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.

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].

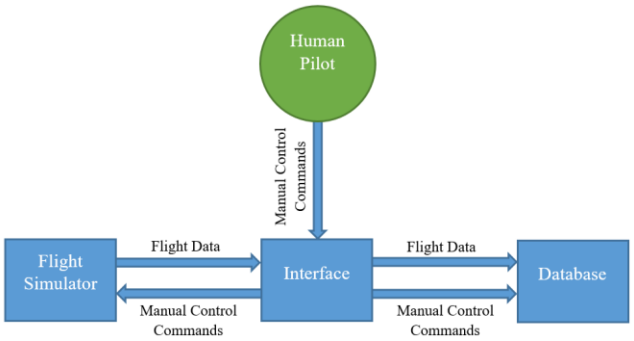

Fig. 1. Block diagram illustrating IAS components used during the pilot data collection step.

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.

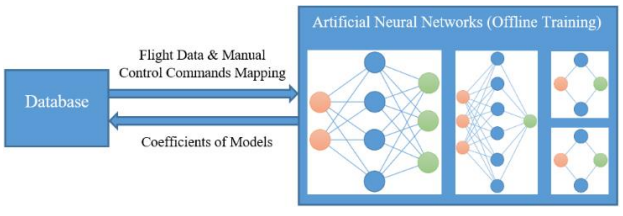

Fig. 2. Block diagram illustrating IAS components used during training.

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).

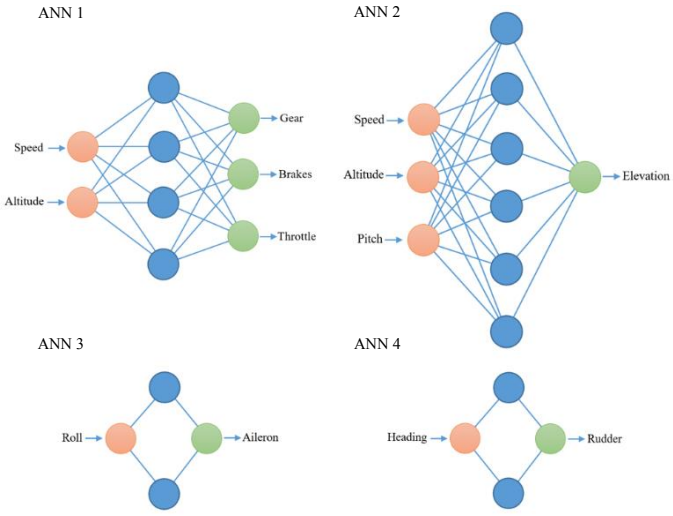

Fig. 3. Topology of ANN 1 trained to handle throttle, gear and brakes (top left), topology of ANN 2 trained to handle elevator control (top right), topology of ANN 3 trained to handle aileron control (bottom left), and topology of ANN 4 trained to handle rudder control (bottom right).

[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.

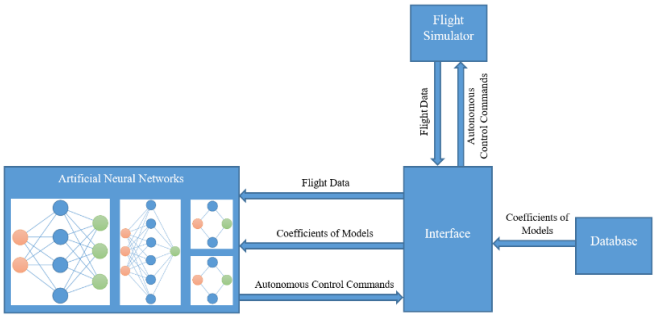

Fig. 4. Block diagram illustrating IAS components used during autonomous control.

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.

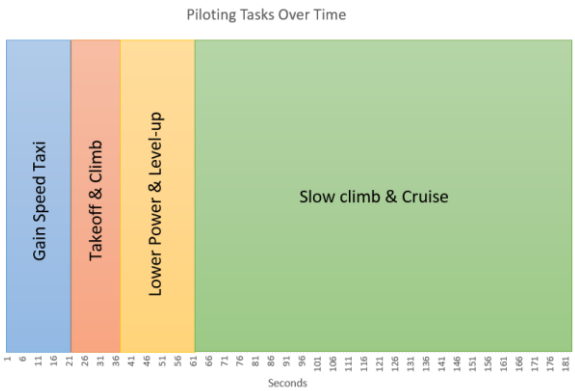

Fig. 5. Piloting tasks over time.

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

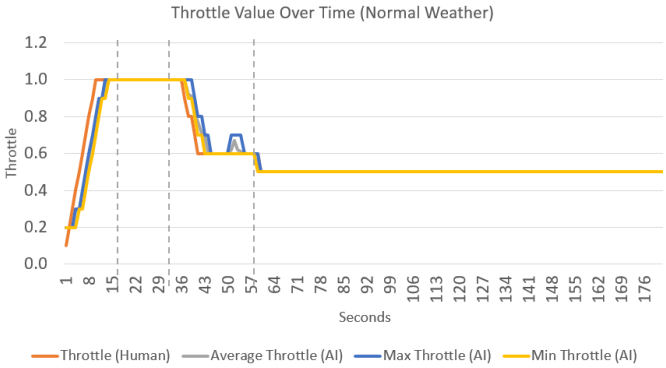

Fig. 6. (Exp. 1) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum throttle commands over time during the four phases –separated by dotted lines- as illustrated in Fig. 4.

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

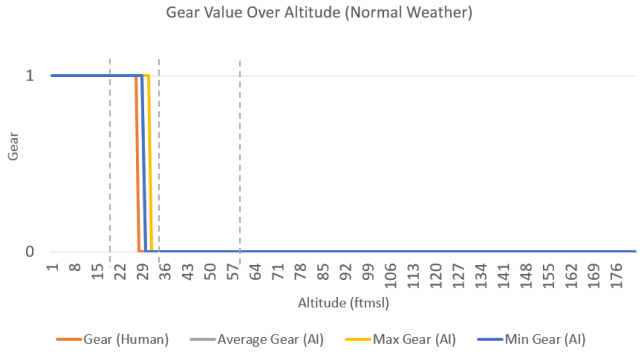

Fig. 7. (Exp. 1) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum gear commands over time.

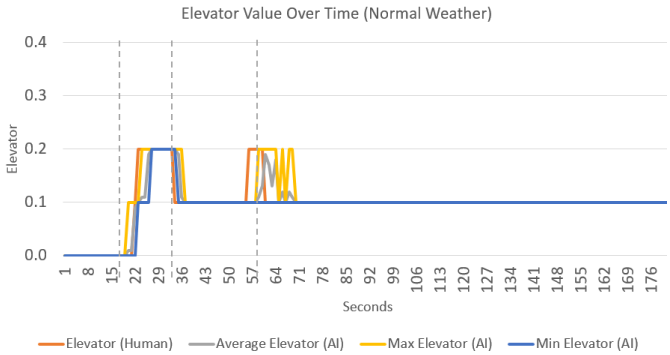

Fig. 8. (Exp. 1) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum elevator commands over time.

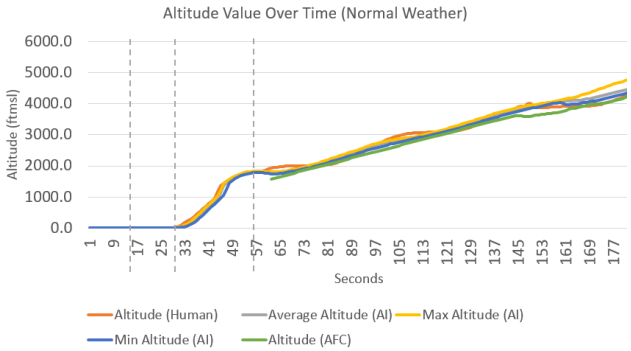

Fig. 9. (Exp. 1) A comparison between the human pilot, the aircraft's AFC/Autopilot, and the Intelligent Autopilot's average, maximum, and minimum altitude over time.

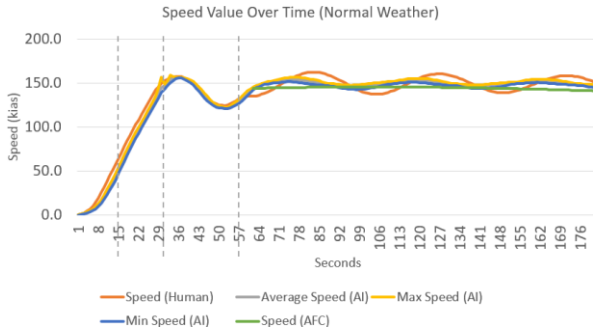

Fig. 10. (Exp. 1) A comparison between the human pilot, the aircraft's AFC/Autopilot, and the Intelligent Autopilot's average, maximum, and minimum speed over time.

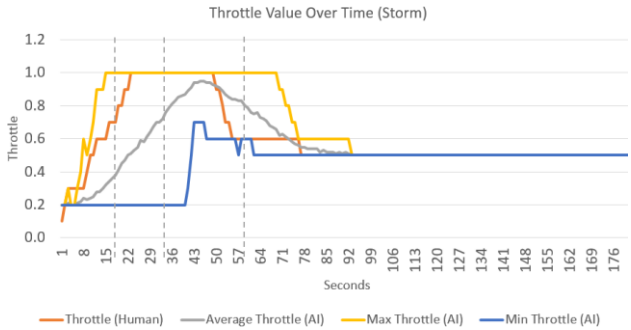

Fig. 11. (Exp. 2) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum throttle commands over time during the four phases –separated by dotted lines- as illustrated in Fig. 4.

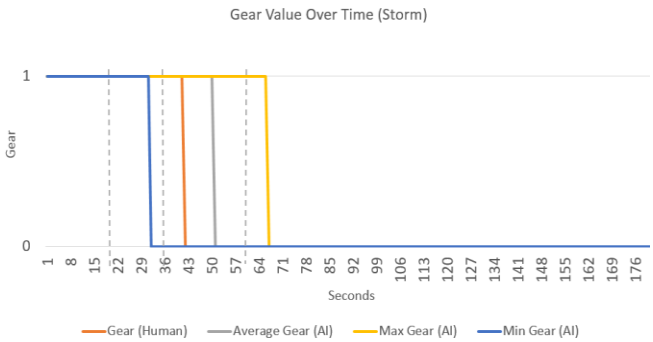

Fig. 12. (Exp. 2) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum gear commands over time.

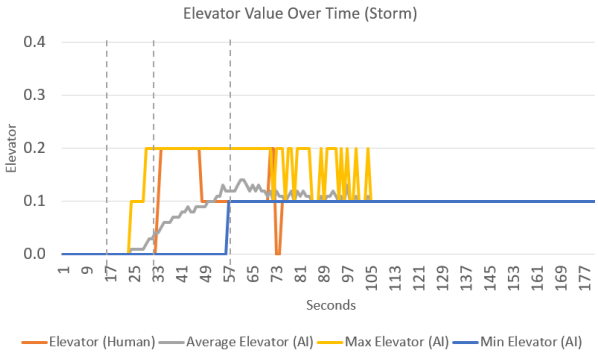

Fig. 13. (Exp. 2) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum elevator commands over time.

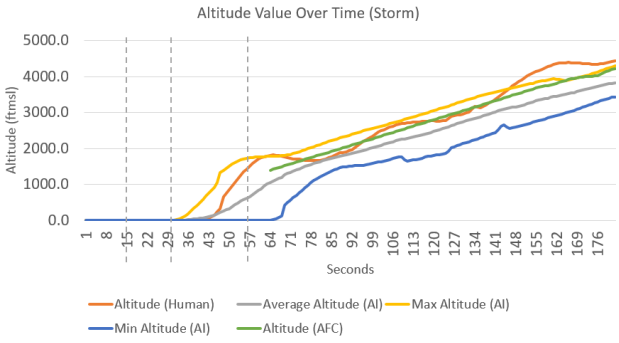

Fig. 14. (Exp. 2) A comparison between the human pilot, the aircraft's AFC/Autopilot, and the Intelligent Autopilot's average, maximum, and minimum altitude over time.

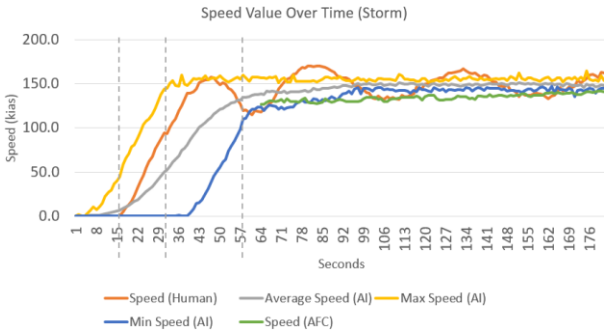

Fig. 15. (Exp. 2) A comparison between the human pilot, the aircraft's AFC/Autopilot, and the Intelligent Autopilot's average, maximum, and minimum speed over time.

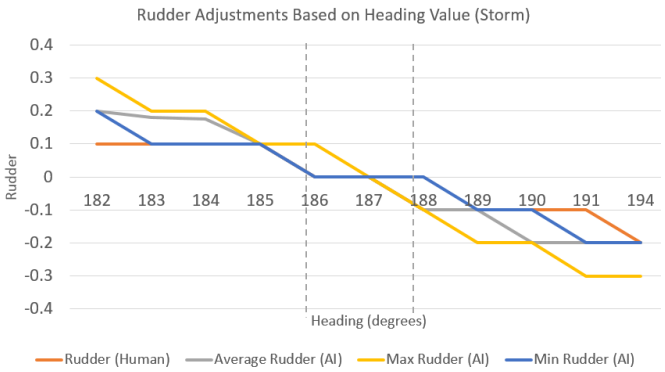

Fig. 16. (Exp. 2) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum heading correction attempts. The middle part between the two dotted lines is the area where no corrections are required (based on a heading of 187 degrees). The right part illustrates a deviation in heading towards the right, while the left part illustrates a deviation in heading towards the left.

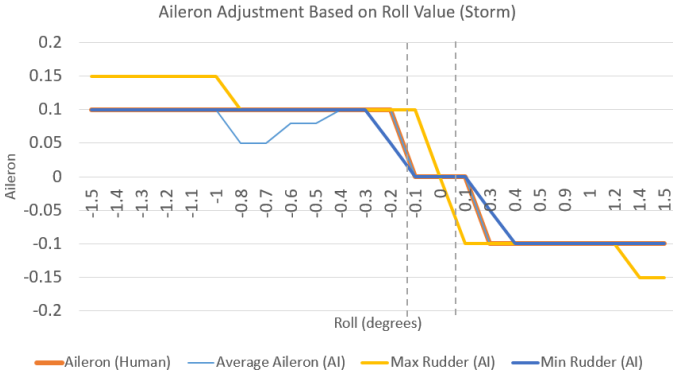

Fig. 17. (Exp. 2) A comparison between the human pilot and the Intelligent Autopilot's average, maximum, and minimum roll correction attempts. The middle part between the two dotted lines is the area where no corrections are required. The right part illustrates a deviation in roll towards the right, 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.