection step.

1) *Flight Simulator*

Before the IAS can be trained or can take control, we must collect data from a pilot. This is performed using X-Plane which is an advanced flight simulator that has been used as the simulator of choice in many research papers such as [14] [15] [16].

X-Plane is used by multiple organizations and industries such as NASA, Boeing, Cirrus, Cessna, Piper, Precession Flight Controls Incorporated, Japan Airlines, and the American Federal Aviation Administration.[1] X-Plane can communicate with external applications by sending and receiving flight status and control commands data over a network through User Datagram Protocol (UDP) packets. For this work, the simulator is set up to send and receive packets comprising desired data every 0.1 second.

2) *The IAS Interface*

The IAS Interface is responsible for data flow between the flight simulator and the system in both directions. The Interface contains control command buttons that provide a simplified yet sufficient aircraft control interface which can be used to perform basic tasks of piloting an aircraft such as take-off and landing in the simulator while being able to control other systems such as fuel and fire systems. It also displays flight data received from the simulator.

Data collection is started immediately before demonstration, then, the pilot uses the Interface to perform the piloting task to be learned. The Interface collects flight data from X-Plane over the network using UDP packets, and collects the pilot's actions while performing the task, which are also sent back to the simulator as manual control commands.

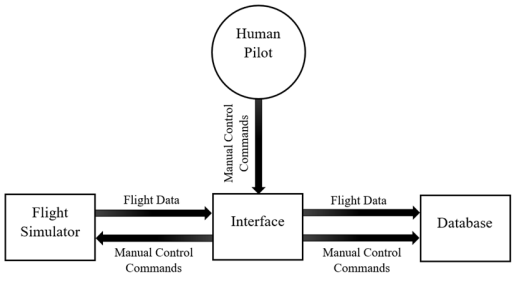

Fig. 1. Block diagram illustrating the IAS components used during the pilot data collection step.

The Interface organizes the collected flight data received from the simulator (inputs), and the pilot's actions (outputs) into vectors of inputs and outputs, which are sent to the database every 1 second.

3) *Database*

An SQL Server database stores all data captured from the pilot demonstrator and X-Plane, which are received from the Interface. The database contains tables designed to store: 1. Flight data as inputs, and 2. Pilot's actions as outputs. These tables are then used as training datasets to train the Artificial Neural Networks of the IAS.

B. *Training*

1) *Artificial Neural Networks*

After the human pilot data collection step is completed, Artificial Neural Networks are used to generate learning models from the captured datasets through offline training. Fig. 2 illustrates the training step.

Fourteen feedforward Artificial Neural Networks comprise the core of the IAS. Each ANN is designed and trained to handle specific controls and tasks. The ANNs that are relevant to this work are: Ground-run ANN, Rudder ANN, Takeoff ANN, Aileron ANN, Cruise Altitude ANN, Cruise Pitch ANN, Final Approach ANN, Final Approach Pitch ANN, Gear ANN, and Landing ANN. The other ANNs that handle emergency situations are discussed in our previous work [3]. The inputs and outputs which represent the gathered data and relevant actions, and the topologies of the ANNs are illustrated in Fig. 3.

The method for choosing ANN topologies in this work is based on an implication [17] which indicates that direct mapping problems requiring more than one hidden layer are rarely encountered, and compared to Deep Learning, this approach means that the system is more understandable and easier to test and verify compared to single deep solutions which are black-boxes unsuited for safety critical applications.

Before training, the datasets are retrieved from the database. Then, the datasets are fed to the ANNs. Next, Sigmoid (1) [18] and Hyperbolic Tangent (Tanh) (2) [18] functions are applied for the neuron activation step.

$$\Phi(x) = \frac{1}{1+ e^{-x}} \tag{1}$$

$$\Phi(x) = tanh(x) \tag{2}$$

where e is the exponential function, and x is the neuron output.

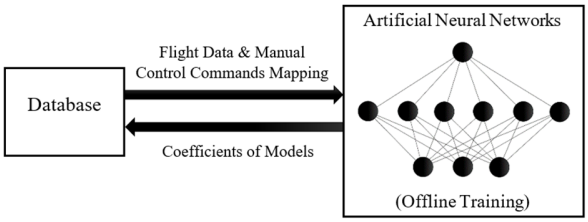

Fig. 2. Block diagram illustrating the IAS components used during training.

[1] X-Plane 10 Global. http://www.x-plane.com

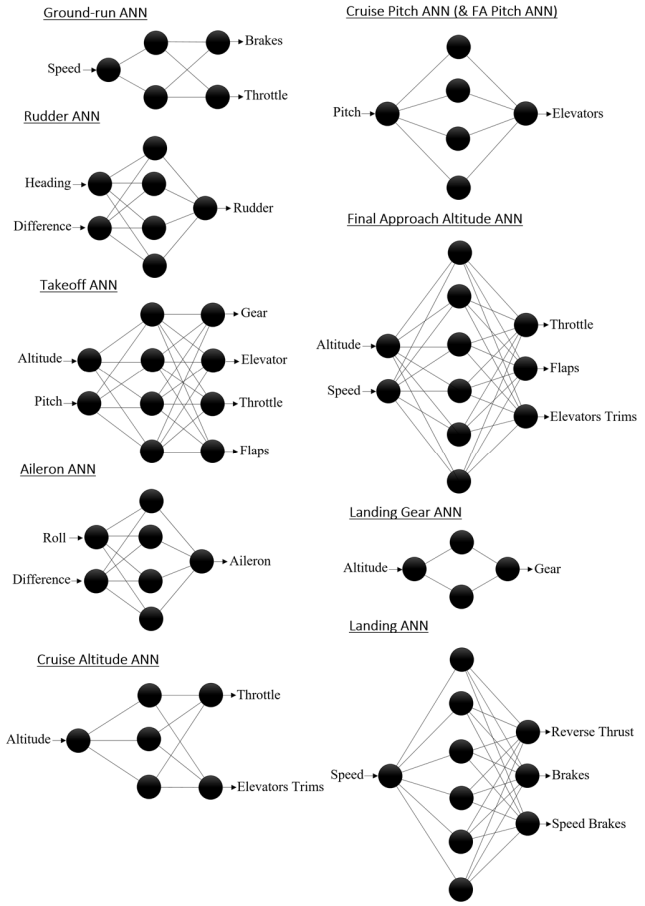

Fig. 3. Inputs, outputs, and the topologies of the ANNs relevant to this work. Each ANN is designed and trained to handle a specific task.

The Sigmoid activation function (1) is used by the Ground-run ANN, Takeoff ANN, Landing Gear ANN, and the Landing ANN, while (2) is used by the rest since their datasets contain negative values.

Next, Backpropagation is applied. Based on the activation function, (3) [18], or (4) [18] are applied to calculate the derivatives of the relevant activation function:

$$\Phi'(x) = \Phi(x)(1 - \Phi(x)) \quad (3)$$

$$\Phi'(x) = 1.0 - \Phi^2(x) \quad (4)$$

where phi (Φ) of x is the result of the activation function.

Finally, coefficients of models (weights and biases) are updated using (5) [17].

$$\Delta w_{(t)} = -\epsilon \frac{\partial E}{\partial w_{(t)}} + \alpha \Delta w_{(t-1)} \quad (5)$$

where ϵ is the learning rate, $\frac{\partial E}{\partial w_{(t)}}$ is the gradient, α is the momentum, and $\Delta w_{(t-1)}$ is the change in the previous weight.

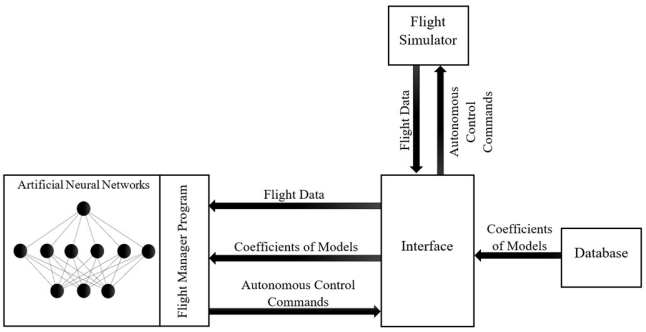

Fig. 4. Block diagram illustrating the IAS components used during autonomous control.

When training is completed, the learning models are generated, and the free parameters or coefficients represented by weights and biases of the models are stored in the database.

C. Autonomous Control

Once trained, the IAS can now be used for autonomous control. Fig. 4 illustrates the components used during the autonomous control step.

1) The IAS Interface

Here, the Interface retrieves the coefficients of the models from the database for each trained ANN, and receives flight data from the flight simulator every 0.1 second. The Interface organizes the coefficients into sets of weights and biases, and organizes data received from the simulator into sets of inputs for each ANN. The relevant coefficients, and flight data input sets are then fed to the Flight Manager and the ANNs of the IAS to produce outputs. The outputs of the ANNs are sent to the Interface which sends them to the flight simulator as autonomous control commands using UDP packets every 0.1 second.

2) The Flight Manager Program

The Flight Manager is a program which resembles a Behaviour Tree [19]. The purpose of the Flight Manager is to manage the fourteen ANNs of the IAS by deciding which ANNs are to be used simultaneously at each moment. In addition, it generates a flight course to the destination airport of choice based on stored GPS waypoints as Fig. 5 illustrates, by applying (6) [20] to calculate the bearing (heading) between the GPS coordinates (latitude and longitude) of the waypoints.

$$\Phi = atan2(sin(\Delta\lambda)cos(\Phi_2), cos(\Phi_1)sin(\Phi_2)cos(\Delta\lambda)) \quad (6)$$

where Φ_1 is the start point, Φ_2 is the end point, and $\Delta\lambda$ is the difference in longitude.

The program constantly measures the deviation between the aircraft's position and the current path line of the flight course represented by the angle between the line that starts at the location point of the aircraft and ends at the location point of the next waypoint, and the line that starts at the location point of the previous waypoint and ends at the location point of the next waypoint as Fig. 6 illustrates.

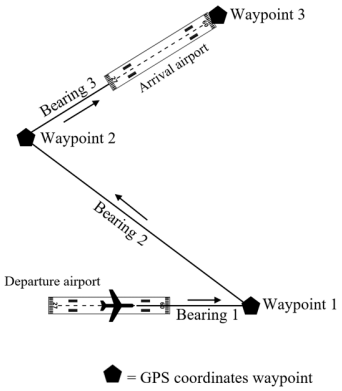

Fig. 5. A flight course from a departure airport to a landing airport consisting of three path lines and their bearings/headings, which connect the three pre-stored GPS coordinates waypoint.

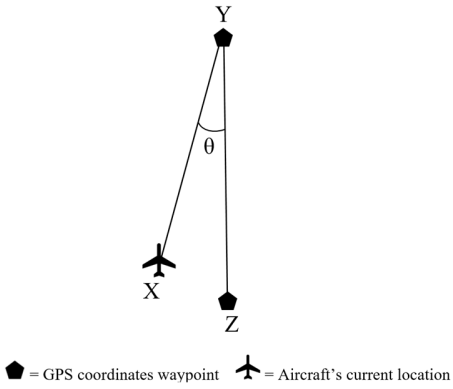

Fig. 6. The angle between the line from the aircraft's location X and the next waypoint Y, and the line from the previous waypoint Z and the next waypoint Y.

The Flight Manager calculates the difference between the bearing of the path line to be intercepted, and the aircraft's current bearing, then, it adds the angle to the difference. As the aircraft's current bearing becomes closer to the desired bearing, and as the angle becomes smaller, the difference becomes smaller as well, which leads to a gradual interception of the path line, and avoids undesired undershooting or overshooting as Fig. 7 illustrates.

The Top of Descent (TOD) is the point at which descending towards the destination airport is initiated. In this work, the Flight Manager calculates the TOD by multiplying the altitude by 0.003 [2]. If the result is less than the distance (in kilometers) to the landing runway, then the TOD is reached, and the descending process starts.

The Glideslope is an altitude slope of a given degree, which leads to a touchdown on the landing runway. The Flight Manager generates a virtual altitude slope by dividing the distance to the runway by 10. The latter method is used based on preliminary empirical testing. Fig. 8 illustrates how the Flight Manager generates the glideslope.

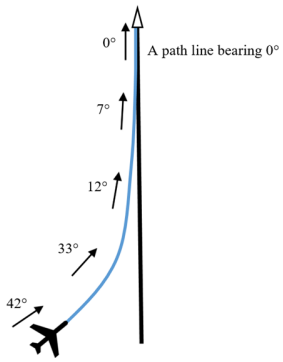

Fig. 7. An example illustrating how the Flight Manager updates the bearing to be followed based on the difference between the bearing of the path line to be intercepted, and the aircraft's bearing. The angle between the aircraft and the path line is added to the difference to ensure a gradual interception.

[2] How to compute the TOD (Top of Descent) - Thumb rule. https://community.infinite-flight.com/

The Flight Manager starts by receiving flight data from the flight simulator through the interface of the IAS, then it detects the flight condition and phase by examining the received flight data, and decides which ANNs are required to be used given the flight condition and phase.

The procedure used by the Flight Manager to handle emergency situations such as an engine fire/failure is discussed in our previous work [3]. Fig. 9 illustrates the process which the Flight Manager follows to handle the execution of complete flights.

3) Artificial Neural Networks

The relevant set of flight data inputs received through the Interface is used by the ANNs' input neurons along with the relevant coefficients to predict control commands given the flight status by applying (1) and (2). The values of the output layers are sent to the Interface which sends them to the flight simulator as autonomous control commands. The design approach of the ANNs intends to breakdown the different tasks required for flying, that take place during the multiple flight phases. For example, the ground-run phase requires the task of gaining takeoff speed by releasing brakes and going to full throttle, and the task of keeping the aircraft on the centerline of the runway using the ruder. For the latter tasks, two ANNs were designed, which control the brakes and throttle (task 1), and the rudder (task 2). To predict the appropriate control commands,

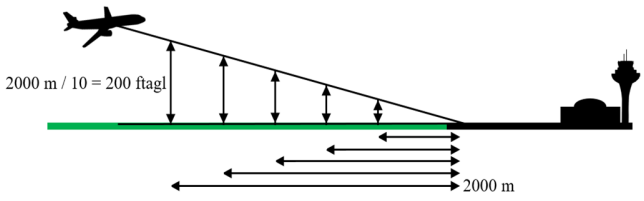

Fig. 8. The Glideslope generated by continuously calculating the altitude during the final approach descent which leads to a touchdown on the landing runway. The desired altitude is the distance to the runway divided by 10.

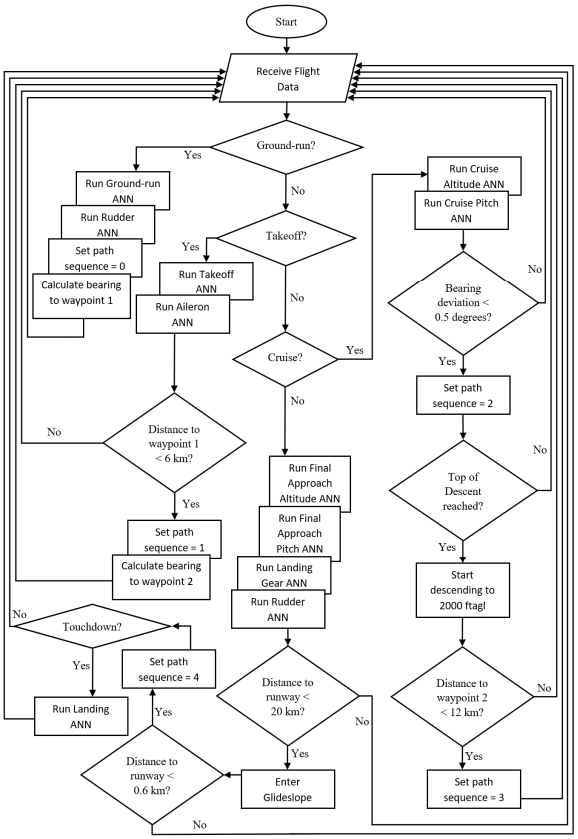

Fig. 9. A Flowchart illustrating the process which the Flight Manager program follows to decided which ANNs are to be used, and how to handle flight phases and navigation points transitions.

the ANNs rely on the relevant flight data inputs as Fig. 3 illustrates. Following the problem breakdown approach, it is possible to achieve a composition of small multiple control units represented by the task-dedicated ANNs that can be designed, integrated, and traced effortlessly compared to systems that rely on a single or few large ANNs designed to handle multiple task. In addition, when following the breakdown approach, it is possible to achieve higher levels of accuracy since each ANN is dedicated towards a single task of controls mapping as follows. Fig. 10 illustrates the ANNs used during the different flight phases.

IV. Experiments

This work discusses the experiments conducted on the modified Aileron ANN which can now bank, and intercept a path line, in addition to controlling the roll degree. This section also discusses the experiments conducted on the new ANNs that are used during the final approach, and landing phases.

The experiments were conducted under calm weather conditions with nil wind speed.

Our previous work [2], [3] provide detailed explanations of the experiments of autonomous ground-run, takeoff, climb, cruise, rudder control, maintaining a desired altitude and pitch, and handling emergency situations.

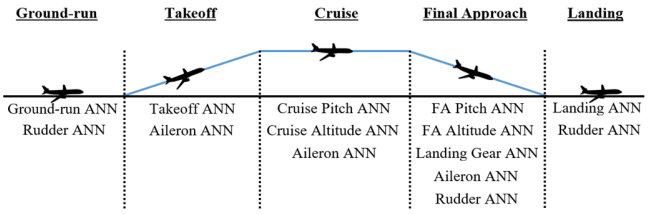

Fig. 10. The ANNs used during the different phases of the flight.

To assess the effectiveness of the proposed approach in this paper, the Intelligent Autopilot System was tested in three experiments: A. Banking turn and path line interception, B. Final approach, and C. Landing. Each experiment is composed of 50 attempts by the IAS to perform autonomously under the given conditions.

The human pilot who provided the demonstrations is the first author. The simulated aircraft used for the experiments is a Boeing 777 as we want to experiment using a complex and large model with more than one engine rather than a light single-engine model. The experiments are as follows:

A. *Banking turn and path line interception*

The purpose of this experiment is to assess the behaviour of the IAS compared to the behaviour of the human pilot when performing a banked turn, and to assess the path line interception technique.

1) Data Collection

In this experiment, the human pilot used the IAS Interface to change the aircraft's heading by performing a banked turn through maintaining a roll of 25 to 35 degrees. While the pilot performed the demonstration, the Interface collected roll and difference values as inputs, and aileron control value as output. The Interface stored the collected data in the database as the training dataset for the Aileron ANN.

2) Training

For this experiment, the Aileron ANN was trained until a low Mean Squared Error (MSE) value was achieved (below 0. 1).

When the aircraft is close to the path line to be intercepted, a large banking turn of 25 to 35 degrees of roll can cause the aircraft to constantly overshoot the path line instead of intercepting it smoothly. So, instead of training an additional Aileron ANN that performs banking turns through smaller degrees of roll, the same generated ANN model can be stimulated differently to alter its behaviour, by feeding its difference input neuron with difference values that are much smaller than the difference values present in the training dataset. The latter exploits the generalization effect which causes the model to behave differently based on the unseen inputs. To achieve this in this experiment, before feeding the difference input neuron of the Aileron ANN with the difference value, the difference is reduced to just 30% of its actual value, which was found through extensive preliminary experiments. This approach was tested between two waypoints represented by a straight path line.

3) Autonomous Control

After training the ANN on the relevant training dataset, the aircraft was reset to the runway in the flight simulator to test

autonomous banking turn and path line interception. After takeoff, when the Flight Manager updates the path sequence of the flight course, calculates the new heading, the angle, and the difference, the Aileron ANN performs a banked turn to minimize the difference, and eventually, intercept the path line gradually. Through the Interface, the ANN receives: 1. relevant flight data from the flight simulator as inputs, and 2. coefficients of the relevant models from the database to predict and output command controls that are sent to the flight simulator. This process allows the IAS to autonomously perform the learned task: autonomous banking turn and path line interception. This was repeated 50 times to assess performance consistency.

B. Final Approach

The purpose of this experiment is to assess the behaviour of the IAS compared to the behaviour of the human pilot during the final approach phase.

1) Data Collection

In this experiment, the human pilot used the IAS Interface to perform the following in the flight simulator: maintain a positive pitch of about 3 to 4 degrees during the final approach phase to decrease airspeed without causing a stall, and to ensure a flare immediately after touchdown, engage full flaps when the airspeed is less than 260 knots, and engage the landing gear when the altitude decreases to 1500 ftagl. The desired descent altitude is continuously updated by the Flight Manager as explained above in section C of part III, and the experiments conducted on the techniques followed by the ANNs to maintain a desired altitude, and a desired pitch are explained in our previous work [3]. For this work, while the pilot performed the demonstration, the Interface collected airspeed and altitude as inputs, and flaps as output. The Interface stored the collected data in the database as the training dataset for the Final Approach Altitude ANN. The Interface also collected altitude as input, and landing gear control data as output. The Interface stored the collected data in the database as the training dataset for the Landing Gear ANN.

2) Training

For this experiment, the Final Approach Altitude ANN, and the Landing Gear ANN were trained until low Mean Squared Error (MSE) values were achieved (below 0.01).

3) Autonomous Control

After training the ANNs on the relevant training datasets, the aircraft was reset to the runway in the flight simulator to test the autonomous final approach procedures. After entering the final approach flight phase, and when the desired airspeed is reached, the Final Approach Altitude ANN engages flaps, and when the desired altitude is reached, the Landing Gear ANN engages the landing gear. Through the Interface, the ANNs receives: 1. relevant flight data from the flight simulator as inputs, and 2. coefficients of the relevant models from the database to predict and output command controls that are sent to the flight simulator. This process allows the IAS to autonomously perform the learned final approach procedures. This was repeated 50 times to assess performance consistency.

C. Landing

The purpose of this experiment is to assess the behaviour of the IAS compared to the behaviour of the human pilot when performing landing procedures.

1) Data Collection

In this experiment, the human pilot used the IAS Interface to perform the landing procedures immediately after touchdown, by engaging reverse thrust, brakes, and speed brakes. While the pilot performed the demonstration, the Interface collected airspeed as input, and reverse thrust, brakes, and speed brakes control data as outputs. The Interface stored the collected data in the database as the training dataset for the Landing ANN.

2) Training

For this experiment, the Landing ANN was trained until low Mean Squared Error (MSE) values were achieved (below 0.01).

3) Autonomous Control

After training the ANN on the relevant training dataset, the aircraft was reset to the runway in the flight simulator to test the ability of performing the landing procedures autonomously, and the IAS was engaged. After the IAS took the aircraft airborne, navigated to the destination airport, and touched down, the system's ability to perform the landing procedures of engaging reverse thrust, brakes, and speed brakes was observed. Through the Interface, the ANN receives: 1. relevant flight data from the flight simulator as inputs, and 2. coefficients of the relevant models from the database to predict and output command controls that are sent to the flight simulator. This process allows the IAS to autonomously perform learned landing procedures. This was repeated 50 times to assess performance consistency.

V. Results

The following section describes the results of the conducted tests.

A. Banking turn and path line interception

One model was generated for the Aileron ANN with an MSE value of 0.0954. Fig. 11 illustrate a comparison between the human pilot and the IAS when performing a banked turn to change the aircraft's bearing by 145 degrees over a period of 40 seconds. Due to high consistency and the lack of wind, the lines representing the 50 attempts by the IAS overlap. The Mean Absolute Deviation (MAD) results of the roll degrees over time (5.02 for the IAS (average) and 4.34 for the human pilot) show a close behaviour between the system and its teacher. Fig. 12 illustrates the smaller roll degrees when intercepting a path line, after altering how the difference input neuron is stimulated, by reducing the input's value to just 30% of its actual value. Since the experiments were conducted on 6 different segments of the same path line while being intercepted under the same weather conditions (nil wind speed), and due to the high consistency of the IAS, 6 different sets of overlapping lines are visible as Fig. 12 illustrates.

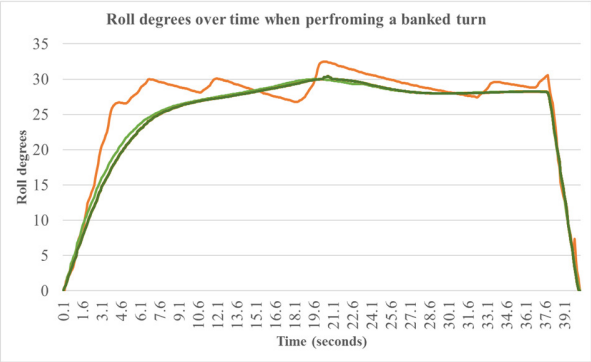

Fig. 11. (Banking turn and path line interception experiment). A comparison between the human pilot (orange line) and the 50 attempts by the IAS (overlapping lines) to perform a banked turn to change the aircraft's bearing by 145 degrees over a period of 40 seconds. The good fit of the generated model allowed the IAS to maintain a steadier and consistent change of roll degrees compared to the human pilot.

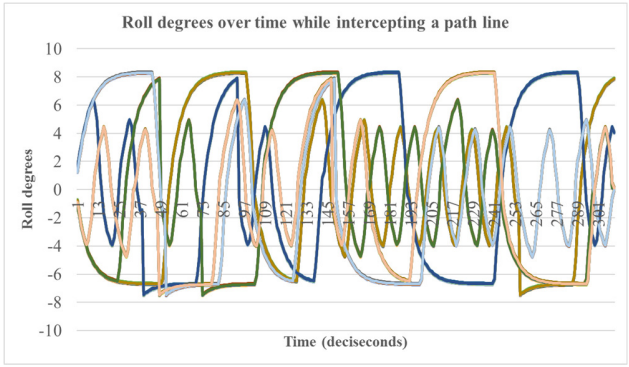

Fig. 12. (Banking turn and path line interception experiment). Smaller degrees of roll when banking continuously to intercept a path line. For the 50 attempts, the roll is below 9 degrees (suitable for small turns while intercepting) compared to a maximum of 31 (suitable for major bearing change) as Fig. 10 illustrates. Due to the consistency, and similarity of the testing scenarios (location and weather), most of the lines are overlapped.

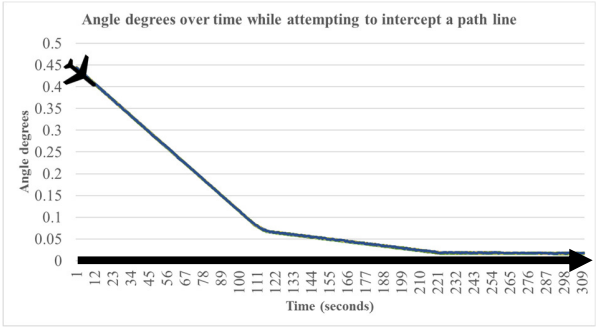

Fig. 13. (Banking turn and path line interception experiment). 50 attempts with strong consistency to gradually intercept and follow a path line (black arrow). The interception attempt is represented by the gradual decrease of the angle between the aircraft and the path line.

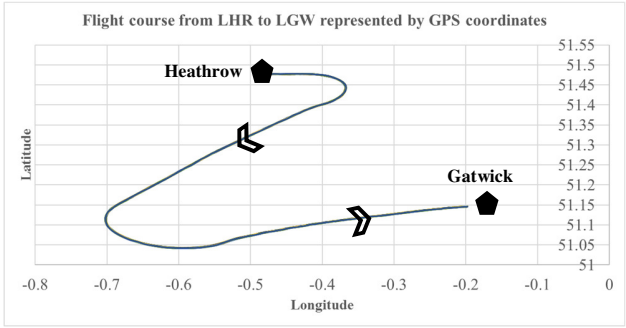

Fig. 14. (Banking turn and path line interception experiment). The 50 flight courses (overlapped lines) with strong consistency flown autonomously by the IAS, starting with takeoff from London Heathrow airport, and landing at Gatwick airport. Since the distance between the two airports is short for an airliner, the generated flight course accounts for the distance required to perform the final approach phase, and therefore, follows an initial path away from Gatwick.

Fig. 13 illustrates how applying the reduced degrees of roll when constantly banking (correcting bearing) to intercept a path line, ensure a steady and gradual interception. Fig. 14 illustrates the flight course that the IAS generated and followed autonomously. Due to high consistency and the lack of wind, the lines representing the 50 attempts by the IAS in Fig. 13 and 14 overlap

B. *Final Approach*

Two models were generated for this experiment, the Final Approach Altitude ANN model with an MSE value of 0.0034, and the Landing Gear ANN model with an MSE value of 0.0046. Fig. 15 illustrate a comparison between the human pilot and the IAS when extending the flaps after the appropriate airspeed is reached. Fig. 16 illustrates a comparison between the human pilot and the IAS when engaging the landing gear after the appropriate altitude is achieved. Due to high consistency, the lines representing the human behaviour, and the 50 attempts by the IAS in Fig. 15 and 16 overlap. Fig. 17 illustrates the glideslope followed during the final approach phase in 50 experiments resulting in lines that overlap due to high consistency and the lack of wind.

C. *Landing*

One model was generated for the Landing ANN with an MSE value of 0.003. Fig. 18 illustrates a comparison between the human pilot and the IAS when engaging the reverse thrust, brakes, and speed brakes immediately after touchdown. Due to high consistency, the lines representing the human behaviour, and the 50 attempts by the IAS in Fig. 18 overlap.

VI. Analysis

As can be seen in Fig. 11 (banking turn and path line interception experiment), the IAS was not only able to imitate the behaviour of its human teacher when performing a banked turn by maintaining a certain degree of roll, it was also able to perform better by being able to maintain a steadier change of roll degrees, which is due to the good fit of the generated learning model. The new method of changing the stimuli of the ANN to cause it to alter its behaviour instead of having to retrain it or generate a different learning model, provided excellent results. Fig. 12 (banking turn and path line interception experiment) shows how changing the stimuli represented by the

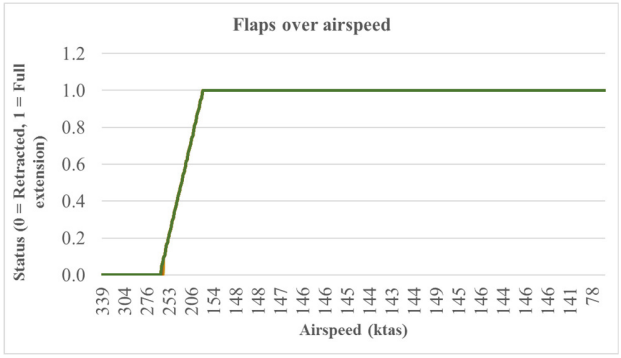

Fig. 15. (Final approach experiment). A comparison between the human pilot and 50 attempts by the IAS (overlapped lines) when extending flaps immediately after an airspeed of 260 (ktas) is reached. The behaviour of the human pilot and the IAS in the 50 attempts are similar with strong consistency.

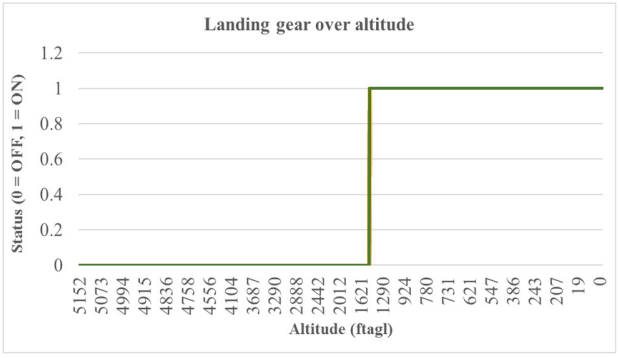

Fig. 16. (Final approach experiment). A comparison between the human pilot and 50 attempts by the IAS (overlapped lines) when engaging the landing gear immediately after an altitude of 1500 (ftagl) is reached. The behaviour of the human pilot and the IAS in the 50 attempts are similar with strong consistency.

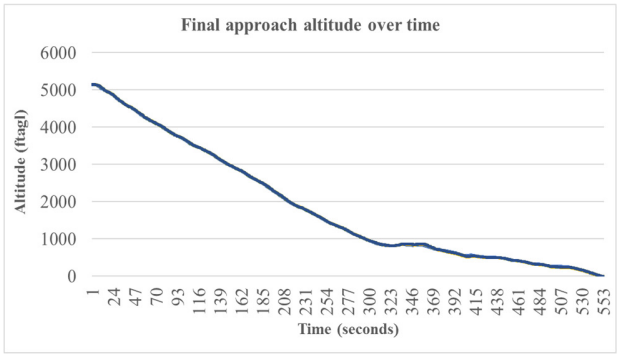

Fig. 17. (Final approach experiment). 50 attempts (overlapped lines) with strong consistency by the IAS to follow the final approach glideslope. The glideslope is adjusted by the IAS after descending to an altitude below 1000 (ftagl) to compensate for the additional drag generated by the landing gear.

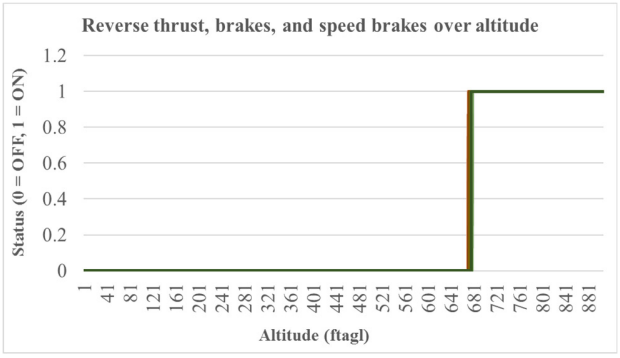

Fig. 18. (Landing experiment). A comparison between the human pilot and 50 attempts by the IAS (overlapped lines) when engaging reverse thrust, brakes, and speed brakes immediately after touchdown. The behaviour of the human pilot and the IAS in the 50 attempts are similar with strong consistency.

difference value which passes through the input neuron of the Aileron ANN, through reducing it by a given percentage, caused the ANN to behave differently. The latter can be seen as the much smaller degrees of roll maintained by the Aileron ANN although its generated learning model was trained to maintain larger degrees of roll. The smaller degrees of roll maintained when banking or correcting the aircraft's bearing to intercept a path line, allowed the IAS to gradually intercept and follow the path line while avoiding undershooting and overshooting as Fig. 13 (banking turn and path line interception experiment) shows, and therefore, the IAS was able to strictly follow the generated flight course with excellent accuracy and consistency as Fig. 14 (banking turn and path line interception experiment) shows, throughout all the experiments.

The IAS was capable of identically imitating the human pilot's actions and behaviour when performing the procedures of the final approach phase, by extending the flaps only when a certain airspeed is reached, and engaging the landing gear only when a certain altitude is reached as Fig. 15 and 16 (final approach experiment) show. The method covered in our previous work [3], which is followed by the ANNs that are responsible for maintaining a given altitude, proved to be adequate for handling a rapidly changing desired altitude which is continuously updated by the Flight Manager during the final approach phase. The latter generated a glideslope that led to a touchdown on the landing runway, and was maintained by the IAS. However, as soon as the extra drag caused by the extracted landing gear generated a larger sink rate which could cause a premature touchdown (touching down before reaching the landing runway), the IAS was able to autonomously alter the glideslope by following a less steep degree towards the runway as Fig. 17 (final approach experiment) shows.

As can be seen in Fig. 18 (landing experiment), the IAS was capable of identically imitating the human pilot's actions and behaviour when performing the procedures of the landing phase, by engaging the reverse thrust, brakes, and speed brakes immediately after touchdown to bring the aircraft to a rapid full stop.

VII. Conclusion

In this work, a novel and robust approach is proposed to "teach" autopilots how to perform complete flights from takeoff to landing with minimum effort by exploiting Learning by Imitation also known as Learning from Demonstration. This approach introduces the possibility to have an autopilot that

behaves like a skilled human pilot rather than a machine with limited capabilities.

The experiments were strong indicators towards the ability of Supervised Learning with Artificial Neural Networks to capture low-level piloting tasks such as the rapid manipulation of the ailerons to maintain a banked turn, and high-level tasks such as coordinating the necessary actions during the final approach and the landing phases.

Breaking down the piloting tasks, and adding more Artificial Neural Networks allows the system to overcome the black-box problem by having multiple small ANNs with single hidden layers that learn from small labelled datasets which have clear patterns. In addition, this approach enhanced performance and accuracy, and allowed the coverage of a wider spectrum of tasks.

The aviation industry is currently working on solutions which should lead to decreasing the dependence on crew members. The reason behind this is to lower workload, human error, stress, and emergency situations where the captain or the first officer becomes incapable, by developing autopilots capable of handling multiple scenarios without human intervention. We anticipate that future Autopilot systems which make of methods proposed here could improve safety and save lives.

VIII. Future Work

Our work [21] covers navigation, landing, and go-around under severe weather conditions with the presence of high crosswind component, wind shear, gust, and turbulence.

Since this work and our other work [1] [2] [21] proved the possibility of teaching an artificially intelligent autopilot how to perform complete flights while being able to handle multiple uncertainties, and since the performance of the Intelligent Autopilot System (IAS) is based on what it learned from its teacher (the first author) who has no real flying experience and know-how, we believe it is time to take this work to the next level which we anticipate would cover teaching the IAS by allowing it to observe new demonstrations from experienced pilots using professional equipment such as CAA-certified flight simulators. Next, we anticipate testing the IAS thoroughly in real-life scenarios by integrating it with a fixed-wing Unmanned Aircraft System (UAS) before collaborating with the civil aviation industry.

References

[1] R. Khan-Persaud, "ECCAIRS Aviation 1.3.0.12 (VL for ATTrID 391 - Event Phases)", *ICAO Safety*, 2013. [Online]. The International Civil Aviation Organization (ICAO). Available: https://www.icao.int/safety/airnavigation/AIG/Documents.

[2] H. Baomar and P. J. Bentley, "An Intelligent Autopilot System that learns piloting skills from human pilots by imitation," *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, Arlington, VA, USA, 2016, pp. 1023-1031.

[3] H. Baomar and P. J. Bentley, "An Intelligent Autopilot System that learns flight emergency procedures by imitating human pilots," *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, Athens, 2016, pp. 1-9.

[4] J. R. G. Braga, H. F. C. Velho, G. Conte, P. Doherty and É. H. Shiguemori, "An image matching system for autonomous UAV navigation based on neural network," *2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, Phuket, 2016, pp. 1-6.

[5] P. Oettershagen, T. Stastny, T. Mantel, A. Melzer, K. Rudin, P. Gohl, G. Agamennoni, K. Alexis, and R. Siegwart, "Long-Endurance Sensing and Mapping using a Hand-Launchable Solar-Powered UAV," *Springer Tracts in Advanced Robotics*, Volume 113, Springer International Publishing, 2016, pp. 441-454.

[6] S. Lin, and BW. Kernighan, "An effective heuristic algorithm for the travelingsalesman problem". Operations research, 1973, 21(2):498–516

[7] S. Karaman, and E. Frazzoli, Incremental sampling-based algorithms for optimal motion planning. 2010, CoRR abs/1005.0416.

[8] L. Ospina, N. Abdullah, O. A. Jahdali and O. Oteniya, "Design of a cruise control system prototype for the GT500's airplane," *2017 Systems and Information Engineering Design Symposium (SIEDS)*, Charlottesville, VA, USA, 2017, pp. 79-83.

[9] L. Duanzhang, C. Peng and C. Nong, "A generic trajectory prediction and smoothing algorithm for flight management system," *2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC)*, Nanjing, 2016, pp. 1969-1974.

[10] M. S. I. Khan, M. Belal Tiasha and S. Barman, "Auto landing sequence for an unmanned aerial vehicle at a fixed point," *2017 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, Cox's Bazar, 2017, pp. 175-180.

[11] D. Tang, Y. Jiao, and J. Chen, "On Automatic Landing System for carrier plane based on integration of INS, GPS and vision," *2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC)*, Nanjing, 2016, pp. 2260-2264.

[12] E. Atkins, "Safe Autonomous Manned and Unmanned Flight in Off-Nominal Conditions", [Presentation]. *Future Technologies Conference*, San Francisco, USA, 2016.

[13] P. Salmon, G. Walker, and N. Stanton, "Pilot error versus sociotechnical systems failure: a distributed situation awareness analysis of Air France 447". Theoretical Issues in Ergonomics Science, 2015, pp.64-79.

[14] F. Wei, A. Bower, L., Gates, A., Rose, and D. T. Vasko, "The Full-Scale Helicopter Flight Simulator Design and Fabrication at CCSU", *57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2016.

[15] M. Jirgl, J. Boril, and R. Jalovecky, "The identification possibilities of the measured parameters of an aircraft model and pilot behavior model on the flight simulator". *International Conference on Military Technologies (ICMT)*, 2015, vol., no., pp.1-5.

[16] A. Kaviyarasu, and S. Kumar, "Simulation of Flapping-wing Unmanned Aerial Vehicle using X-plane and Matlab/Simulink". "*Defence Science Journal"*, 2014, 64(4), pp.327-331.

[17] J. McCaffrey, "Understanding Neural Networks using .NET", [Presentation]. *The Microsoft 2014 Build Conference*, San Francisco, USA, 2014.

[18] J. Heaton, *Artificial Intelligence for Humans, Volume 3: Deep Learning and Neural Networks*. St. Louis, MO, USA: Heaton Research, Inc., 2015.

[19] K. Winter, I. J. Hayes, and R. Colvin, "Integrating Requirements: The Behavior Tree Philosophy," 2010 8th IEEE International Conference on Software Engineering and Formal Methods, Pisa, 2010, pp. 41-50.

[20] J. M., "Calculating a bearing between points in location-aware apps", *Intel Software*, 2012. [Online]. Available: https://software.intel.com/en-us/blogs/2012/11/30/calculating-a-bearing-between-points-in-location-aware-apps.

[21] H. Baomar and P. J. Bentley, "Autonomous Landing and Go-around of Airliners Under Severe Weather Conditions Using Artificial Neural Networks," *The 2017 International Workshop on Research, Education and Development on Unmanned Aerial Systems (RED-UAS)*, Linköping, Sweden, 2017. (Submitted)

Autonomous Landing and Go-around of Airliners Under Severe Weather Conditions Using Artificial Neural Networks

Haitham Baomar, Peter J. Bentley

Dept. of Computer Science, University College London, Gower Street, London, WClE 6BT, U.K.
Email: {h.baomar, p.bentley} @ cs.ucl.ac.uk

Abstract— We introduce the Intelligent Autopilot System (IAS) which is capable of autonomous landing, and go-around of large jets such as airliners under severe weather conditions. The IAS is a potential solution to the current problem of Automatic Flight Control Systems of being unable to autonomously handle flight uncertainties such as severe weather conditions, autonomous complete flights, and go-around. A robust approach to control the aircraft's bearing using Artificial Neural Networks is proposed. An Artificial Neural Network predicts the appropriate bearing to be followed given the drift from the path line to be intercepted. In addition, the capabilities of the Flight Manager of the IAS are extended to detect unsafe landing attempts, and generate a go-around flight course. Experiments show that the IAS can handle such flight skills and tasks effectively, and can even land aircraft under severe weather conditions that are beyond the maximum demonstrated landing of the aircraft model used in this work as reported by the manufacturer's operations limitations. The proposed IAS is a novel approach towards achieving full control autonomy of large jets using ANN models that match the skills and abilities of experienced human pilots.

I. Introduction

Human pilots are trained to perform piloting tasks that are required during the different phases of the flight. They are trained to perform landing under difficult weather conditions such as strong crosswind, and abort landing by executing a go-around if needed.

In contrast, Automatic Flight Control Systems (AFCS/Autopilot) are highly limited, capable of performing minimal piloting tasks. Although modern autopilots can perform auto-land, they cannot handle complete flight cycles automatically, they must be engaged and operated manually by the human pilots to constantly change and update the desired parameters, and they cannot handle severe weather conditions, such as strong crosswind components combined with wind shear, gust, and turbulence. The reason for such limitations of conventional AFCS is that it is not feasible to anticipate everything that could go wrong with a flight, and incorporate all of that into the set of rules or control models "hardcoded" in an AFCS.

This work aims to address this problem by creating an Intelligent Autopilot System (IAS) with the capability to handle landing, and go-around under severe weather conditions using Artificial Neural Networks. The IAS is a novel approach which introduces the possibility to transfer human intelligence and intuitions required to pilot an aircraft under such conditions, to an autonomous system. By using this approach, we aim to extend the capabilities of modern autopilots and enable them to autonomously adapt their piloting to suit multiple scenarios ranging from normal to emergency situations. This work builds on previous work by the authors [1][2][3] which introduced the ability to follow a flight course and land autonomously under calm conditions, however, this approach was not able to handle landing under severe weather conditions. Therefore, this paper provides a new approach to enable the system to cope under such difficult conditions, or to safely abort when impossible to land.

This paper is structured as follows: part (II) reviews related literature on wind effects during the cruise, and landing flight phases. Part (III) explains the Intelligent Autopilot System (IAS). Part (IV) describes the experiments, Part (V) describes the results by observing the behaviour of the Intelligent Autopilot System in a flight simulator, and part (VI) provides an analysis of the results. Finally, we provide conclusions.

II. Background

A. Wind Effects on Autonomous Flying

Wind disturbance causes the UAV to drift from the desired course, and when added to the accumulated errors of the navigation systems, maintaining a desired flight path or course becomes a significant challenge [4][5].

In [6], the physical properties of the Vehicle Dynamic Model (VDM) are used to study the effects of wind on navigation systems in addition to the control inputs within the algorithm of the navigation filter. In [7], an approach to tackle strong wind effects during flights is proposed by estimating wind effects that are steady and strong in nature, and delivers a maneuvering strategy to tackle such conditions [7].

B. Crosswind Landing

To tackle crosswind during an approach, two methods are used, the first method is known as Crabbing where a certain

degree of drift or crab is induced to change the orientation of the aircraft's nose heading towards the direction of the wind [8]. The second method is known as Wing-down, in which a steady sideslip is induced to tackle the drift caused by the crosswind [8]. In practice, it is common to combine both methods, following degrees which could vary during the approach phase [9]. For the Boeing 777 of which a simulated model is used in this work, the maximum crosswind components are 45 knots for a dry runway, and 40 knots for a wet runway [8].

Artificial Neural Networks (ANNs) were used to estimate a mapping relationship between the given situation, and the human pilot inputs while performing the crabbing maneuver [10] [11]. In addition, the possibility of using conventional Control Theory fault tolerance techniques, that are used for Proportional Integral Derivative (PID) controllers to tackle the crosswind landing challenge, is being investigated such as applying the Integral Windup handling methods [12].

Introducing intelligent autonomy to the aviation industry through developing intelligent control techniques that fit into an overall flight management system capable of making the highest level of decisions, is expected to significantly enhance safety, and lower costs [13].

In addition of having limited capabilities, modern autopilots can contribute to catastrophes since they can only operate under certain conditions that fit their design and programming, otherwise, they cede control to the pilots, and with the lack of proper situational awareness and reaction, the result could be fatal [14]. Although the civil aviation sector that uses medium to large jets equipped with such autopilots, is the largest with the highest risk and costs, the current focus of the relevant and recent research efforts is on investigating and developing autonomous autopilots for Unmanned Aircraft Systems especially small and micro drones by introducing solutions that may not be suitable for Large jets such as airliners. Therefore, we propose a solution that can be applied to multiple aircraft categories including airliners and cargo airplanes. We believe that manned aircraft especially airliners require significant attention to enhance safety by addressing the limitations of modern autopilots and flight management systems, and the human error factor as well. A review of the Autopilot problem, Artificial Neural Networks, autonomous navigation and landing are presented in our previous work [1][3].

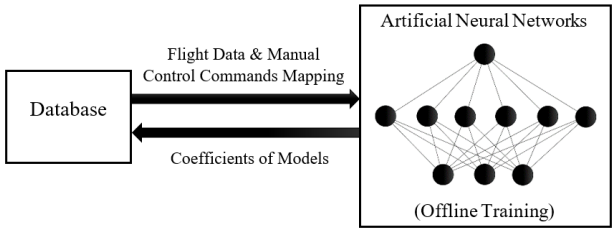

Fig. 1. Block diagram illustrating the IAS components used during training.

III. The Intelligent Autopilot System

The IAS is made of the following components: a flight simulator, an interface, a database, a flight manager program, and Artificial Neural Networks. The IAS implementation method has three steps that start with pilot data collection which is a process [1][2][3] that records human pilot demonstrations in a flight simulator of the piloting tasks to be learned by the IAS. The recorded demonstrations are transformed into training datasets for the ANNs.

In this paper, we discuss: A. Training, and B. Autonomous Control. In each step, different IAS components are used. The following sections describe each step and the components used in turn. The approach applied to allow the IAS to learn from human teachers is covered in our previous work [1][2][3].

A. Training

1) Artificial Neural Networks

Artificial Neural Networks are used to generate learning models from the captured datasets through offline training. Fig. 1 illustrates the training step.

Fourteen feedforward Artificial Neural Networks comprise the core of the IAS. Each ANN is designed and trained to handle specific controls and tasks by taking flight data as inputs, and producing control commands as outputs. Fig. 2 illustrates the main ANNs used during the different phases of the flight. The fourteen ANNs including the emergency situations ANNs are discussed in [1][2][3]. In this work, we introduce the Bearing Adjustment ANN as Fig. 3 illustrates.

The method for choosing ANN topologies in this work is based on an implication [15] which indicates that direct mapping problems requiring more than one hidden layer are rarely encountered, and compared to Deep Learning, this approach means that the system is more understandable and easier to test and verify compared to single deep solutions which are black-boxes unsuited for safety critical applications.

Before training, the dataset is retrieved from the database. Then, the dataset is fed to the ANN. Next, supervised feedforward training using the Hyperbolic Tangent (Tanh) function [16] which is selected given its ability to handle negative values, and the Backpropagation algorithm [16] are applied to train the ANNs.

When training is completed, the learning model is generated, and the free parameters or coefficients represented by weights and biases of the model are stored in the database.

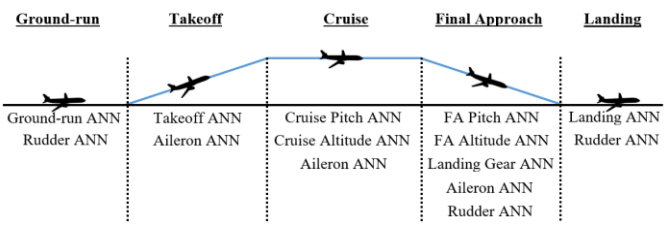

Fig. 2. The ANNs used during the different phases of the flight.

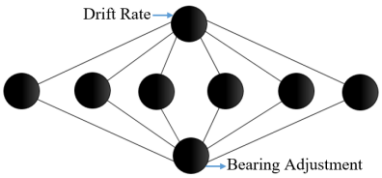

Fig. 3. Input, output, and the topology of the Bearing Adjustment ANN.

2) Database

An SQL Server database stores the free parameters or coefficients represented by weights and biases of the generated learning models.

B. Autonomous Control

Once trained, the IAS can now be used for autonomous control. Fig. 4 illustrates the components used during the autonomous control step.

1) The Flight Simulator

The simulator of choice in this work is X-Plane 10 which is an advanced flight simulator that has been used in many research papers such as [17] [18] [19].

2) The IAS Interface

The IAS Interface is responsible for data flow between the flight simulator and the system in both directions over the network using UDP packets. Here, the Interface retrieves the coefficients of the models from the database for each trained ANN, and receives flight data from the flight simulator every 0.1 second. The Interface organizes the coefficients into sets of weights and biases, and organizes data received from the simulator into sets of inputs for each ANN. The relevant coefficients, and flight data input sets are then fed to the Flight Manager and the ANNs of the IAS to produce outputs. The outputs of the ANNs are sent to the Interface which sends them to the flight simulator as autonomous control commands using UDP packets every 0.1 second.

3) The Flight Manager Program

The Flight Manager is a program which resembles a Behaviour Tree [20]. The purpose of the Flight Manager is to manage the ANNs of the IAS by deciding which ANNs are to be used simultaneously at each moment. In addition, it generates a flight course to the destination airport of choice based on stored GPS waypoints.

The go-around maneuver is performed to abort landing, by going to takeoff thrust levels, pulling up to climb, and retracting the landing gear. This is performed when the pilot decides that proceeding with landing might be unsafe, and therefore, it is favorable to climb, go around through a given flight course which brings the aircraft back to the point that precedes the final approach phase, and reattempt landing.

Landing safety check techniques are used to ensure that the aircraft is within safe landing conditions, otherwise, go-around is initiated. These techniques, such as the Runway Overrun Prevention System (ROPS)[1] from Airbus, analyze

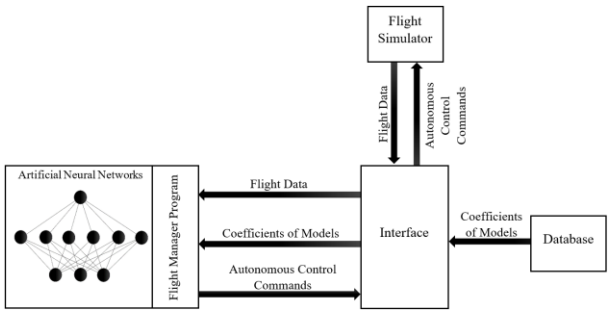

Fig. 4. Block diagram illustrating the IAS components used during autonomous control.

multiple parameters continuously including the available landing runway data and condition to ensure safe landing.

During final approach and just before touchdown, and at a specific altitude that ensures the possibility for the aircraft to climb safely before touchdown, the Flight Manager of the IAS initiates the continuous landing safety check. The selected altitude at which this process starts is equal to or slightly greater than 60 (ftagl) based on preliminary empirical testing. First, the Flight Manager checks if the angle between the aircraft and the centerline of the landing runway is less than a specific degree based on the runway's width. Then, it checks if the beginning of the landing runway has been reached. Finally, it checks if the remaining distance to the end of the runway is safe for landing. The parameters used during this checking process can be modified based on the available information about the landing runway such as its width and length. If the Flight Manager detects an unsafe landing, it generates a go-around flight course based on the available GPS coordinates as Fig. 5 illustrates, changes the flight status from final approach to takeoff, and activates the takeoff ANN.

Fig. 6 illustrates the process which the Flight Manager follows to handle the go-around process. The methods used by the Flight Manager to handle the different tasks including generating flight courses, managing flights, and handling emergency situations is discussed in [2][3].

4) Artificial Neural Networks

The flight data input received through the Interface is used by the ANNs' input neuron along with the relevant coefficients to predict the appropriate output. The Interface sends the relevant output layer value to the flight simulator as autonomous control command.

Since this work aims to expand the capabilities of the IAS to handle landing under severe weather condition including

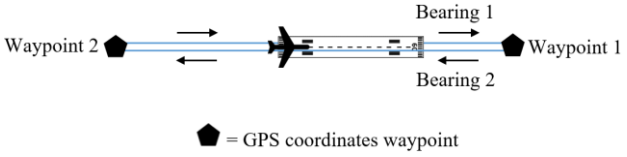

Fig. 5. The generated go-around flight course represented by the blue lines. The aircraft navigates to waypoint 1, then to waypoint 2, and finally, back to the landing runway.

[1] Airbus ROPS. http://www.aircraft.airbus.com/support-services/services/flight-operations/fuel-efficiency-and-runway-overrun-protection-systems/

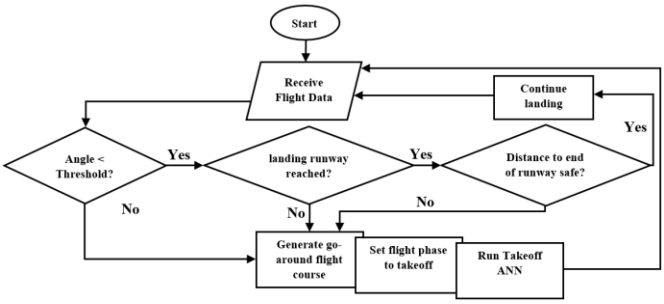

Fig. 6. A Flowchart illustrating the process which the Flight Manager program follows to check the landing conditions, and initiate a go-around if necessary.

strong crosswind components, wind shear, gust, and turbulence, the Bearing Adjustment ANN is introduced to predict the necessary adjustment of the aircraft's bearing based on the drift rate either towards or away from the path line to be intercepted. Based on preliminary empirical testing, the desired drift rate towards the path line is 0.0025 degrees every decisecond. First, the average rate of change of the angle -between the aircraft and the path line to be intercepted- is calculated using (1) [21].

$$A(x) = \frac{f(x) - f(a)}{x - a} \quad (1)$$

where $x - a$ is the change in the input of the function f, and $f(x) - f(a)$ is the change in the function f as the input changes from a to x. Then, the result is added to the difference between the bearing of the path line to be intercepted, and the aircraft's current bearing to generate the required bearing to be followed. The difference between the latter and the current bearing of the aircraft is fed to the Aileron ANN [1][3] which takes the difference as input, and predicts through its output neuron, the appropriate control command to the ailerons, to bank, and intercept the path line.

IV. Experiments

This work discusses the experiments conducted on the Bearing Adjustment ANN which aids the Aileron ANN to intercept a path line under severe weather conditions. This section also discusses the experiments conducted on performing go-around.

The experiments were conducted under severe weather conditions with the presence of high crosswind component, wind shear, gust, and turbulence.

Our previous work [1][2][3] provide detailed explanations of the experiments of autonomous ground-run, takeoff, climb, cruise, rudder control, maintaining a desired altitude and pitch, navigating from departure to arrival airports, landing, and handling emergency situations.

To assess the effectiveness of the proposed approach in this paper, the Intelligent Autopilot System was tested in two experiments: A. Path line interception during final approach, and B. Go-around.

The simulated aircraft used for the experiments is a Boeing 777 as we want to experiment using a complex and large model with more than one engine rather than a light single-engine model. The experiments are as follows:

A. Path line interception during final approach

The purpose of this experiment is to assess the behaviour of the IAS when intercepting a path line that represents the centerline of the landing runway during final approach under severe weather conditions.

1) Training

For this experiment, the Bearing Adjustment ANN was trained until a low Mean Squared Error (MSE) value was achieved (below 0. 01).

2) Autonomous Control

After training the ANN on the relevant training dataset, the aircraft was reset to the runway in the flight simulator, and the IAS was engaged to test the ability of intercepting a final approach and landing path line under severe weather conditions autonomously. After the IAS took the aircraft airborne, and navigated to the destination airport, the output of the Bearing Adjustment ANN was used to assist the Aileron ANN to intercept the final approach path line. This was repeated 50 times under different and random weather conditions as table I shows, to assess consistency. The weather conditions included a 0.015 turbulence value, and rain precipitation around 0.3 mm during all attempts.

B. Go-around

The purpose of this experiment is to assess the behaviour of the IAS when performing go-around autonomously.

1) Training

For this experiment, the same approach [3] used to navigate autonomously from a given point A to a given point B is applied. Therefore, no additional training was required.

2) Autonomous Control

The aircraft was reset to the runway in the flight simulator to test the autonomous go-around task. Just before touchdown, deviation from the path line is induced manually by stopping the IAS, and manually engaging the ailerons by the human pilot to deviate from the path line. Then, the IAS is started immediately. This approach was applied since the IAS excelled at landing within the safe zone of the landing runway regardless of how severe the weather conditions as long as these conditions are not exaggerated to a no-fly condition. To assess consistency, this was repeated 10 times under different and random weather conditions with minimum wind speed of 20 knots up to 35 knots, and random directions between 0 and 360 degrees including shear of 20

TABLE I
THE DIFFERENT WEATHER CONDITIONS USED FOR THE FINAL APPROACH PATH LINE INTERCEPTION EXPERIMENT.

Attempts Count	Wind Speed (knots)	Wind Gust (knots)	Wind Direction (degrees)	Wind Shear (degrees)
10	20	12	0	20
10	23	14	180	20
10	27	15	90	22
10	27	15	270	22
10	50	0	90	0

degrees. The weather conditions included a 0.015 turbulence value, and rain precipitation around 0.3 mm during all attempts.

V. Results

The following section describes the results of the conducted tests.

A. Path line interception during final approach

One model was generated for the Bearing Adjustment ANN with an MSE value of 0.0089. Utilizing the output value of the Bearing Adjustment ANN to enhance the path line interception performance, resulted in the system flying the aircraft using a technique known as crabbing, where although the aircraft flies in a straight line, the nose of the aircraft is pointed towards a bearing different from the bearing of the landing runway's centerline due to wind conditions. Unlike other systems where this technique must be explicitly hard-coded, here, the IAS naturally discovered the technique itself.

Fig. 7 illustrates the different bearings the IAS followed under random severe weather conditions as table I shows, compared to the bearing of the landing runway (326 degrees), where the lines in the upper area represent bearings followed when the aircraft was pushed to the left side of the landing runway's centerline, which happens in the presence of east crosswind for example, and vice versa. The lines on top are the bearings the IAS followed under a sustained weather condition with a constant crosswind of 50 knots at 90 degrees. Fig. 8 illustrates the average rate of change of the angle when drifting towards the path line. Fig. 9 illustrates the angle representing the difference between the aircraft's position, and the centerline of the landing runway. Based on the width of the landing runway used in the experiments, a safe touchdown angle is between 0.045 and -0.045 which was found based on preliminary empirical testing.

B. Go-around

No new models were generated for this experiment. Fig. 10 illustrates the flight paths that the IAS followed autonomously back to the landing runway. Since no strict go-around path was applied, the IAS followed two different paths based on the aircraft's location with respect to the landing runway's centerline, where a position on the right of the runway due to wind blowing from the left would cause the IAS to bank right towards the next waypoint, and vice versa.

VI. Analysis

As can be seen in Fig. 7 (Path line interception during final approach experiment), the IAS was able to produce a natural crabbing behaviour in a direction that is perpendicular to the

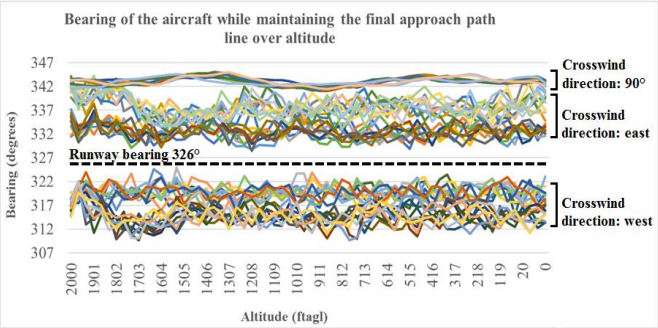

Fig. 7. 50 attempts showing Aircraft bearings (crabbing) during final approach under random severe weather conditions at table I shows, compared to the bearing of the landing runway (326 degrees). Lines in the upper area are bearings followed when the aircraft was pushed to the left side of the landing runway's centerline, and vice versa.

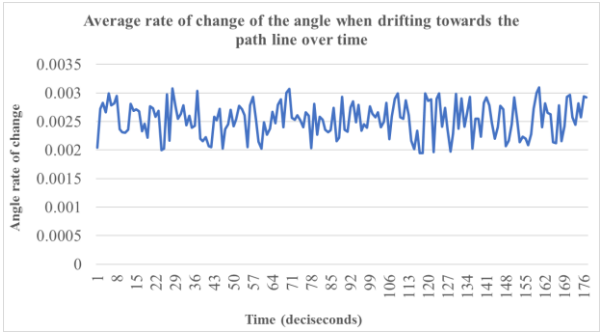

Fig. 8. The average rate of change of the angle when drifting towards the path line in the presence of random and severe weather conditions at table I shows, compared to a desired rate of change of 0.0025 degrees every decisecond.

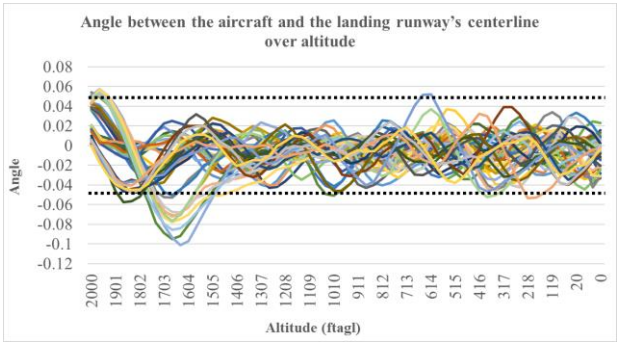

Fig. 9. 50 lines showing angle values between the aircraft's position, and the centerline of the landing runway (0 degrees) of all the attempts illustrated in Fig. 6. Based on the width of the landing runway used in the experiments, a safe touchdown angle is between 0.045 and -0.045, which is the area between the dotted lines (landing runway's safe touchdown zone).

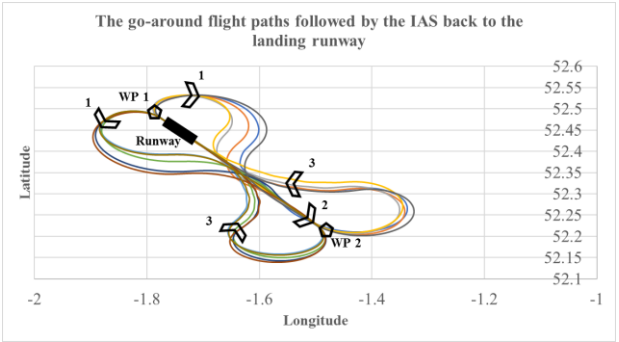

Fig. 10. The 10 go-around flight paths followed autonomously by the IAS back to the landing runway. The aircraft navigates to waypoint 1, then to waypoint 2, and finally, back to the landing runway. the IAS followed two different paths based on the aircraft's location with respect to the landing runway's centerline. Birmingham airport (BHX) was used.

constantly changing speed and direction of wind without being explicitly trained to do so. In addition, the IAS was able to handle persistent strong crosswind of 50 knots at 90 degrees which is beyond the demonstrated crosswind landing of a Boeing 777 as the top lines in Fig. 7 show. Keeping the angle rate of change close to 0.0025 degrees despite the random severe weather conditions proved the effectiveness of the Bearing Adjustment ANN as Fig. 8 (Path line interception during final approach experiment) illustrates. In all the attempts, the IAS was able to touchdown within the safe landing zone with respect to the centerline of the runway as Fig. 9 (Path line interception during final approach experiment) illustrates. This compares extremely well with the previous version of the IAS without the Bearing Adjustment ANN, which was unable to land under the same conditions. Under most weather conditions the IAS piloted so well that go-arounds were not needed, therefore, manual intervention was required to induce a go-around maneuver by stopping the IAS just before touchdown, manually banking the aircraft away from the centerline, then restarting the IAS. The system was able to detect unsafe landings through the Flight Manager, and followed go-around paths back to the landing runway under random severe weather conditions successfully as Fig. 10 (go-around experiment) illustrates.

VII. Conclusion

In this work, a novel and robust approach is proposed to perform autonomous final approach path line interception, and go-around under severe weather conditions.

The experiments were strong indicators towards the ability of Supervised Learning with Artificial Neural Networks to capture low-level piloting tasks such as the rapid manipulation of the ailerons to intercept a path line under severe weather conditions.

The novelties presented in our work, and dedicated to introducing intelligent autonomy to large jets such as airliners are robust solutions that could enhance flight safety in the civil aviation domain. They provide solutions to the difficult problem of autonomous navigation and landing under severe wind disturbance by enabling autonomous behaviour that was not possible before.

The aviation industry is currently working on solutions which should lead to decreasing the dependence on crew members. The reason behind this is to lower workload, human error, stress, and emergency situations where the captain or the first officer becomes incapable, by developing autopilots capable of handling multiple scenarios without human intervention. We anticipate that future Autopilot systems which make of methods proposed here could improve safety and save lives.

References

[1] H. Baomar and P. J. Bentley, "An Intelligent Autopilot System that learns piloting skills from human pilots by imitation," *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, Arlington, VA, USA, 2016, pp. 1023-1031.

[2] H. Baomar and P. J. Bentley, "An Intelligent Autopilot System that learns flight emergency procedures by imitating human pilots," *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, Athens, 2016, pp. 1-9.

[3] H. Baomar and P. J. Bentley, "Autonomous Navigation and Landing of Airliners Using Artificial Neural Networks and Learning by Imitation," *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, Hawaii, 2017. (submitted)

[4] H. Xiong, R. Yuan, J. Yi, G. Fan and F. Jing, "Disturbance Rejection in UAV's velocity and attitude control: Problems and solutions," *Proceedings of the 30th Chinese Control Conference*, Yantai, 2011, pp. 6293-6298.

[5] J. Smith, J. Su, C. Liu and W. Chen, "Disturbance Observer Based Control with Anti-Windup Applied to a Small Fixed Wing UAV for Disturbance Rejection", *Journal of Intelligent & Robotic Systems*, 2017.

[6] M. Khaghani and J. Skaloud, "Evaluation of Wind Effects on UAV Autonomous Navigation Based on Vehicle Dynamic Model", *Proceedings of the 29th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2016)*, pp. 1432 - 1440, Portland, Oregon, USA, 2016.

[7] S. Park, "Autonomous crabbing by estimating wind using only GPS velocity," in *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 3, pp. 1399-1407, June 2016.

[8] G. van Es, P. van der Geest and T. Nieuwpoort, "Safety aspects of aircraft operations in crosswind", National Aerospace Laboratory NLR, Amsterdam, Netherlands, 2001.

[9] G. van Es et. al., "Safety Aspects of Aircraft Performance on Wet and Contaminated Runways", *the 10th annual European Aviation Safety Seminar (FSF)*, Amsterdam, March 16-18, 1998.

[10] R. Mori and S. Suzuki, "Analysis of Pilot Maneuver under Crosswind Condition using Neural Network," Mathematical Problems in Engineering, Aerospace and Sciences, June 25-27, 2008.

[11] R. Mori and S. Suzuki, "Analysis of pilot landing control in crosswind using neural networks," *2009 IEEE Aerospace conference*, Big Sky, MT, 2009, pp. 1-10.

[12] S. Ismail, A. A. Pashilkar and R. Ayyagari, "Phase compensation and anti-windup design for neural-aided sliding mode fault-tolerant autoland controller," *2015 International Conference on Cognitive Computing and Information Processing(CCIP)*, Noida, 2015, pp. 1-7.

[13] P. Salmon, G. Walker, and N. Stanton, "Pilot error versus sociotechnical systems failure: a distributed situation awareness analysis of Air France 447". Theoretical Issues in Ergonomics Science, 2015, pp.64-79.

[14] E. Atkins, "Safe Autonomous Manned and Unmanned Flight in Off-Nominal Conditions", [Presentation]. *Future Technologies Conference*, San Francisco, USA, 2016.

[15] J. McCaffrey, "Understanding Neural Networks using .NET", [Presentation]. The Microsoft 2014 Build Conference, San Francisco, USA, 2014.

[16] J. Heaton, *Artificial Intelligence for Humans, Volume 3: Deep Learning and Neural Networks*. St. Louis, MO, USA: Heaton Research, Inc., 2015.

[17] F. Wei, A. Bower, L., Gates, A., Rose, and D. T. Vasko, "The Full-Scale Helicopter Flight Simulator Design and Fabrication at CCSU", *57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2016.

[18] M. Jirgl, J. Boril, and R. Jalovecky, "The identification possibilities of the measured parameters of an aircraft model and pilot behavior model on the flight simulator". *International Conference on Military Technologies (ICMT)*, 2015, vol., no., pp.1-5.

[19] A. Kaviyarasu, and S. Kumar, "Simulation of Flapping-wing Unmanned Aerial Vehicle using X-plane and Matlab/Simulink". "*Defence Science Journal"*, 2014, 64(4), pp.327-331.

[20] K. Winter, I. J. Hayes, and R. Colvin, "Integrating Requirements: The Behavior Tree Philosophy," 2010 8th IEEE International Conference on Software Engineering and Formal Methods, Pisa, 2010, pp. 41-50.

[21] J. Steig, "Average Rate of Change Function", *Mesa Community College*, 2009. [Online]. Available: http://www.mesacc.edu/~marfv02121/readings/average/.

Training and Evaluating the Intelligent Autopilot System for Full Flight Cycles with Oman Air, and Landing Autonomously Beyond the Current Limits and Capabilities Using Artificial Neural Networks

Haitham Baomar[1], Peter J. Bentley[2]

Dept. of Computer Science, University College London, UK

[1]h.baomar@cs.ucl.ac.uk; [2]p.bentley@cs.ucl.ac.uk

Abstract*— We describe the Intelligent Autopilot System (IAS), a fully autonomous autopilot capable of piloting large jets such as airliners by learning from experienced human pilots using Artificial Neural Networks. The IAS is capable of autonomously executing the required piloting tasks and handling the different flight phases to fly an aircraft from one airport to another including takeoff, climb, cruise, navigate, descent, approach, and land in simulation. In addition, the IAS is capable of autonomously landing large jets in the presence of extreme weather conditions including severe crosswind, gust, wind shear, and turbulence. The IAS is a potential solution to the limitations and robustness problems of modern autopilots such as the inability to execute complete flights, the inability to handle extreme weather conditions especially during approach and landing where the aircraft's speed is relatively low, and the uncertainty factor is high, and the pilots shortage problem compared to the increasing aircraft demand. In this paper we present the work done by collaborating with Oman Air to provide training data for the IAS to learn from. The training data are used by Artificial Neural Networks to generate control models automatically. The control models imitate the skills of the human pilot when executing all the piloting tasks required to pilot an aircraft between two airports. In addition, we introduce new ANNs trained to control the aircraft's elevators, elevators' trim, throttle, flaps, and new ailerons and rudder ANNs to counter the effects of extreme weather conditions and land safely. Experiments show that small datasets containing single demonstrations are sufficient to train the IAS and achieve excellent performance by using clearly separable and traceable neural network modules which eliminate the black-box problem of large Artificial Intelligence methods such as Deep Learning. In addition, experiments show that the IAS can handle landing in extreme weather conditions beyond the capabilities of modern autopilots and even experienced human pilots. The proposed IAS is a novel approach towards achieving full control autonomy of large jets using ANN models that match the skills and abilities of experienced human pilots and beyond.

I. INTRODUCTION

Human pilots are trained to perform piloting tasks that are required during the different phases of the flight. Performing a complete flight cycle starts with a ground-run on the runway to gain speed, rotate after a certain airspeed is achieved, climb, cruise while navigating between waypoints, descend, prepare for final approach while intercepting the landing runway path line, touchdown, flare, and slowdown to taxi speed [1].

In contrast, Automatic Flight Control Systems (AFCS) or autopilots are highly limited, capable of performing minimal piloting tasks. Although modern autopilots can maintain or hold a desired heading, speed, altitude, and even perform auto-land, they cannot handle complete flight cycles automatically, and they must be engaged and operated manually by the human pilots to constantly change and update the desired parameters. In addition, modern autopilots cannot handle severe weather conditions, such as strong crosswind components combined with wind shear, gust, and turbulence especially during final approach and landing. The reason for such limitations of conventional AFCS is that it is not feasible to anticipate all the potential uncertainties such as weather conditions and incorporate all of that into the set of rules or control models "hardcoded" in an AFCS, and the robustness issues of PID controllers which modern autopilots rely on.

This work aims to address this problem by creating an Intelligent Autopilot System (IAS) with the capability to perform complete flights autonomously using Artificial Neural Networks. The IAS is a novel approach which introduces the possibility to transfer human intelligence and intuitions required to pilot an aircraft to an autonomous system. By using this approach, we aim to extend the capabilities of modern autopilots and enable them to autonomously perform all the necessary piloting tasks to complete full flight cycles.

The work in this paper builds on previous work by the authors [2][3][4][5] which describe previous versions of the IAS that learned from training data provided by the first author who does not have piloting experience. Although the latter work presented cockpit autonomy capabilities, the IAS did not fully behave like an experienced human pilot of an airliner especially when manipulating the different control surfaces to maintain desired parameters such as altitude and the final approach

glideslope. In addition, the previous versions did not have the ability to maintain desired speeds, climb/sink rates, and correctly control the flaps settings. Therefore, the work in this paper describe the effort conducted to alter and enhance the behaviour of the IAS to mimic the behaviour of experienced human pilots of airliners by redesigning the system's Artificial Neural Networks to learn from new training data collected from demonstrations performed by an experienced Oman Air captain through a joint training project. Furthermore, this work aims to equip the IAS with the ability to surpass the current limits and abilities of landing in extreme weather conditions.

This paper is structured as follows: part (II) reviews related literature on autonomous flight control systems. Part (III) explains the Intelligent Autopilot System (IAS). Part (IV) describes the experiments, Part (V) describes the results by comparing the behaviour of IAS with the behaviour of the human pilot and the behaviour of the standard PID-based autopilot as well, and part (VI) provides an analysis of the results. Finally, we provide conclusions and future work.

II. BACKGROUND

The concept of introducing intelligent autonomy to the cockpit is gaining significant interest due to multiple factors such as the robustness issues of current automation technology, human error, and the shortage of pilots compared to the increasing aircraft demand. Selecting the suitable intelligent autonomy technology for such safety-critical domain is a subject of interest and debate. In [6], an active disturbance rejection control (ADRC) strategy based on fuzzy control is proposed, which is designed to improve the ability of anti-interference, meanwhile, fuzzy control is adopted to adjust the ADRC parameters online, which makes control performance better. Simulation results show that compared with conventional PID the Fuzzy-ADRC strategy can suppress the disturbances quickly and efficiently, with higher control accuracy, stronger robustness and so on [6]. In [7], a fuzzy self-tuning PID (FSPID) controller to tackle the disadvantages of conventional PID controllers in aircraft autopilots is proposed where fuzzy self-tuning PID tunes the PID parameters to achieve the optimal performance, which based on the results in simulation, the proposed controller can adaptively improve the system response by on-line setting of PID parameters. Other methods were used to enhance the pitch control performance such as Linear Quadratic Regulator (LQR) [8], and Fuzzy Logic Controllers (FLC) [9].

For altitude control, a non-minimum phase (NMP) dynamic control systems is proposed in [10] where an invert closed loop system performed better than conventional Linear Quadratic Gaussian (LQG) when holding a given altitude. In [11], Artificial Neural Network's direct inverse control (DIC-ANN) with the PID control system is proposed where the linearization simplified the solving process for such mathematical based model, omitting the nonlinear and the coupling terms is unsuitable for the dynamics of the multirotor vehicle.

Applying intelligent control methods to aircraft speed control is investigated in [12] where a speed command controller is enhanced by applying a command filter as well as an additional feed forward command.

For flaps control, [13] proposes a dynamic flaps controller that continuously adjusts the flaps settings based on speed to achieve optimal flight dynamics throughout the flight. In addition, [14], The controllability of a flap-controlled system is analysed based on nonlinear controllability theory.

Autonomous landing is a subject that is being covered extensively in research due to the need to introduce intelligent control systems that can handle the difficult problem of landing safely especially in severe weather conditions such as crosswind and low visibility. In [15], a vision-based method for determination of the position of a fixed-wing aircraft approaching a runway is proposed where the method determines the location of an aircraft based on positions of precision approach path indicator lights and approach light system with sequenced flashing lights in the image captured by an on-board camera. In [16], A comprehensive Autoland design for a representative model of a twin-engine commercial aircraft is proposed where a cascaded control structure is selected which resembles integrator chains. The classical loop shaping is used to design the individual control loops where the emphasis is on providing a complete and comprehensive qualitative design strategy [16]. In [17], a control system architecture with strong disturbance rejection characteristics for Unmanned Aircraft is presented where the primary objective is to accurately land a fixed-wing aircraft under adverse weather conditions. A synergistic controller architecture is presented, where the aim is to design a structure capable of executing one of three landing techniques, or combination thereof, by simply activating various controllers at different stages of the landing phase [17]. An acceleration-based controller architecture is used for the inner-loop controllers to reject disturbances at the acceleration level before they manifest as deviations in inertial position and velocity [17]. [18] proposes an autonomous approach and landing navigation method whose accuracy is comparable to Inertial/Differential GPS (DGPS) integration. The method integrates inertial data, forward-looking infrared (FLIR) images, and runway geographic information to estimate kinetics states of aircraft during approach and landing [18]. An existing method is enhanced to robustly detect runway, accurately extract three vertexes of runway contour from FLIR images and synthesize the virtual runway features by runway geo-information and aircraft's pose parameters [18]. Then, real and synthetic runway features are used to create vision cues and integrate them with inertial data in square-root unscented Kalman filter to estimate the motion errors [18]. [19] proposes an improved multi-group swarm-based optimization method that can not only optimize the parameters of the lateral flight control system, but also find diversity solutions of the underlying optimization problem. During the optimizing process, several swarm groups are generated to search

potential areas for the optimal solution [19]. These groups exchange information with each other during the searching process and focus on their different but continuous spaces [19].

Controlling the aircraft's roll by using the ailerons is in the heart of navigation and path interception. In [20], an autopilot system that uses sliding mode control (SMC) method is proposed. The results show enhanced performance using MATLAB/Simulink environment [20]. A variant of the sliding technique used in [20] is used in [21] as well. The proposed SMC algorithm-based on nonlinear sliding surface is derived using the kinematic equations for bank-to-turn vehicles [21].

In addition, utilizing the rudder ensures the interception of the runway's centreline during takeoff and landing in the presence of crosswind. In [22], a grid method for computing the value function and optimal feedback strategies for the control and disturbance is used to optimize the control of the rudder by handling nonlinear and linearized model of the aircraft on the ground.

During final approach, maintaining a desired glideslope ensures safe and soft landings. In [23], controllers that modify the reference model associated with aircraft pitch angle are proposed. The control of the pitch angle and longitudinal velocity is performed by a neural network adaptive control system, based on the dynamic inversion concept [23]. In [24], a network model optimization algorithm based on onboard flight recorder data is suggested.

In a report [25] prepared for NASA by Honeywell Aerospace and Defence, the Intelligent Autopilot System (IAS) described in this paper is briefly evaluated with the emphasis on the problem-breakdown approach of the IAS where multiple and independent small components designed to handle specific tasks are managed by a high-level component, which is in line with the recommendations of the report. However, [25] mistakenly claims that the IAS uses Inverse Reinforcement Learning where capturing a sequence of sub-tasks or reward functions that makeup a high-level task becomes quite challenging [26]; in fact, our system uses Supervised Learning by applying fully connected single-layer Artificial Neural Networks (ANNs), which is a method that can undergo Verification and Validation (V&V) given the absence of a black-box. [25] emphasizes the need for assuring that the intelligent control system must not behave unexpectedly and must have a certain level of situational awareness where the behaviour is altered to handle an emergency for example. Although the IAS is a proof-of-concept designed to prove the possibility of introducing intelligent autonomy to the cockpit, not a fully developed mature autopilot, we have already paid attention to the assurance points by making sure the training datasets contain specific patterns that guarantee the elimination of unexpected behaviour. In addition, the IAS is capable of detecting several unusual conditions such as emergency situations where the behaviour is altered to cope with the situation.

It is clear that intelligent autonomy is covered in the literature in many recent research papers, however, the work is dedicated to tackling specific flight automation problems such as maintaining speed or altitude rather than proposing comprehensive cockpit autonomy solutions such as the IAS. In most papers, the authors tackle the robustness issues of PID controllers that modern autopilots rely on by applying a layer of intelligent control to those conventional controllers such as adding Artificial Neural Networks or Fuzzy Logic to the PID controllers closed loop to enhance performance and accuracy [7][11] instead of fully replacing them with intelligent control solutions, which increases the complexity of the proposed solution rather than attempting to simplify it. In addition, most of the work effort is focused on solutions for small to medium Unmanned Aerial System with few attention to large airplanes such as airliners.

III. The Intelligent Autopilot System

The proposed Intelligent Autopilot System (IAS) in this paper can be viewed as an apprentice that observes the demonstration of a new task by the experienced human teacher, and then performs the same task autonomously. A successful generalization of learning should take into consideration the capturing of low-level models and high-level models which can be viewed as rapid and dynamic sub-actions that occur in fractions of a second, and actions governing the whole process and how it should be performed strategically. It is important to capture and imitate both levels to handle different piloting tasks successfully. The IAS is made of the following components: a flight simulator, an interface, flight control hardware, a database, a flight manager program, and Artificial Neural Networks. The IAS implementation method has three steps: A. Data Collection, B. Training, and C. Autonomous Control. In each step, different IAS components are used. The following sections describe each step and the components used in turn.

A. Data Collection

Fig. 1 illustrates the IAS components used during the data collection step.

1) Flight Simulator

Before the IAS can be trained or can take control, in most cases, we must collect data from a human pilot. This is performed using X-Plane which is an advanced flight simulator that has been used as the simulator of choice in many research papers such as [27] [28] [29]. X-Plane is used by multiple organizations and industries such as NASA, Boeing, Cirrus, Cessna, Piper, Precession Flight Controls Incorporated, Japan Airlines, and the American Federal Aviation Administration. X-Plane can communicate with external applications by sending and receiving flight data and control commands data over a network

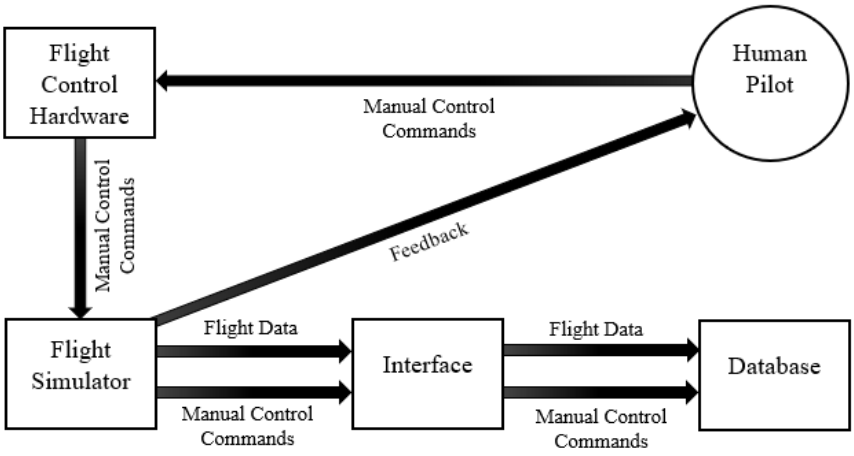

Fig. 1. Block diagram illustrating the IAS components used during the pilot data collection step.

through User Datagram Protocol (UDP) packets. For this work, the simulator is set up to send and receive packets comprising desired data every 0.1 second.

2) The IAS Interface

The IAS interface is responsible for data flow between the flight simulator and the system in both directions. It provides a Graphical User Interface (GUI) for the user to select pre-flight options such as the destination airport, altitude, speed, and climb-rate. The interface displays flight data received from the simulator, and control command data sent back to the simulator. In addition, the interface provides data collection options through the GUI, which sends the collected data to the database. After selecting the desired data collection options, the human teacher uses the flight control hardware to perform the piloting task to be learned. The interface collects data from X-Plane over the network using UDP packets including current flight data and the pilot's actions while piloting the aircraft which are organized into vectors of inputs and mapped outputs, and sent to the database to be stored as training data.

3) Flight Control Hardware

In this work, we use a HOTAS (Hands On Throttle-And-Stick) system by Logitech called G Saitek X52 Pro Flight Control System which provides a comprehensive set of hardware interface for the human pilot to use including a stick to control the aircraft's roll, yaw, and pitch, in addition to a throttle handle to control the engines' thrust, and a group of buttons and switches to control brakes, gear, speed-brakes, flaps, etc.

4) Database

An SQL Server database stores the data captured from X-Plane and the pilot's demonstrations which are received from the interface. The database contains tables designed to store: 1. Flight data as inputs, and 2. Pilot's actions as outputs. These tables are then used as training datasets to train the Artificial Neural Networks of the IAS.

B. Training

1) Artificial Neural Networks

After the human pilot data collection step is completed, Artificial Neural Networks are used to generate learning models from the captured datasets through offline training. Fig. 2 illustrates the training step. Twenty-two feedforward Artificial Neural Networks comprise the core of the IAS. Each ANN is designed and trained to handle specific control or task. The ANNs that are relevant to this work are: the Pitch Rate of Change ANN, the Elevators ANN, the Altitude Rate of Change ANN, the Elevators Trim ANN, the Speed Rate of Change ANN, the Throttle ANN, the Flaps ANN, the Roll ANN, the Ailerons ANN, the Heading ANN, the Rudder ANN, the Glideslope Rate of Change ANN, and the Glideslope Elevators Trim ANN. The remaining ANNs that handle other tasks such as brakes control, gear control, and emergency situations are discussed in our previous work [2][3][4][5]. Table I describes the inputs and outputs of the ANNs which represent the gathered data and relevant actions, and the flight phase in which each ANN is used. The topologies of the ANNs are illustrated in Fig. 3. The method for choosing ANN topologies in this work is based on an implication [30] which indicates that direct mapping problems requiring more than one hidden layer are rarely encountered, and compared to Deep Learning, this approach means that the system is more understandable and easier to test and verify compared to single deep solutions which are black-boxes unsuited for safety

critical applications. Before training, the datasets are retrieved from the database. Then, the datasets are fed to the ANNs. Next, Hyperbolic Tangent (Tanh) (1) [31] function is used for the neuron activation step where x is the neuron output.

TABLE I

THE ARTIFICIAL NEURAL NETWORKS DEVELOPED FOR THIS WORK, THE FLIGHT PHASE IN WHICH THEY ARE USED, AND THEIR DESCRIPTION.

Artificial Neural Network (ANN)	**Flight Phase**	**Description**
Pitch Rate of Change ANN	Takeoff	Takes the difference between the aircraft's pitch and the desired pitch as input, and predicts the appropriate rate of change of pitch degrees that is required to reach the desired pitch.
Elevators ANN	Takeoff	Takes the difference between the current rate of change of pitch degrees and the desired rate of change (predicted by the Pitch Rate of Change ANN) as input, and predicts the appropriate command to be sent to the elevators.
Altitude Rate of Change ANN	Cruise	Takes the difference between the aircraft's altitude and the desired altitude as input, and predicts the desired rate of change (climb/sink rate).
Elevators Trim ANN	Cruise	Takes the difference between the current rate of change and the desired rate of change (predicted by the Altitude Rate of Change ANN) as input, and predicts the appropriate command to be sent to the elevators' trim.
Speed Rate of Change ANN	All	Takes the difference between the aircraft's speed and the desired speed as input, and predicts the desired rate of change of speed.
Throttle ANN	All	Takes the difference between the current rate of change of speed and the desired rate of change (predicted by the Speed Rate of Change ANN) as input, and predicts the appropriate command to be sent to the throttle.
Flaps ANN	Takeoff, Approach, and Final Approach	Takes the aircraft's altitude and the flight phase as inputs, and predicts the appropriate command to be sent to the flaps.
Roll ANN	All	Takes the difference between the aircraft's current angle and the desired angle (0 degrees/centreline) as input, and predicts the desired roll degree to bank the aircraft towards the path-line.
Ailerons ANN	All	Takes the difference between the current roll and the desired roll (predicted by the Roll ANN) as input, and predicts the appropriate command to be sent to the ailerons to bank.

Heading ANN	Landing	Used on the runway to align the aircraft with the centreline of the runway. It takes the difference between the aircraft's current angle and the desired angle (0 degrees/centreline) as input, and predicts the desired heading degree that the aircraft should follow on tarmac to be aligned with the centreline of the runway.
Rudder ANN	Final Approach, and Landing	Takes the difference between the current heading and the desired heading (predicted by the Heading ANN), and predicts the appropriate command to be sent to the rudder.
Glideslope Rate of Change ANN	Approach, and Final Approach	Takes the difference between the aircraft's glideslope degree and the desired glideslope degree as input, and predicts the desired rate of change of the glideslope angle that is necessary to align the aircraft with the desired glideslope angle.
Glideslope Elevators Trim ANN	Approach, and Final Approach	Takes the difference between the current rate of change of the glideslope angle and the desired rate of change (predicted by the Glideslope Rate of Change ANN) as input, and predicts the appropriate command to be sent to the elevators' trim.

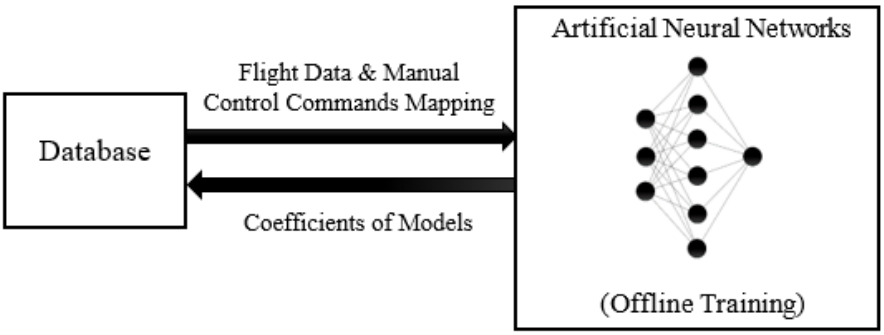

Fig. 2. Block diagram illustrating the IAS components used during training.

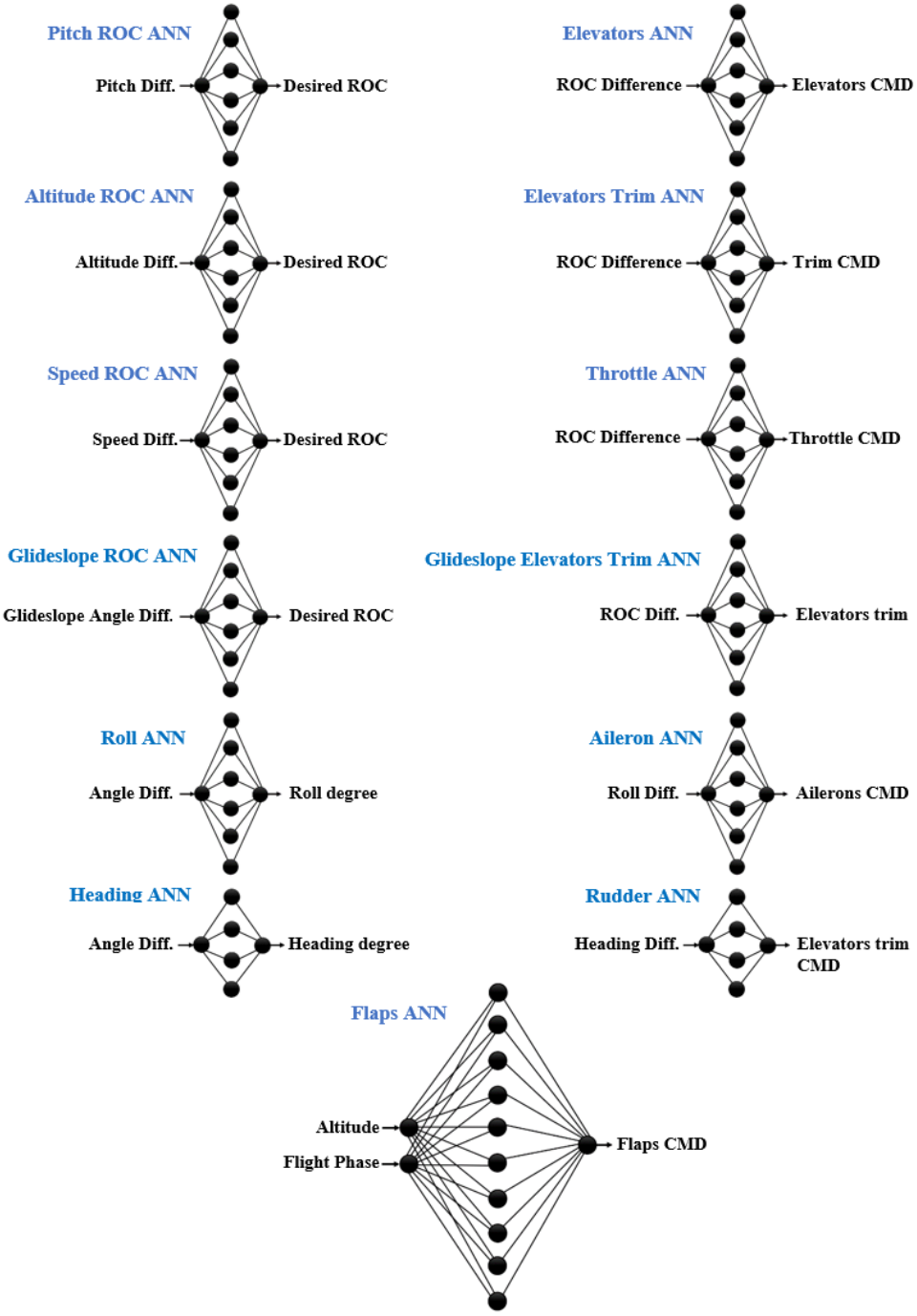

Fig. 3. Inputs, outputs, and the topologies of the ANNs relevant to this work. Each ANN is designed and trained to handle a specific task.

$$\Phi(x) = tanh(x) \quad (1)$$

Next, Backpropagation is applied as follows:

$$\Phi'(x) = 1.0 - \Phi^2(x) \quad (2)$$

where phi (Φ) of x is the result of the activation function. Then, coefficients of models (weights and biases) are updated using (3) [31].

$$\Delta w_{(t)} = -\epsilon \frac{\partial E}{\partial w_{(t)}} + \alpha \Delta w_{(t-1)} \qquad (3)$$

where ϵ is the learning rate, $\frac{\partial E}{\partial w_{(t)}}$ is the gradient, α is the momentum, and $\Delta w_{(t-1)}$ is the change in the previous weight.

After training is completed, the learning models are generated, and the free parameters or coefficients represented by weights and biases of the models are stored in the database.

C. Autonomous Control

Once trained, the IAS can now be used for autonomous control. Fig. 4 illustrates the components used during the autonomous control step.

1) The IAS Interface

Here, the interface retrieves the coefficients of the models from the database for each trained ANN, and receives flight data from the flight simulator every 0.1 second. The interface organizes the coefficients into sets of weights and biases, and organizes data received from the simulator into sets of inputs for each ANN. The relevant coefficients, and flight data input sets are then fed to the Flight Manager and the ANNs of the IAS to produce outputs. The outputs of the ANNs are sent to the interface which sends them to the flight simulator as autonomous control commands using UDP packets every 0.1 second.

2) The Flight Manager Program

The Flight Manager is a program which resembles a Behaviour Tree [32]. The purpose of the Flight Manager is to manage the transition between the different flight phases and their desired speed and altitude if not already selected by the user, generate and set the navigation course, and manage all the ANNs of the IAS by deciding which ANNs are to be used simultaneously at each moment. Fig. 5 illustrates how the Flight Manager manages the flight phases shown in Fig. 6 by continuously examining the speed and altitude of the aircraft, and the distance to the next waypoint to detect the transition points between the different flight phases. In addition, the Flight Manager detects the Top of Descent (TOD) point where the IAS starts the descent towards the destination airport by applying (4) [33]. The Flight Manager receives the required data from the interface of the IAS as Fig. 4 shows. The methods used by the Flight Manager to manage the ANNs and the navigation course are explained in our previous work [3][4][5].

$$TOD = \frac{(Altitude_{(cruise)} - Altitude_{(flx)}) \,/\, 100}{Descent_{(angle)}} \qquad (4)$$

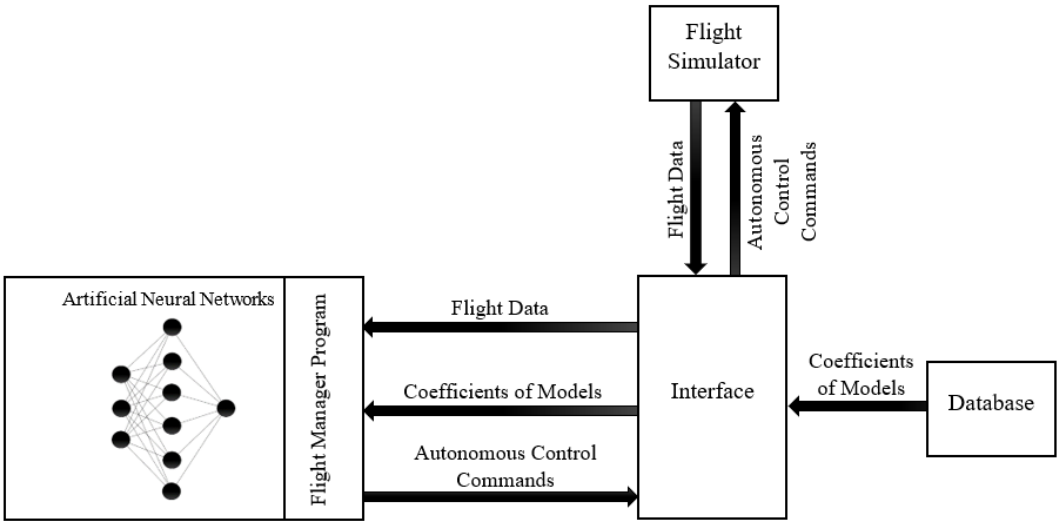

Fig. 4. Block diagram illustrating the IAS components used during autonomous control.

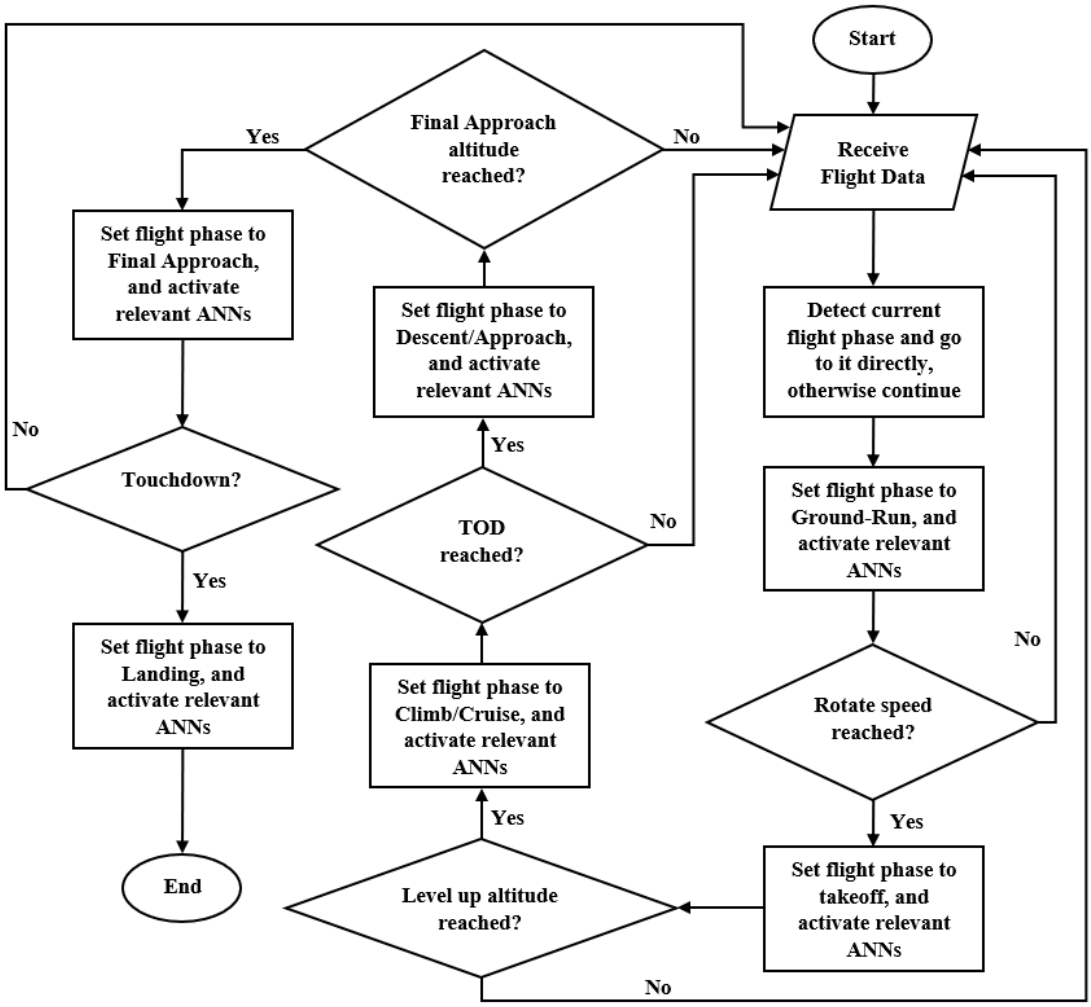

Fig. 5. A Flowchart illustrating the process which the Flight Manager program follows to handle the transmission between the different flight phases.

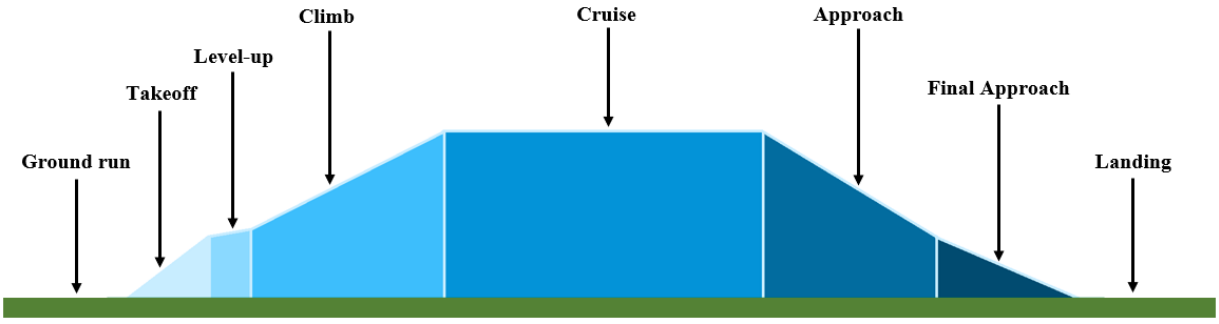

Fig. 6. The different flight phases followed and managed by the Flight Manager.

3) Artificial Neural Networks

The relevant set of flight data inputs received through the interface is used by the ANNs' input neurons along with the relevant coefficients to predict control commands or other data given the flight status by applying (1). The values of the output layers are sent to the interface which sends them to the flight simulator as autonomous control commands. The design approach

of the ANNs intends to breakdown the different tasks required to pilot an aircraft during the multiple flight phases to small components. Following the problem breakdown approach, it is possible to achieve a composition of small multiple control units represented by the task-dedicated ANNs that can be designed, integrated, and traced effortlessly compared to systems that rely on a single or few large ANNs designed to handle multiple tasks. In addition, when following the breakdown approach, it is possible to achieve higher levels of accuracy since each ANN is dedicated towards a single task such as specific control mapping.

IV. EXPERIMENTS

Although the previous versions of the Intelligent Autopilot System (IAS) that learned from training data provided by the first author who does not have piloting experience presented cockpit autonomy capabilities [2][3][4][5], the IAS did not fully behave like an experienced human pilot of an airliner, therefore, the latest version of the Intelligent Autopilot System (IAS) was redesigned and trained with Oman Air to achieve the desired autonomous behaviour that can be compared to the behaviour of experienced human pilots of airliners. This section discusses the experiments conducted on the latest version of the IAS.

The human teacher who provided the demonstrations is Captain Khalid Al Hashmi, Senior Manager Crew Training at Oman Air. The simulated aircraft used for the experiments is a certified Boeing B787 Dreamliner model as we want to experiment using a complex and large model with more than one engine rather than a light single-engine model. Since the design approach of the IAS which utilizes Supervised Learning and many small single-hidden-layer ANNs requires single demonstrations of the tasks to be learned, Captain Al Hashmi provided one demonstration of a short flight from one airport to another in X-Plane. Captain Al Hashmi took off from London Heathrow (EGLL), cruised at 10,000 ft, then landed in Birmingham (EGBB). Captain Al Hashmi followed the standard piloting procedures where he started the ground-run phase on the takeoff runway, rotated, and maintained a 15 degrees pitch angle during takeoff. Then, he engaged the aircraft's autopilot to climb to the cruise altitude of 10,000 ft, to maintain a cruise speed of 240 knots, and to follow the preloaded flight path using GPS waypoints. Immediately after reaching the Top of Descent (TOD) point, Captain Al Hashmi initiated the approach flight phase by updating the speed parameter in the aircraft's autopilot to 205 knots and starting the decent to follow the standard 3 degrees glide slope. Then, he updated the speed parameter to reach the landing speed of 150 knots before reaching the final approach flight phase. During the latter flight phases, he engaged the flaps at different altitudes to extend them to certain degrees accordingly. Finally, after the speed reached 150 knots, and at around 1,500 ft, he disengaged the autopilot, and took full control of the aircraft to continue maintaining the landing speed and the 3 degrees glideslope until touchdown.

The data of interest that was collected and used to train the IAS are the inputs and outputs of the different ANNs discussed in this work and illustrated in Fig. 3. The experiments were conducted on the Elevators ANN to test the ability of maintaining the desired takeoff pitch angle, the new Elevators Trim ANNs to test the ability of maintaining different altitudes, climb rates, and the glideslope during approach and final approach, the new Throttle ANN to test the ability of maintaining different desired speeds, and the modified Flaps ANN to test the ability of extending the flaps correctly. The latter capabilities were not available in the previous versions of the IAS. Furthermore, additional experiments were conducted on the enhanced Ailerons, Rudder, and Roll ANN which replaced the Bearing Adjustment ANN from our previous work [4] to handle runway centreline maintenance during the final approach and landing flight phases in extreme weather conditions beyond the capability of the previous version of the IAS [4] and the capabilities of modern autopilots and even human pilots, as well as the Glideslope Elevators Trim ANN to test its ability to maintain the desired 3 degrees glideslope in the same extreme weather conditions. Our previous work [2][3][4][5] provide detailed explanations of the experiments of autonomous ground-run, navigation, landing procedures after touchdown, and handling emergency situations.

To assess the effectiveness of the proposed approach in this paper, the Intelligent Autopilot System (IAS) was tested in eight experiments: A. Takeoff Pitch Maintenance, B. Altitude Maintenance, C. Climb Rate Maintenance, D. Speed Maintenance, E. Flaps Setting, F. Final Approach Glideslope Maintenance, and G. Runway Centreline Maintenance. The experiments are as follows:

A. Takeoff Pitch Maintenance

The purpose of this experiment is to assess the behaviour of the IAS when maintaining the 15 degrees pitch angle during the takeoff phase, and compare it to the demonstration provided by the human pilot. Since no standard modern autopilot is capable of performing autonomous takeoff, no comparison with the standard autopilot is provided.

1) Training

For this experiment, the Elevators ANN and the Pitch Rate of Change ANN were trained until a low Mean Squared Error (MSE) value was achieved (below 0. 01).

2) Autonomous Control

For this experiment, the aircraft was reset to the runway in the flight simulator, and the IAS was engaged to test the ability of maintaining the standard takeoff pitch angle of 15 degrees. After the IAS completed the ground-run flight phase on the runway, the output of the Elevators ANN and the Pitch Rate of Change ANN were used to hold and maintain the desired pitch angle.

B. Altitude Maintenance

The purpose of this experiment is to assess the behaviour of the IAS compared to the standard autopilot of the model aircraft when maintaining a given altitude since the human pilot used the standard autopilot to handle this task.

1) Training

For this experiment, the Elevators Trim ANN and the Climb Rate ANN were trained until a low Mean Squared Error (MSE) value was achieved (below 0. 01).

2) Autonomous Control

After training the ANNs, the aircraft was reset to the runway in the flight simulator, and the IAS was engaged to test the ability of maintaining different altitudes selected manually by the user. After the IAS took the aircraft airborne and reached the cruise flight phase, the output of the Altitude Rate of Change ANN and the Elevators Trim ANN were used to hold and maintain three different altitudes at three different speeds, and maintain three different altitudes while speed is increasing from one speed to another and decreasing from one speed to another.

C. Climb Rate Maintenance

The purpose of this experiment is to assess the behaviour of the IAS compared to the standard autopilot of the model aircraft when maintaining a given climb or sink rate while changing altitude since the human pilot used the standard autopilot to handle this task.

1) Training

For this experiment, the same models generated after training the Elevators Trim ANN and the Climb Rate ANN in the previous experiments (A. Altitude Maintenance) were used without having to provide additional training.

2) Autonomous Control

For this experiment, the aircraft was reset to the runway in the flight simulator, and the IAS was engaged to test the ability of maintaining different climb rates selected manually by the user. After the IAS took the aircraft airborne and reached the cruise flight phase, the output of the Altitude Rate of Change ANN and the Elevators Trim ANN were used to hold and maintain six different climb or sink rates.

D. Speed Maintenance

The purpose of this experiment is to assess the behaviour of the IAS compared to the standard autopilot of the model aircraft when maintaining a given speed since the human pilot used the standard autopilot to handle this task.

1) Training

For this experiment, the Throttle ANN and the Speed Rate of Change ANN were trained until a low Mean Squared Error (MSE) value was achieved (below 0. 01).

2) Autonomous Control

After training the ANNs, the aircraft was reset to the runway in the flight simulator, and the IAS was engaged to test the ability of maintaining different speeds selected manually by the user. After the IAS took the aircraft airborne and reached the cruise flight phase, the output of the Throttle ANN and the Speed Rate of Change ANN were used to hold and maintain three different speeds at three different altitudes.

E. Flaps Setting

The purpose of this experiment is to assess the behaviour of the IAS compared to the human pilot when extending and retracting the flaps given the altitude during the different flight phases.

1) Training

For this experiment, the Flaps ANN was trained until a low Mean Squared Error (MSE) value was achieved (below 0. 01).

2) Autonomous Control

After training the ANN, the aircraft was reset to the runway in the flight simulator, and the IAS was engaged to test the ability of correctly deploying and retracting the flaps using different settings during the ground-run phase, takeoff, approach, and final approach. The output of the Flaps ANN was used to select the different flaps settings.

F. Final Approach Glideslope Maintenance

The purpose of this experiment is to assess the behaviour of the IAS compared to the standard autopilot of the model aircraft and the human pilot as well (during the last moments of final approach after disengaging the standard autopilot and taking full control) when maintaining the standard 3 degrees glideslope during the approach and the final approach flight phases in calm weather. In addition, this experiment assesses the behaviour of the IAS compared to the standard autopilot (Autoland) when maintaining the standard 3 degrees glideslope during the approach and the final approach flight phases in extreme weather conditions.

1) Training

For this experiment, the Glideslope Rate of Change ANN and the Glideslope Elevators Trim ANN were trained until a low Mean Squared Error (MSE) value was achieved (below 0. 01).

2) Autonomous Control

After training the ANNs, the aircraft was reset to the runway in the flight simulator, and the IAS was engaged to test the ability of maintaining the standard 3 degrees glideslope during approach and final approach in calm and extreme weather conditions. After the IAS took the aircraft airborne reached the approach flight phase, the output of the Glideslope Rate of Change ANN and the Glideslope Elevators Trim ANN were used to maintain the desired glideslope. The extreme weather conditions provided strong crosswind, gust, shear, and turbulence.

G. Runway Centreline Maintenance

The purpose of this experiment is to assess the behaviour of the IAS compared to the standard autopilot of the model aircraft and the human pilot as well (during the last moments of final approach after disengaging the standard autopilot and taking full control) when maintaining the centreline of the runway during the approach, final approach, and landing flight phases in calm weather. In addition, this experiment assesses the behaviour of the IAS compared to the standard autopilot (Autoland) when maintaining the centreline of the runway during the approach, final approach, and landing flight phases in extreme weather conditions.

1) Training

For this experiment, the Roll ANN and the Ailerons ANN were trained until a low Mean Squared Error (MSE) value was achieved (below 0. 01).

2) Autonomous Control

After training the ANNs, the aircraft was reset to the runway in the flight simulator, and the IAS was engaged to test the ability of maintaining the centreline of the landing runway in calm and extreme weather conditions. After the IAS took the aircraft airborne and reached the approach flight phase, the output of the Roll ANN, the Ailerons ANN, and the Rudder ANNs were used to maintain the centreline of the landing runway. The extreme weather conditions provided strong wind including crosswind, gust, shear, and turbulence.

V. RESULTS

The following section describes the results of the conducted tests.

A. Takeoff Pitch Maintenance

Two models were generated for the Elevators ANN and the Pitch Rate of Change ANN with Mean Squared Error (MSE) values of 0.004 and 0.001 consecutively. Fig. 7 shows the pitch degree over time during ten different takeoffs where the IAS is controlling the elevators to maintain the standard fifteen degrees pitch angle (the lines in different shades of blue) compared to the demonstration of the human pilot (the green line). Since the standard autopilot is not capable of performing takeoff autonomously, no comparison is provided. Table II shows the result of applying the Two One-Sided Test (TOST) [34] to examine the equivalence of the pitch degrees held by the IAS to the desired fifteen degrees takeoff pitch.

B. Altitude Maintenance

Two models were generated for the Elevators Trim ANN and the Climb Rate ANN with MSE values of 0.01 and 0.0003 consecutively. Fig. 8, 9, and 10 illustrate a comparison between the IAS and the standard autopilot when maintaining three different altitudes over time. Since the human pilot used the standard autopilot to maintain the altitude, the comparison is done between the IAS and the standard autopilot. Fig. 11 illustrates a comparison between the latest version of the IAS and the previous version when holding an altitude. The previous version used the throttle to maintain a given altitude, while the latest version uses the correct flight control surface (elevators trim) to maintain a given altitude. Tables III, IV, and V show the results of applying TOST to examine the equivalence between the altitude hold performance of the IAS and the standard autopilot.

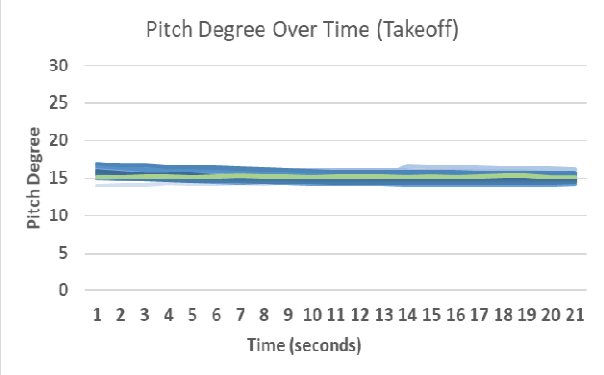

Fig. 7. The pitch degrees held by the IAS over time during fifteen different takeoffs (the lines in different shades of blue) compared to the demonstration of the human pilot (the green line) when maintaining a 15 degrees pitch.

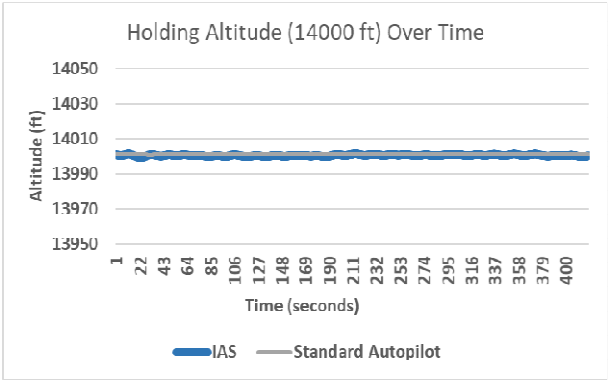

Fig. 8. A comparison between the IAS and the standard autopilot when maintaining an altitude of 14000 ft (speed is 250 knots).

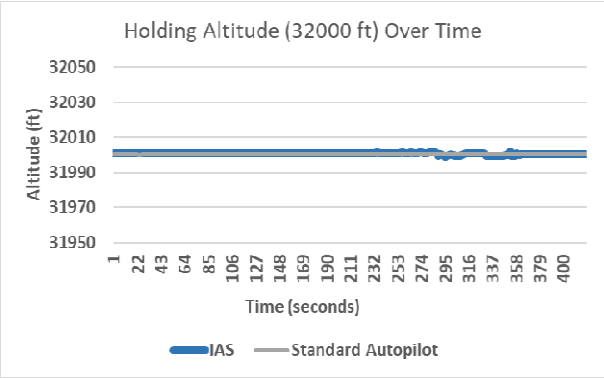

Fig. 9. A comparison between the IAS and the standard autopilot when maintaining an altitude of 32000 ft (speed is 340 knots).

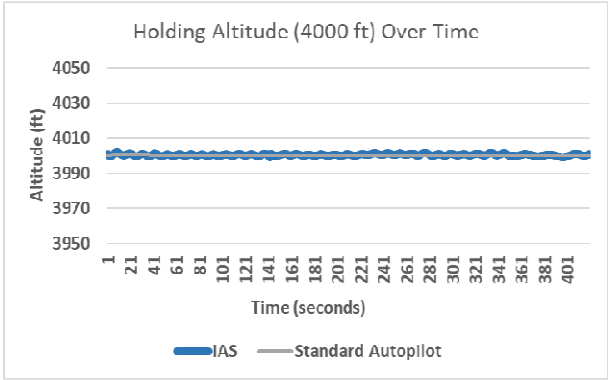

Fig. 10. A comparison between the IAS and the standard autopilot when maintaining an altitude of 4000 ft (speed is 220 knots).

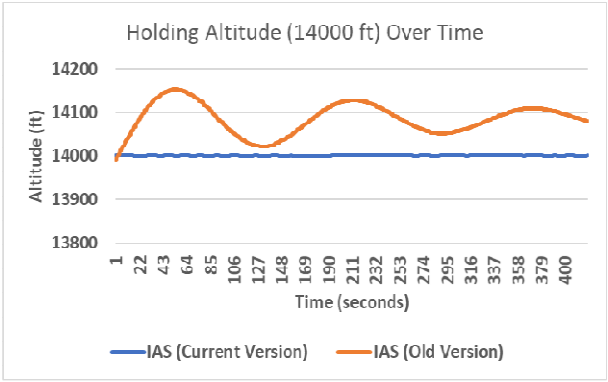

Fig. 11. A comparison between the latest version and the previous version of the IAS when maintaining an altitude of 14,000 ft. The previous version used the throttle, while the latest version uses the elevators trim to maintain a given altitude.

TABLE III
RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS WHEN MAINTAINING A FIFTEEN DEGREES PITCH DURING TAKEOFF COMPARED TO THE HUMAN PILOT.

Equivalence Test for Means
Unequal Sample Sizes
$\alpha = 0.05$

	IAS	Human
Mean	15.24	15.17
Variance	0.36	0.005
Observations	315	21
Pooled Variance	0.34	
Hypothesized Mean Difference	0.8	
df	334	
t Stat	5.63	-6.55
P(T<=t) one-tail	0.000	0.000
T Critical one-tail	1.65	
P(T<=t) two-tail	0.000	
T Critical Two-tail	1.98	

Means are Equivalent because p1 & p2 < 0.05

TABLE II
RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED TO THE STANDARD AUTOPILOT WHEN MAINTAINING AN ALTITUDE OF 14000 FT.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	AP
Mean	14000.49	14000.92
Variance	0.13	0.00
Observations	420	420
Pooled Variance	0.06	
Hypothesized Mean Difference	0.80	
df	838.00	
t Stat	70.87	-20.68
P(T<=t) one-tail	0.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.96	

Means are Equivalent because p1 & p2 < 0.05

TABLE IV
RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED TO THE STANDARD AUTOPILOT WHEN MAINTAINING AN ALTITUDE OF 32000 FT.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	AP
Mean	32000.72	32000.03
Variance	0.24	0.00
Observations	420	420
Pooled Variance	0.12	
Hypothesized Mean Difference	0.80	
df	838.00	
t Stat	4.36	-61.79
P(T<=t) one-tail	0.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.96	

Means are Equivalent because p1 & p2 < 0.05

TABLE V
RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED TO THE STANDARD AUTOPILOT WHEN MAINTAINING AN ALTITUDE OF 4000 FT.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	AP
Mean	4000.38	4000.28
Variance	0.23	0.01
Observations	420	420
Pooled Variance	0.12	
Hypothesized Mean Difference	0.80	
df	838.00	
t Stat	29.38	-37.06
P(T<=t) one-tail	0.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.96	

Means are Equivalent because p1 & p2 < 0.05

C. Climb Rate Maintenance

The same models generated for altitude maintenance (B. Altitude Maintenance) were used to maintain a given climb rate without having to retrain the models. Fig. 12, 13, 14, 15, 16, and 17 illustrate a comparison between the IAS and the standard autopilot when maintaining six different climb rates over time. Since the human pilot used the standard autopilot to maintain the climb rates, the comparison is done between the IAS and the standard autopilot. No comparison with the previous version of the IAS is presented since the previous version did not have the ability to maintain climb rates. Tables VI, VII, VIII, IX, X, and XI show the results of applying TOST to examine the equivalence between the climb rate hold performance of the IAS and the standard autopilot.

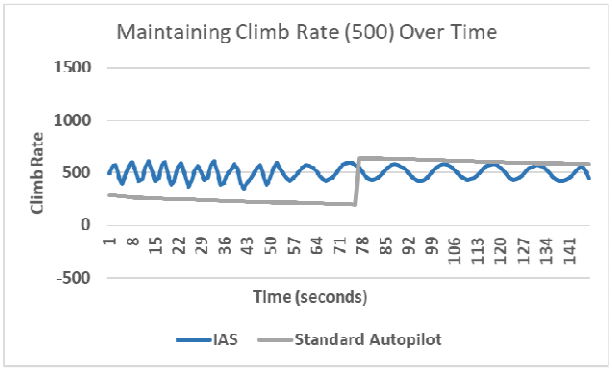

Fig. 12. A comparison between the IAS and the standard autopilot when maintaining a climb rate of 500 ft/min (speed is 250 knots).

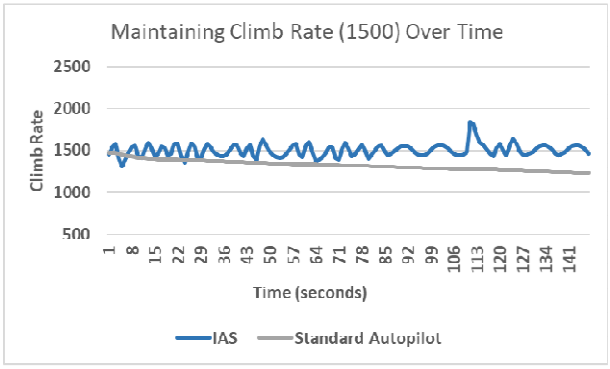

Fig. 13. A comparison between the IAS and the standard autopilot when maintaining a climb rate of 1500 ft/min (speed is 280 knots).

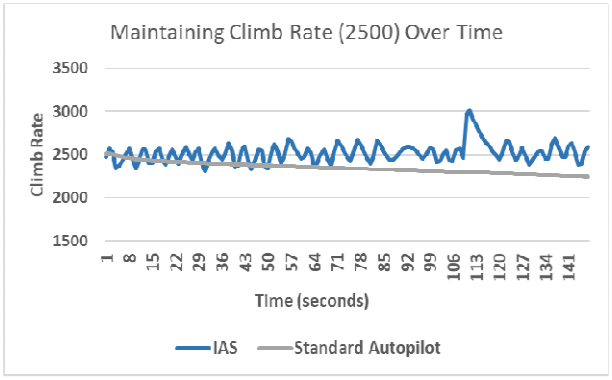

Fig. 14. A comparison between the IAS and the standard autopilot when maintaining a climb rate of 2500 ft/min (speed is 310 knots).

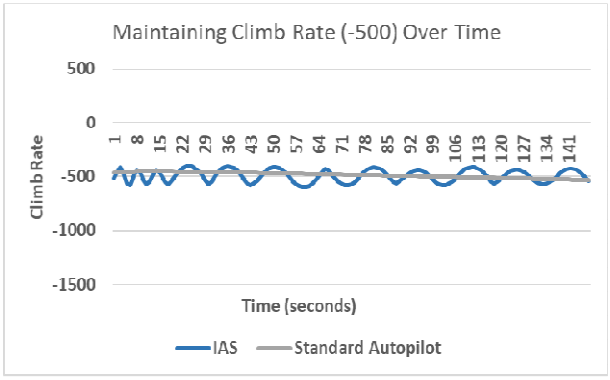

Fig. 15. A comparison between the IAS and the standard autopilot when maintaining a climb (sink) rate of -500 ft/min (speed is 230 knots).

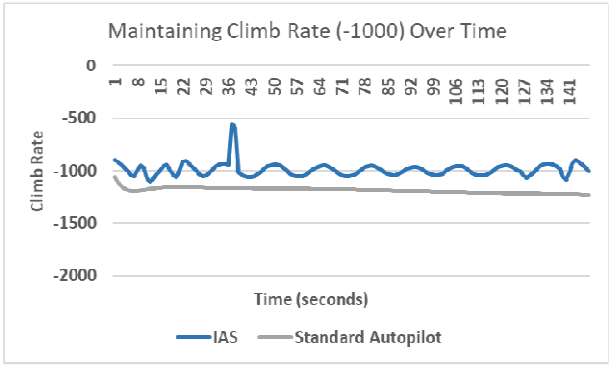

Fig. 16. A comparison between the IAS and the standard autopilot when maintaining a climb (sink) rate of -1000 ft/min (speed is 240 knots).

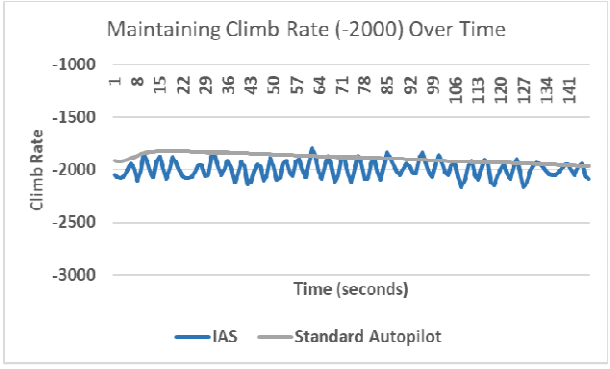

Fig. 17. A comparison between the IAS and the standard autopilot when maintaining a climb (sink) rate of -2000 ft/min (speed is 270 knots).

TABLE VI
RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED TO THE STANDARD AUTOPILOT WHEN MAINTAINING A CLIMB RATE OF 500 FT/MIN.

Equivalence Test for Means
Unequal Sample Sizes
$\alpha = 0.05$

	IAS	AP
Mean	498.89	413.88
Variance	4056.47	36247.80
Observations	147	147
Pooled Variance	20152.14	
Hypothesized Mean Difference	0.80	
df	292.00	
t Stat	-5.09	-5.18
P(T<=t) one-tail	1.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.97	

Cannot conclude means are equivalent

TABLE VII
RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED TO THE STANDARD AUTOPILOT WHEN MAINTAINING A CLIMB RATE OF 1500 FT/MIN.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	AP
Mean	1503.47	1327.31
Variance	5403.78	3514.27
Observations	147	147
Pooled Variance	4459.03	
Hypothesized Mean Difference	0.80	
df	292.00	
t Stat	-22.51	-22.72
P(T<=t) one-tail	1.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.97	

Cannot conclude means are equivalent

TABLE VIII
RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED TO THE STANDARD AUTOPILOT WHEN MAINTAINING A CLIMB RATE OF 2500 FT/MIN.

Equivalence Test for Means
Unequal Sample Sizes
$\alpha = 0.05$

	IAS	AP
Mean	2519.27	2347.46
Variance	12673.10	4014.60
Observations	147	147
Pooled Variance	8343.85	
Hypothesized Mean Difference	0.80	
df	292.00	
t Stat	-16.05	-16.20
P(T<=t) one-tail	1.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.97	

Cannot conclude means are equivalent

TABLE IX
RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED TO THE STANDARD AUTOPILOT WHEN MAINTAINING A CLIMB (SINK) RATE OF -500 FT/MIN.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	AP
Mean	-491.51	-486.45
Variance	3297.94	640.19
Observations	147	147
Pooled Variance	1969.07	
Hypothesized Mean Difference	0.80	
df	292.00	
t Stat	1.13	0.82
P(T<=t) one-tail	0.13	0.21
T Critical one-tail	1.65	
P(T<=t) two-tail	0.26	
T Critical Two-tail	1.97	

Cannot conclude means are equivalent

TABLE X
RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED TO THE STANDARD AUTOPILOT WHEN MAINTAINING A CLIMB (SINK) RATE OF -1000 FT/MIN.

Equivalence Test for Means
Unequal Sample Sizes
$\alpha = 0.05$

	IAS	AP
Mean	-988.85	-1187.84
Variance	4295.21	647.30
Observations	147	147
Pooled Variance	2471.25	
Hypothesized Mean Difference	0.80	
df	292.00	
t Stat	-34.18	-34.46
P(T<=t) one-tail	1.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.97	

Cannot conclude means are equivalent

TABLE XI
RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED TO THE STANDARD AUTOPILOT WHEN MAINTAINING A CLIMB (SINK) RATE OF -2000 FT/MIN.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	AP
Mean	-1996.08	-1886.80
Variance	6133.68	1901.15
Observations	147	147
Pooled Variance	4017.41	
Hypothesized Mean Difference	0.80	
df	292.00	
t Stat	14.89	14.67
P(T<=t) one-tail	0.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.97	

Means are Equivalent because p1 & p2 < 0.05

D. Speed Maintenance

Two models were generated for the Throttle ANN and the Speed Rate of Change ANN with MSE values of 0.0009 and 0.0006 consecutively. Fig. 18, 19, and 20 illustrate a comparison between the IAS and the standard autopilot when maintaining three different speeds over time. Since the human pilot used the standard autopilot to maintain speed, the comparison is done between the IAS and the standard autopilot, however, Fig. 21 illustrates a comparison between the IAS and the human pilot when managing the different speeds throughout the complete flight from takeoff to landing. No comparison with the previous version of the IAS is presented since the previous version did not have the ability to maintain a given speed. Tables XII, XIII, and XIV show the results of applying TOST to examine the equivalence between the speed hold performance of the IAS and the standard autopilot.

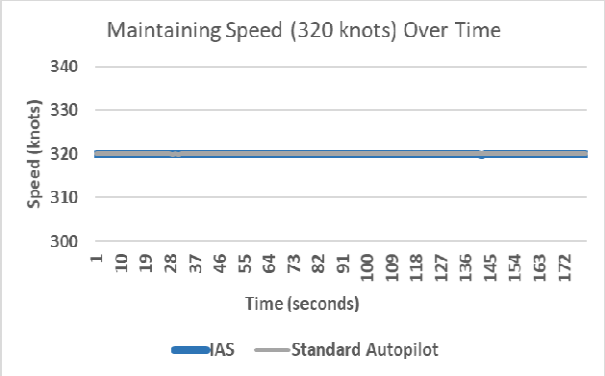

Fig. 18. A comparison between the IAS and the standard autopilot when maintaining a speed of 320 knots (altitude is 22000 ft.).

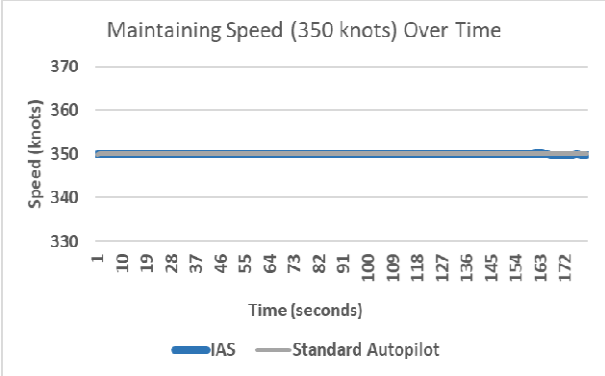

Fig. 19. A comparison between the IAS and the standard autopilot when maintaining a speed of 350 knots (altitude is 30000 ft.).

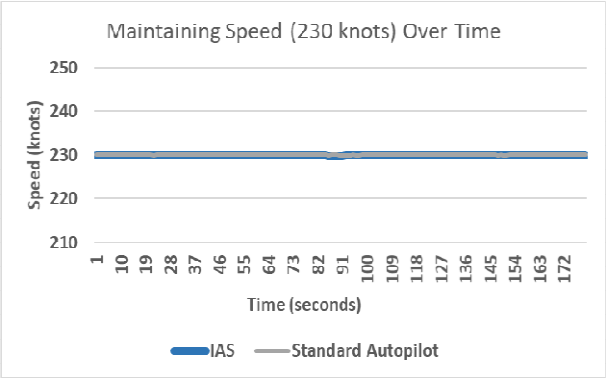

Fig. 20. A comparison between the IAS and the standard autopilot when maintaining a speed of 230 knots (altitude is 10000 ft.).

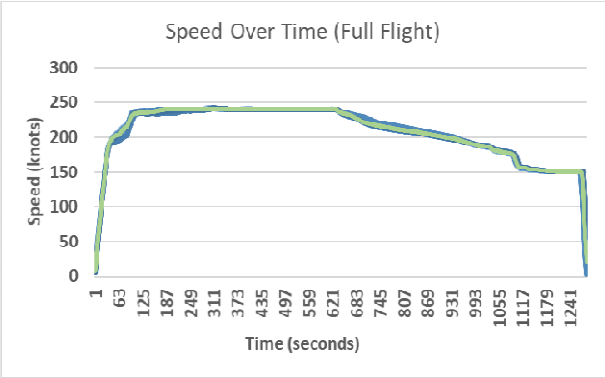

Fig. 21. A comparison between the IAS (10 flights represented by the overlapping lines in different blue shades) and the human pilot (1 demonstration represented by the green line) when managing the different speeds over time throughout the complete flight from takeoff to landing (London Heathrow to Birmingham). As can be seen, both the IAS and the human pilot accelerated sharply until the cruise speed of 240 knots was achieved, then, decelerated gradually until the landing speed of 150 knots was achieved before coming to a full stop on the landing runway.

TABLE XII
RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED TO THE STANDARD AUTOPILOT WHEN MAINTAINING A SPEED OF 320 KNOTS.

Equivalence Test for Means
Unequal Sample Sizes
$\alpha = 0.05$

	IAS	AP
Mean	319.98	320.00
Variance	0.00	0.00
Observations	180	180
Pooled Variance	0.00	
Hypothesized Mean Difference	0.80	
df	358.00	
t Stat	337.34	-321.25
P(T<=t) one-tail	0.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.97	

Means are Equivalent because p1 & p2 < 0.05

TABLE XIII
RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED TO THE STANDARD AUTOPILOT WHEN MAINTAINING A SPEED OF 350 KNOTS.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	AP
Mean	349.98	350.00
Variance	0.00	0.00
Observations	180	180
Pooled Variance	0.00	
Hypothesized Mean Difference	0.80	
df	358.00	
t Stat	167.35	-159.95
P(T<=t) one-tail	0.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.97	

Means are Equivalent because p1 & p2 < 0.05

TABLE XIV
RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED TO THE STANDARD AUTOPILOT WHEN MAINTAINING A SPEED OF 230 KNOTS

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	**AP**
Mean	229.95	230.00
Variance	0.00	0.00
Observations	180	180
Pooled Variance	0.00	
Hypothesized Mean Difference	0.80	
df	358.00	
t Stat	305.61	-268.03
P(T<=t) one-tail	0.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.97	

Means are Equivalent because p1 & p2 < 0.05

E. Flaps Setting

One model was generated for the Flaps ANN with an MSE value of 0.006. Fig. 22 and 23 show the flaps setting over altitude where Fig. 22 shows the flaps setting during the ground-run, takeoff, level-up, climb, and cruise flight phases, while Fig. 23 shows the flaps setting during the cruise, approach, final approach and landing flight phases. Since the standard autopilot is not capable of controlling the flaps autonomously, the provided comparison is between the IAS and the human pilot. Table XV shows the corresponding flaps settings given the deflection value. Table XVI shows the mean, minimum, and maximum altitudes that correlate to each flaps setting in addition to the standard deviation.

TABLE XV
THE APPLIED FLAPS DEFLECTION VALUES AND THEIR CORRESPONDING FLAPS SETTINGS.

Flaps Deflection Value	**Flaps Setting**
0	Flaps Zero
0.166	Flaps One
0.332	Flaps Five
0.664	Flaps Twenty
1	Flaps Full

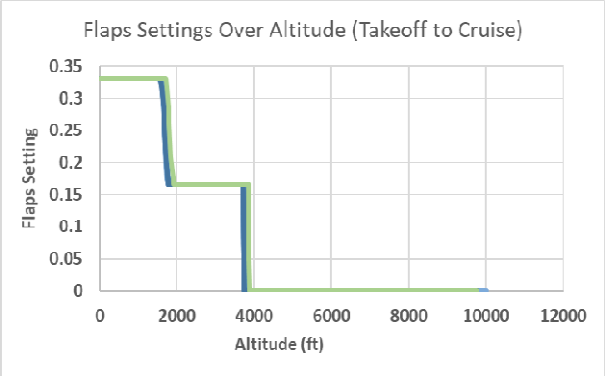

Fig. 22. A comparison between the IAS (10 flights represented by the overlapping lines in different blue shades) and the human pilot (1 demonstration represented by the green line) when managing the different flaps settings over altitude from takeoff to cruise.

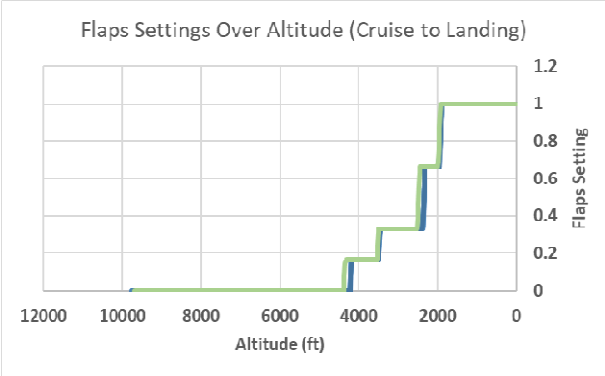

Fig. 23. A comparison between the IAS (10 flights represented by the overlapping lines in different blue shades) and the human pilot (1 demonstration represented by the green line) when managing the different flaps settings over altitude from cruise to landing.

TABLE XVI
A COMPARISON BETWEEN THE HUMAN PILOT AND THE IAS WHEN MANAGING THE CORRELATION BETWEEN THE ALTITUDE (FT) AND FLAPS SETTING INCLUDING MEAN, MINIMUM, AND MAXIMUM ALTITUDES BY THE IAS THAT CORRELATE TO EACH FLAPS SETTING DURING THE DIFFERENT FLIGHT PHASES IN ADDITION TO THE STANDARD DEVIATION.

	Takeoff to Cruise		**Cruise to Landing**			
	Flaps 1	**Flaps 0**	**Flaps 1**	**Flaps 5**	**Flaps 20**	**Flaps Full**
Altitude (Human Pilot)	1800	3800	4150	3450	2330	1890
MIN Altitude (IAS)	1754	3753	4186	3440	2324	1871
MAX Altitude (IAS)	1821	3801	4198	3463	2334	1894
MEAN Altitude (IAS)	1792	3775	4192	3452	2329	1886
STD (IAS)	20	18	4	7	3	7

F. Final Approach Glideslope Maintenance

Two models were generated for the Glideslope Rate of Change ANN and the Glideslope Elevators Trim ANN with MSE values of 0.0006 and 0.0008 consecutively. Fig. 24 illustrates a comparison between the IAS, the standard autopilot, and the human pilot (the final moments of final approach after the human pilot disengaged the autopilot and took full control of the aircraft) when attempting to maintain the standard 3 degrees glideslope during final approach in calm weather. Fig. 25 and 26 illustrate a comparison between the IAS and the standard autopilot (Autoland) when attempting to maintain the standard 3 degrees glideslope during final approach in extreme weather conditions with the presence of strong wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees (around 360 degrees), and turbulence. Table XVII shows the result of applying the Two One-Sided Test (TOST) to examine the equivalence of the glideslope degrees held by the IAS, the standard autopilot, and the human pilot in calm weather. Table XVIII shows the result of applying the Two One-Sided Test (TOST) to examine the equivalence of the glideslope degrees held by the IAS and the standard autopilot (Autoland) in extreme weather.

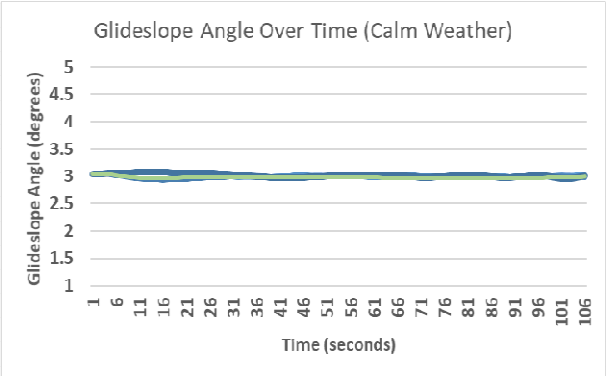

Fig. 24. A comparison between the IAS (10 flights represented by the overlapping lines in different blue shades), the standard autopilot, and the human pilot after he took full control of the aircraft during the last moments of final approach (1 demonstration represented by the green line) when maintaining the three degrees glideslope angle from final approach to landing in calm weather.

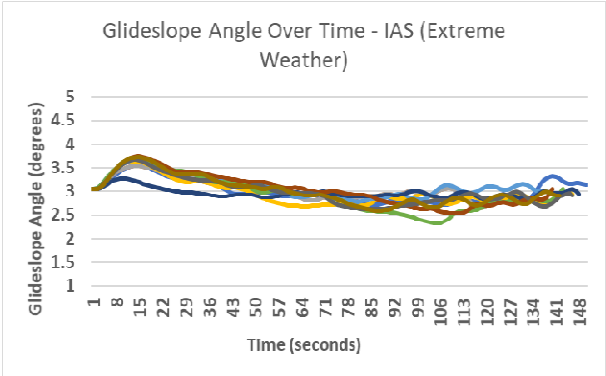

Fig. 25. The glideslope angle of the aircraft (flown by the IAS) from final approach to landing. The goal is to try to maintain the standard 3 degrees glideslope. The weather conditions include 360 degrees wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and minor turbulence.

TABLE XVII
RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED TO THE STANDARD AUTOPILOT AND THE HUMAN PILOT WHEN MAINTAINING A THREE DEGREES GLIDESLOPE DURING FINAL APPROACH IN CALM WEATHER.

Equivalence Test for Means
Unequal Sample Sizes
$\alpha = 0.05$

	IAS	**AP/Human**
Mean	3.02	2.99
Variance	0.0009	0.0002
Observations	1059	106
Pooled Variance	0.0009	
Hypothesized Mean Difference	0.8	
df	1163	
t Stat	248.07	-268.5
P(T<=t) one-tail	0	0
T Critical one-tail	1.64	
P(T<=t) two-tail	0	
T Critical Two-tail	1.96	

Means are Equivalent because p1 & p2 < 0.05

TABLE XVIII
RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED TO THE STANDARD AUTOPILOT WHEN MAINTAINING A THREE DEGREES GLIDESLOPE DURING FINAL APPROACH IN EXTREME WEATHER CONDITIONS.

Equivalence Test for Means
Equal Sample Sizes
$\alpha = 0.05$

	IAS	**AP**
Mean	3.03	2.93
Variance	0.07	0.02
Observations	1429	1429
Pooled Variance	0.05	
Hypothesized Mean Difference	0.80	
df	2856.00	
t Stat	89.52	-110.65
P(T<=t) one-tail	0.00	0.00
T Critical one-tail	1.65	
P(T<=t) two-tail	0.00	
T Critical Two-tail	1.96	

Means are Equivalent because p1 & p2 < 0.05

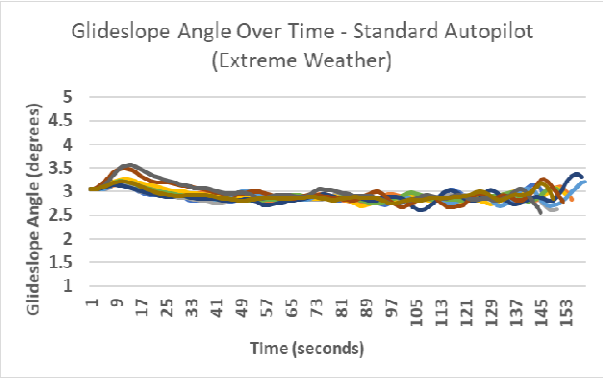

Fig. 26. The glideslope angle of the aircraft (flown by the standard autopilot) from final approach to landing. The goal is to try to maintain the standard 3 degrees glideslope. The weather conditions include 360 degrees wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and minor turbulence.

G. Runway Centreline Maintenance

Four models were generated for the Roll ANN, the Ailerons ANN, the Heading ANN, and the Rudder ANN with MSE values of 0.0002, 0.001, 0.003, and 0.002 consecutively. Fig. 27 illustrates a comparison between the IAS, the standard autopilot of the aircraft model, and the human pilot (the final moments of final approach after the human pilot disengaged the autopilot and took full control of the aircraft) when attempting to maintain the centreline of the landing runway in calm weather. Table XIX shows the result of applying the Two One-Sided Test (TOST) to examine the equivalence of the angle between the aircraft and the landing runway held by the IAS, the standard autopilot, and the human pilot in calm weather. Fig. 28 shows the angle between the aircraft's location and the centreline of the landing runway before landing in extreme weather conditions with the presence of 90 degrees crosswind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and strong turbulence. In the latter weather conditions, the standard autopilot kept disengaging every time, therefore, the comparison is given between the current and the old versions of the IAS. The previous version of the IAS was able to handle severe weather conditions with wind speed up to 50 knots, and a maximum wind shear of around 22 degrees [4].

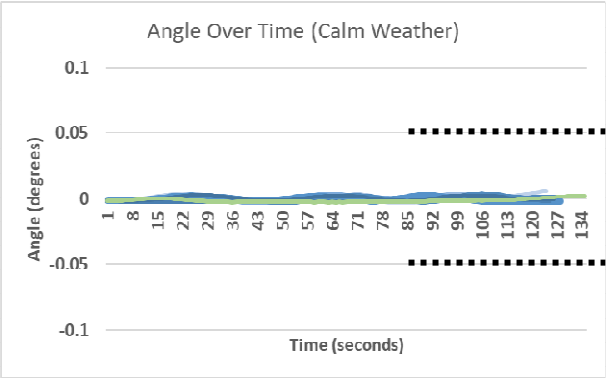

Fig. 27. A comparison between the IAS (10 flights represented by the overlapping lines in different blue shades), the standard autopilot, and the human pilot after he took full control of the aircraft during the last moments of final approach (1 demonstration represented by the green line) when maintaining the centreline of the landing runway (0 degrees) during final approach (airborne) and landing (on the ground after touchdown) in calm weather. The angle must be between 0.05 and -0.05 degrees especially during the last moments of the final approach to ensure landing within the safe touchdown zone of the landing runway as the two dashed black lines show (right part of the chart).

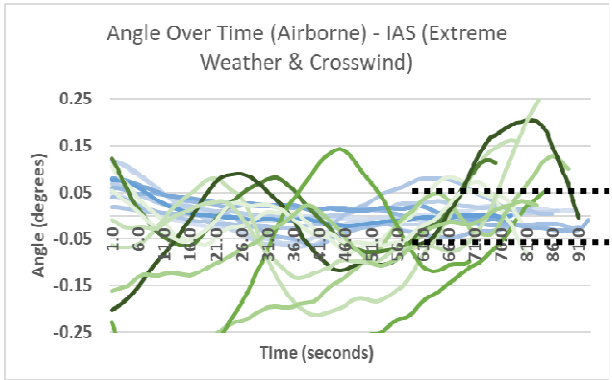

Fig. 28. A comparison between the current version of the IAS (represented by the overlapping lines in different blue shades) and the previous version of the IAS (represented by the lines in different green shades) during 10 flights each when maintaining the centreline of the landing runway (0 degrees) during final approach (airborne). The angle must be between 0.05 and -0.05 degrees especially during the last moments of the final approach to ensure landing within the safe touchdown zone of the landing runway as the two dashed black lines show (right part of the chart). The extreme weather conditions include 90 degrees crosswind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and strong turbulence.

However, to perform a comparison between the IAS and the Autoland feature of the standard autopilot without facing the disengagement issue, the weather conditions were slightly modified by replacing the 90 degrees crosswind direction with 360 degrees, and lowering the intensity of turbulence. Fig. 29 and 30 illustrate a comparison between the IAS and the standard autopilot when attempting to intercept the centreline of the landing runway (airborne) in the slightly modified weather conditions. Table XX shows the number of successful and unsuccessful attempts to keep the aircraft within the safe zone (angle between 0.05 and -0.05 degrees) during final approach while airborne. Fig. 31 and 32 illustrate a comparison between the IAS and the standard autopilot when attempting to intercept the centreline of the landing runway after touchdown in extreme weather conditions with the presence of strong wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees (around 0 degrees), turbulence, and high precipitation (wet runway). Table XX shows the number of successful and unsuccessful attempts to keep the aircraft within the safe zone of the runway (angle between 0.05 and -0.05 degrees) after touchdown while attempting to decrease the speed to taxi speed.

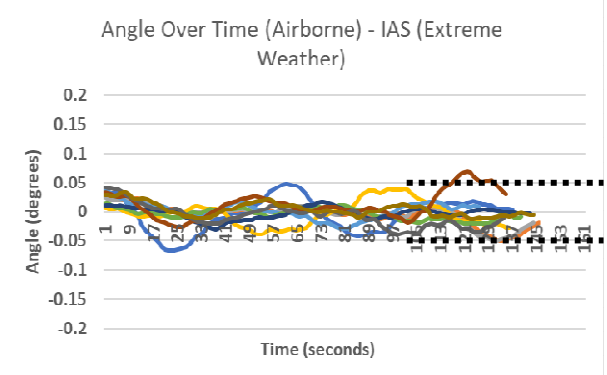

Fig. 29. The angle between the aircraft (flown by the IAS) and the centreline of the runway (0) during ten different final approach attempts (airborne). The angle must be between 0.05 and -0.05 degrees especially during the last moments of the final approach to ensure landing within the safe touchdown zone of the landing runway as the two dashed black lines show (right part of the chart). The weather conditions include 360 degrees wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and minor turbulence.

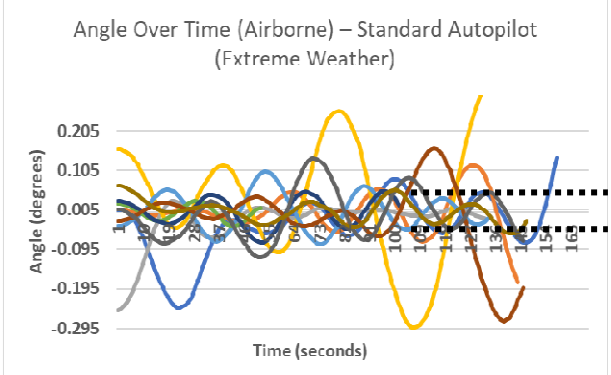

Fig. 30. The angle between the aircraft (flown by the standard autopilot) and the centreline of the runway (0) during ten different final approach attempts (airborne). The angle must be between 0.05 and -0.05 degrees especially during the last moments of the final approach to ensure landing within the safe touchdown zone of the landing runway as the two dashed black lines show (right part of the chart). The weather conditions include 360 degrees wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and minor turbulence.

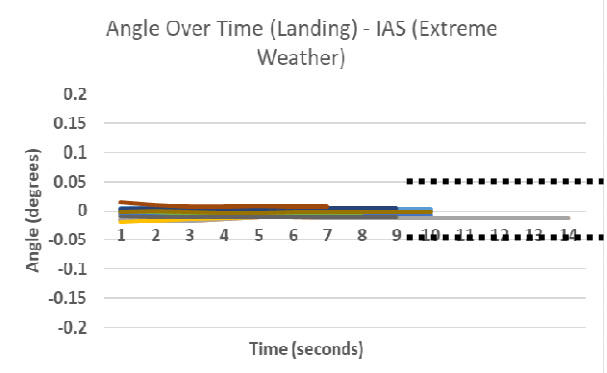

Fig. 31. The angle between the aircraft (flown by the IAS) and the centreline of the runway (0 degrees) during ten different landing attempts. The angle must be between 0.05 and -0.05 degrees during touchdown to ensure landing within the safe touchdown zone of the landing runway as the two dashed black lines show (right part of the chart), and during the attempt to decrease the aircraft's speed on the runway to taxi speed. The weather conditions include 360 degrees wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and minor turbulence.

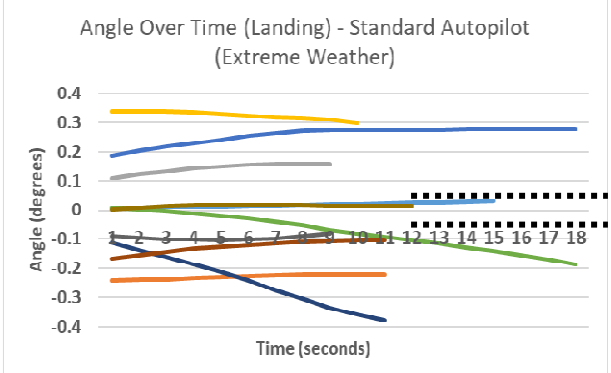

Fig. 32. The angle between the aircraft (flown by the standard autopilot) and the centreline of the runway (0 degrees) during ten different landing attempts. The angle must be between 0.05 and -0.05 degrees during touchdown to ensure landing within the safe touchdown zone of the landing runway as the two dashed black lines show (right part of the chart), and during the attempt to decrease the aircraft's speed to taxi speed. The weather conditions include 360 degrees wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and minor turbulence.

TABLE XIX

RESULTS OF APPLYING THE EQUIVALENCE TEST TO EXAMINE THE PERFORMANCE OF THE IAS COMPARED TO THE STANDARD AUTOPILOT AND THE HUMAN PILOT WHEN MAINTAINING THE CENTRELINE OF THE LANDING RUNWAY (0 DEGREES ANGLE) DURING FINAL APPROACH AND LANDING IN CALM WEATHER.

Equivalence Test for Means

Unequal Sample Sizes

$\alpha = 0.05$

	IAS	**AP/Human**
Mean	0.00004	-0.00002
Variance	0.000	0.000
Observations	1246	135
Pooled Variance	0.000	
Hypothesized Mean Difference	0.8	
df	1379	
t Stat	4926.46	-4944.38
P(T<=t) one-tail	0.000	0.000
T Critical one-tail	1.64	
P(T<=t) two-tail	0.000	
T Critical Two-tail	1.96	

Means are Equivalent because p1 & p2 < 0.05

TABLE XX

RESULTS OF COMPARING THE CURRENT AND THE OLD VERSION OF THE IAS WHEN ATTEMPTING TO MAINTAIN THE CENTRELINE OF THE RUNWAY DURING THE FINAL MOMENTS OF FINAL APPROACH (AIRBORNE) IN EXTREME WEATHER CONDITIONS INCLUDING 90 DEGREES CROSSWIND AT A SPEED OF 50 KNOTS WITH GUST UP TO 70 KNOTS, WIND SHEAR DIRECTION OF 70 DEGREES, AND STRONG TURBULENCE. SUCCESSFUL ATTEMPTS ARE WITHIN THE ANGLE THRESHOLD BETWEEN 0.05 AND -0.05 AND VICE VERSA.

	Runway centreline maintenance (airborne)	
Pilot	**Successful**	**Unsuccessful**
The IAS (current version)	10 out of 10	0 out 10
The IAS (old version)	4 out of 10	6 out of 10

TABLE XXI

RESULTS OF COMPARING THE IAS WITH THE STANDARD AUTOPILOT WHEN ATTEMPTING TO MAINTAIN THE CENTRELINE OF THE RUNWAY DURING THE FINAL MOMENTS OF FINAL APPROACH (AIRBORNE) AND AFTER TOUCHDOWN WHILE DECREASING THE SPEED OF THE AIRCRAFT TO TAXI SPEED ON THE LANDING RUNWAY IN EXTREME WEATHER CONDITIONS. SUCCESSFUL ATTEMPTS ARE WITHIN THE ANGLE THRESHOLD BETWEEN 0.05 AND -0.05 AND VICE VERSA.

	Runway centreline maintenance (airborne)		**Runway centreline maintenance (ground)**	
Pilot	**Successful**	**Unsuccessful**	**Successful**	**Unsuccessful**
The IAS	20 out of 20	0 out 20	10 out of 10	0 out of 10
Standard Autopilot	5 out of 20	15 out of 20	2 out of 10	8 out of 10

VI. Analysis

As can be seen in Fig. 7 (A. Takeoff Pitch Maintenance), the IAS was able to maintain the standard pitch angle of 15 degrees during the takeoff phase. Table II shows that the IAS was able to maintain a pitch angle mean of 15.24 degrees which is equivalent to the 15.17 degrees mean maintained by the human pilot as the equivalence test shows.

Fig. 8, 9, and 10 (B. Altitude Maintenance) show that the IAS was able to maintain three different altitudes at three different speeds as did the standard autopilot. Tables III, IV, and V show that the performance of the IAS when maintaining a given altitude at a given speed is equivalent to the performance of the standard autopilot. Fig. 11 shows the significant improvement in the ability of maintaining a given altitude by comparing the previous version of the IAS which was not able to accurately maintain altitudes with the current version which now have the ability to handle this task precisely.

Fig. 12, 13, 14, 15, 16, and 17 (C. Climb Rate Maintenance), and tables VI, VII, VIII, IX, X, and XI show that the IAS performed better than the standard autopilot when maintaining six different climb/sink rates at six different speeds.

Fig. 18, 19, and 20 illustrate the equivalent performances of the IAS and the standard autopilot when maintaining three different speeds at three different altitudes. Tables XII, XIII, and XIV confirm the equivalence between the performances of the IAS and the standard autopilot when handling this task. Fig. 21 shows that the IAS was able to manage and maintain the different speeds in the different flight phases from takeoff to landing in a manner that is identical to the human pilot throughout the same flight.

Fig. 22 and 23 (E. Flaps Setting) illustrate the consistent behaviour of the IAS when extending and retracting the flaps given the flight phase and altitude, which is identical to the behaviour of the human pilot when handling this task. The minor differences shown in table XVI are due to the terrain variation below the aircraft since the applied altitude here is feet above ground level instead of sea level.

Fig. 24 (F. Final Approach Glideslope Maintenance) shows the identical performance of the IAS, the standard autopilot, and the human pilot when maintaining the standard 3 degrees glideslope angle during final approach and landing in calm weather. Table XVII confirms the equivalence between the performance of the IAS, the standard autopilot, and the human pilot when handling this task. Fig. 25 and 26 show the similar performance of the IAS and the standard autopilot (Autoland) while maintaining the standard 3 degrees glideslope angle in extreme weather conditions including 360 degrees wind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and minor turbulence. Table XVIII shows that the means of the glideslope angle maintained by the IAS and the standard autopilot are equivalent, however, the IAS performed better since the glideslope mean is 3.01 which is significantly closer to the desired 3 degrees glideslope compared to the 2.93 mean achieved by the standard autopilot.

As can be seen in Fig. 27 (G. Runway Centreline Maintenance), the IAS was able to maintain the centreline of the landing runway as did the human pilot and the standard autopilot in calm weather. Table XIX confirms the equivalence between the performance of the IAS, the human pilot, and the standard autopilot when handling this task. Fig. 28 and 29 show that the IAS was able to keep the aircraft within the safe zone (between 0.05 and -0.05 degrees from the centreline of the runway) in extreme weather conditions including 90 degrees crosswind at a speed of 50 knots with gust up to 70 knots, wind shear direction of 70 degrees, and strong turbulence, while the standard autopilot kept disengaging every time in the latter weather conditions. Compared to the previous version of the IAS which achieved a success rate of 40% (4 successful attempts out of 10 trials), the current version achieved a success rate of 100% (10 successful attempts out of 10 trials) as table XX and Fig. 28 show when intercepting the centreline of the landing runway in such extreme weather conditions. After altering the weather conditions by replacing the 90 degrees crosswind with 360 degrees wind and lowering the intensity of turbulence, the standard autopilot was able to land, however, as Fig. 30 shows, the standard autopilot struggled to keep the aircraft within the safe zone (between 0.05 and -0.05 degrees from the centreline of the runway). Table XXI shows that the IAS was able to achieve a success rate of 100% (20 successful attempts out of 20 trials), while the standard autopilot achieved a success rate of 25% (5 successful attempts out of 20 trials) which confirms that the IAS can perform significantly beyond the capabilities of modern standard autopilots in extreme weather conditions. In addition, Fig. 31 illustrates the excellent performance of the IAS while trying to keep the aircraft within the safe zone of the landing runway (between 0.05 and -0.05 degrees from the centreline of the runway) after touchdown while decreasing the speed of the aircraft on the runway to taxi speed, while Fig. 32 illustrates the poor performance of the standard autopilot when attempting to handle the same task in the same extreme weather conditions. In addition, table XXI shows that the IAS was able to achieve a success rate of 100% (10 successful attempts out of 10 trials), while the standard autopilot achieved a success rate of 20% (2 successful attempts out of 10 trials) while maintaining the centreline of the landing runway after landing, which further confirms the superior performance of the IAS which is beyond the capabilities of standard autopilots.

Overall, the distinct performance of the IAS, which shows a natural and dynamic behaviour when handling the different tasks by manipulating the different control surfaces especially in extreme weather conditions proved its superiority compared to the mechanical-precision-like performance of the standard autopilot, which according to the literature, suffers from robustness issues when facing uncertainty, which hinders the reaction time, and the ability to cope with such extreme and sometimes sudden conditions.

VII. EVALUATING THE IAS BY OMAN AIR

To involve the aviation industry in evaluating the performance of the IAS, and in addition to providing training data for the IAS, Captain Khalid Al Hashmi, Oman Air, provided his feedback after being presented with complete (airport to airport) flight demonstrations of the IAS, and landings in calm and extreme weather conditions as the experiments above show. We asked him the following questions, and he answered as follows:

1. Compared to the standard modern autopilot, what is your impression of the performance of the IAS when executing complete flights in calm and severe weather conditions? "*Good. I Wish we can try the IAS in a 6-axis full motion flight simulator to evaluate it further.*"
2. Although flying in such conditions is probably against regulations, but for testing purposes, is the IAS capable of preforming crosswind landings beyond the current limits and capabilities of modern autopilots? What about experienced human pilots? "*Yes. It is always a challenge, human pilots are allowed to land in crosswind conditions up to the demonstrated limit such as 38 knots, whereas the autopilots limit is less. I Hope that the IAS can help in increasing the crosswind limit which is sometimes limited due to flight controllability rather than pure capability.*"
3. Is the current performance of the IAS in general comparable with human pilots? If yes, as an experienced captain and instructor, how would you rate its performance if it were human? novice, intermediate, or experienced? "*Yes, I would say intermediate although I suggest comparing it more with other autopilot.*"
4. Do you agree that the IAS has the potential to introduce new advantages to the aviation industry such as enhancing safety as a dependable autopilot compared to the modern ones? "*Yes, it does. It just needs to be trained more on scenarios and various conditions and malfunctions.*"

VIII. CONCLUSION

In this work, a novel and robust approach is proposed to "teach" the Intelligent Autopilot System (IAS) how to perform the necessary set of piloting tasks while flying from one airport to another in a manner that is comparable to experienced human pilots of airliners. Compared to the previous versions of the IAS, the newly designed and trained ANNs can now handle tasks including maintaining the desired pitch angle during takeoff, maintaining different altitudes, climb/sink rates, speeds, and controlling the flaps by manipulating the appropriate control surfaces. This approach introduces the possibility to have an autopilot that behaves like a skilled human pilot rather than a machine with limited capabilities. In addition, the newly acquired abilities include performing landings in extreme weather conditions with extreme crosswind, gust, shear, and turbulence beyond the current limits and abilities, which increases safety significantly. Exploiting Supervised Learning, and applying Artificial Neural Networks proved to be an effective approach to train the IAS how to handle such conditions as parts of the overall ability of the IAS to perform complete flights from takeoff to landing autonomously with minimum effort. The experiments were strong indicators towards the ability of Supervised Learning with Artificial Neural Networks to capture low-level piloting tasks such as the rapid manipulation of the elevators and the elevators trim to maintain the required takeoff pitch angle, different altitudes, different climb/sink rates, and the rapid manipulation of the ailerons, rudder, and the elevators trim to intercept the centreline of the landing runway, and to maintain the required three degrees glideslope during final approach. Furthermore, the experiments were strong indicators towards the ability of the proposed approach to capture high-level tasks as well such as extracting and retracting the flaps according to the flight phase and altitude.

Breaking down the piloting tasks and adding more Artificial Neural Networks allows the system to overcome the black-box problem by having multiple small ANNs with single-hidden-layers that learn from small labelled datasets which have clear patterns. In addition, this approach enhanced performance and accuracy, and allowed the coverage of a wider spectrum of tasks.

The aviation industry is currently working on solutions which would lead to decreasing the dependence on human pilots. The reason behind this is to lower the workload, human error, stress, and handle the pilots shortage problem compared to the high demand for new airplanes, by developing autopilots capable of performing complete flights without human intervention. We anticipate that future autopilot systems which make of methods proposed here could improve safety and handle the challenges faced by the industry.

ACKNOWLEDGEMENT

We wish to acknowledge Oman Air, Oman Air Flight Training Centre, and Captain Khalid Al Hashmi, Senior Manager Crew Training at Oman Air for providing us with their valuable time, facilities, and support which allowed for the collection of the necessary training data to train the Intelligent Autopilot System (IAS) as well as evaluating it. This collaboration contributed to the preparation of this research paper.

REFERENCES

[1] R. Khan-Persaud, "ECCAIRS Aviation 1.3.0.12 (VL for ATTrID 391 - Event Phases)", *ICAO Safety*, 2013. [Online]. The International Civil Aviation Organization (ICAO). Available: https://www.icao.int/safety/airnavigation/AIG/Documents.

[2] H. Baomar and P. J. Bentley, "An Intelligent Autopilot System that learns piloting skills from human pilots by imitation," *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, Arlington, VA, USA, 2016, pp. 1023-1031.

[3] H. Baomar and P. J. Bentley, "An Intelligent Autopilot System that learns flight emergency procedures by imitating human pilots," *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, Athens, 2016, pp. 1-9.

[4] H. Baomar and P. J. Bentley, "Autonomous landing and go-around of airliners under severe weather conditions using Artificial Neural Networks," *2017 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS)*, Linkoping, 2017, pp. 162-167.

[5] H. Baomar and P. J. Bentley, "Autonomous navigation and landing of large jets using Artificial Neural Networks and learning by imitation," *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, Honolulu, HI, 2017, pp. 1-10.

[6] D. We, H. Xiong and J. Fu, "Aircraft Autopilot Pitch Control Based on Fuzzy Active Disturbance Rejection Control," *2015 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration*, Wuhan, 2015, pp. 144-147.

[7] Beygi, Nima & Beigy, Maani & Siahi, Mehdi. (2015). Design of Fuzzy self-tuning PID controller for pitch control system of aircraft autopilot.

[8] H. L. Jeevan, H. K. Narahari and A. T. Sriram, "Development of pitch control subsystem of autopilot for a fixed wing unmanned aerial vehicle," *2018 2nd International Conference on Inventive Systems and Control (ICISC)*, Coimbatore, 2018, pp. 1233-1238.

[9] Z. Motea, H. Wahid, J. Zahid, S. Lwin, and A. Hassan. (2018). A Comparative Analysis of Intelligent and PID Controllers for an Aircraft Pitch Control System. *APPLICATIONS OF MODELLING AND SIMULATION (AMS)*, 2(1), pp.17-25.

[10] J. Qiu, I. S. Delshad, Q. Zhu, M. Nibouche and Y. Yao, "A U-model based controller design for non-minimum phase systems: Application to Boeing 747 altitude-hold autopilot," *2017 9th International Conference on Modelling, Identification and Control (ICMIC)*, Kunming, 2017, pp. 122-127.

[11] J. Muliadi, and B. Kusumoputro. (2018). Neural Network Control System of UAV Altitude Dynamics and Its Comparison with the PID Control System. *Journal of Advanced Transportation*, 2018, pp.1-18.

[12] N. C. Mumm and F. Holzapfel, "Vertical speed command performance improvement of a load factor command based autopilot for automatic landing by shaping the desired command during flare," *2017 IEEE Conference on Control Technology and Applications (CCTA)*, Mauna Lani, HI, 2017, pp. 1117-1122.

[13] P. Stukenborg, and R. Luckner. (2018). Evaluating the influence of continuous flap settings on the take-off performance of an airliner using flight simulation. *CEAS Aeronautical Journal*, 9(4), pp.671-681.

[14] J. Dong, D. Zhou, C. Shao, and S. Wu. (2018). Nonlinear system controllability analysis and autopilot design for bank-to-turn aircraft with two flaps. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 233(5), pp.1772-1783.

[15] Oszust, M., Kapuscinski, T., Warchol, D., Wysocki, M., Rogalski, T., Pieniazek, J., Kopecki, G., Ciecinski, P. and Rzucidlo, P. (2018). A vision-based method for supporting autonomous aircraft landing. Aircraft Engineering and Aerospace Technology, 90(6), pp.973-982.

[16] Theis, J., Ossmann, D., Thielecke, F. and Pfifer, H. (2018). Robust autopilot design for landing a large civil aircraft in crosswind. Control Engineering Practice, 76, pp.54-64.

[17] de Bruin, A. and Jones, T. (2016). Accurate Autonomous Landing of a Fixed-Wing Unmanned Aircraft under Crosswind Conditions. IFAC-PapersOnLine, 49(17), pp.170-175.

[18] Zhang, L., Zhai, Z., He, L. and Niu, W. (2019). Infrared-Based Autonomous Navigation for Civil Aircraft Precision Approach and Landing. IEEE Access, 7, pp.28684-28695.

[19] Bian, Q., Nener, B. and Wang, X. (2018). Control parameter tuning for aircraft crosswind landing via multi-solution particle swarm optimization. Engineering Optimization, 50(11), pp.1914-1925.

[20] Viswanathan, R. and Lakshmi, P. (2017). A Novel Method for Controlling the Roll Angle of Aircraft using Sliding Mode Control Methodology. Journal of The Institution of Engineers (India): Series C, 99(4), pp.369-372.

[21] S. U. Ali, R. Samar and M. Z. Shah, "UAV lateral path following: Nonlinear sliding manifold for limited actuation," *2017 36th Chinese Control Conference (CCC)*, Dalian, 2017, pp. 1348-1353.

[22] Nikolai Botkin, Varvara Turova. Aircraft Runway Acceleration in the Presence of Severe Wind Gusts. 27th IFIP Conference on System Modeling and Optimization (CSMO), Jun 2015, Sophia Antipolis, France. pp.147-158.

[23] Lungu, R. and Lungu, M. (2017). Automatic landing system using neural networks and radio-technical subsystems. Chinese Journal of Aeronautics, 30(1), pp.399-411.

[24] Evdokimenkov V., Kim R., Krasilshchikov M., Sebrjakov G. (2016) Individually Adapted Neural Network for Pilot's Final Approach Actions Modeling. In: Cheng L., Liu Q., Ronzhin A. (eds) Advances in Neural Networks – ISNN 2016. ISNN 2016.

[25] E. Alvis, D. Bhatt, B. Hall, K. Driscoll, A. Murugesan. Aerospace Advanced Technology Labs, Honeywell International, Inc. (2018). Final Technical Report for NASA Project. Assurance Reasoning for Increasingly Autonomous Systems (ARIAS). [online] Available at: https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20180006312.pdf

[26]Abbeel, P., & Ng, A. Y. (2004). Apprenticeship learning via inverse reinforcement learning. In proc of the twenty-first international conference on Machine learning (p. 1).

[27] F. Wei, A. Bower, L., Gates, A., Rose, and D. T. Vasko, "The Full-Scale Helicopter Flight Simulator Design and Fabrication at CCSU", *57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2016.

[28] M. Jirgl, J. Boril, and R. Jalovecky, "The identification possibilities of the measured parameters of an aircraft model and pilot behavior model on the flight simulator". *International Conference on Military Technologies (ICMT)*, 2015, vol., no., pp.1-5.

[29] A. Kaviyarasu, and S. Kumar, "Simulation of Flapping-wing Unmanned Aerial Vehicle using X-plane and Matlab/Simulink". "*Defence Science Journal"*, 2014, 64(4), pp.327-331.

[30] J. McCaffrey, "Understanding Neural Networks using .NET", [Presentation]. *The Microsoft 2014 Build Conference*, San Francisco, USA, 2014.

[31] J. Heaton, *Artificial Intelligence for Humans, Volume 3: Deep Learning and Neural Networks*. St. Louis, MO, USA: Heaton Research, Inc., 2015.

[32] K. Winter, I. J. Hayes, and R. Colvin, "Integrating Requirements: The Behavior Tree Philosophy," 2010 8th IEEE International Conference on Software Engineering and Formal Methods, Pisa, 2010, pp. 41-50.

[33] E. L'hotellier and J. Salzmann. (2017). *Top of descent calculation*. [online] The International Virtual Aviation Organisation, IVAO. Available at: https://www.ivao.aero/training/documentation/books/SPP_APC_Top_of_descent.pdf.

[34] D. J. Schuirmann. (1987). "A comparison of the Two One-Sided Tests Procedure and the Power Approach for assessing the equivalence of average bioavailability". *Journal of Pharmacokinetics and Biopharmaceutics*. 15 (6): 657–680.