Autonomous Navigation and Landing of Large Jets 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 aircrafts 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.

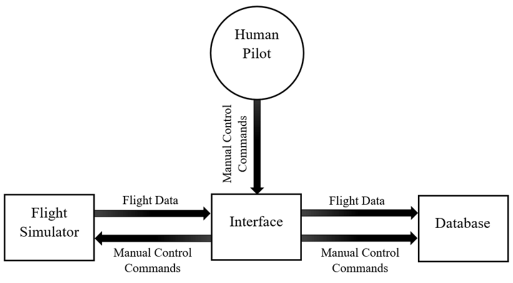

Fig. 1. Block diagram illustrating the IAS components used during the pilot data collection step.

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.

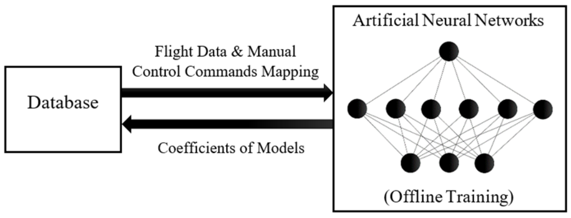
Artificial Neural Networks
Flight Data & Manual Control Commands Mapping
Database
Coefficients of Models
(Offline Training)

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

[1] X-Plane 10 Global. http://www.x-plane.com

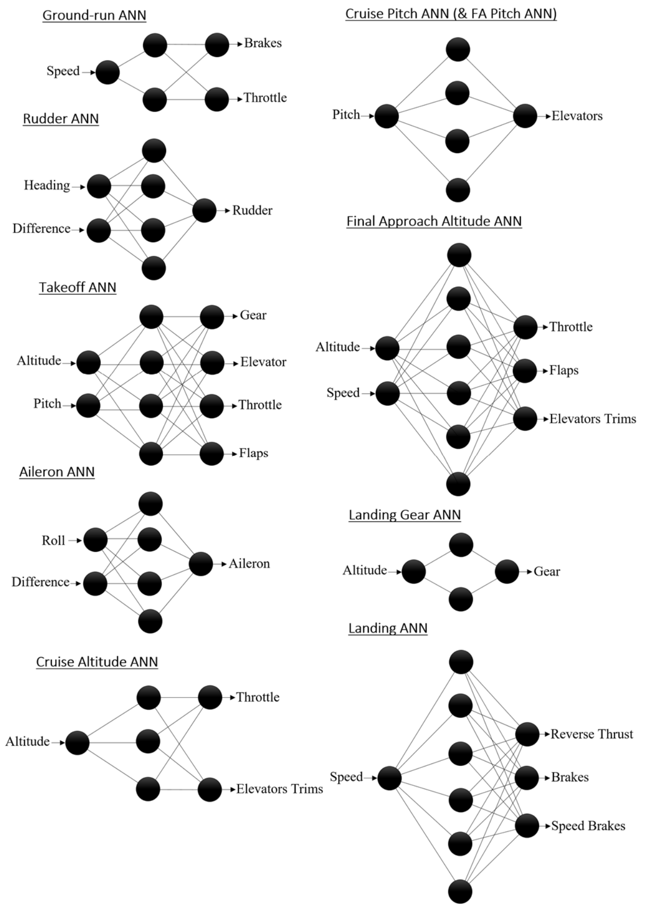

Fig. 3. Inputs, outputs, and the topologies of the ANNs relevant to this work. Each ANN is designed and trained to handle a specific task.

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) [18].

$$\Delta w_{(t)} = -\epsilon \frac{\partial E}{\partial w_{(t)}} + \alpha \Delta w_{(t-1)} \quad (5)$$

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.

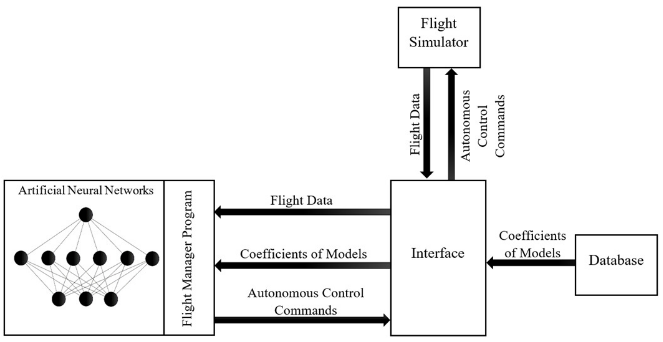

Fig. 4. Block diagram illustrating the IAS components used during autonomous control.

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.

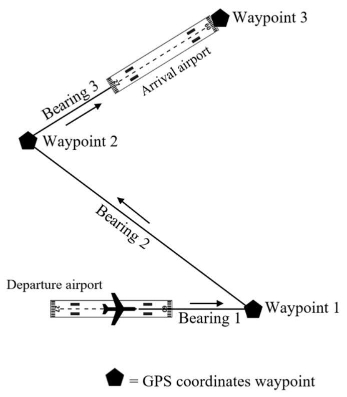

Fig. 5. A flight course from a departure airport to a landing airport consisting of three path lines and their bearings/headings, which connect the three pre-stored GPS coordinates waypoint.

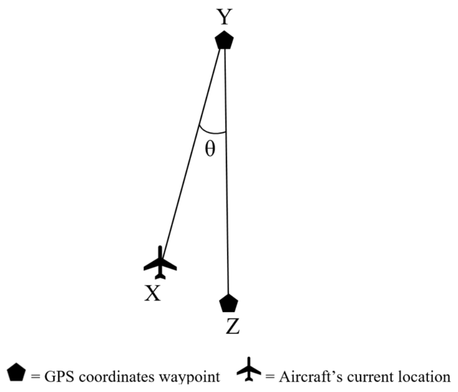

Fig. 6. The angle between the line from the aircraft's location X and the next waypoint Y, and the line from the previous waypoint Z and the next waypoint Y.

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.

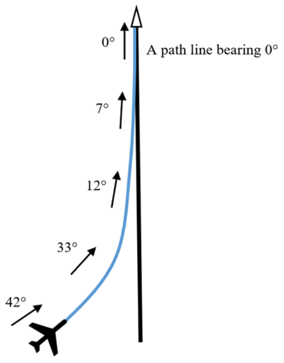

Fig. 7. An example illustrating how the Flight Manager updates the bearing to be followed based on the difference between the bearing of the path line to be intercepted, and the aircraft's bearing. The angle between the aircraft and the path line is added to the difference to ensure a gradual interception.

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,

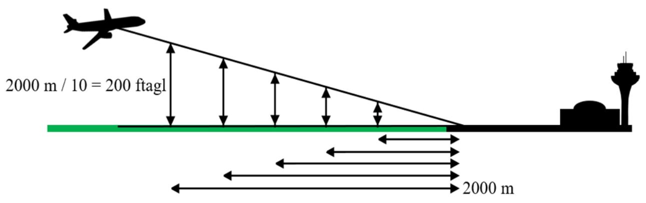

Fig. 8. The Glideslope generated by continuously calculating the altitude during the final approach descent which leads to a touchdown on the landing runway. The desired altitude is the distance to the runway divided by 10.

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

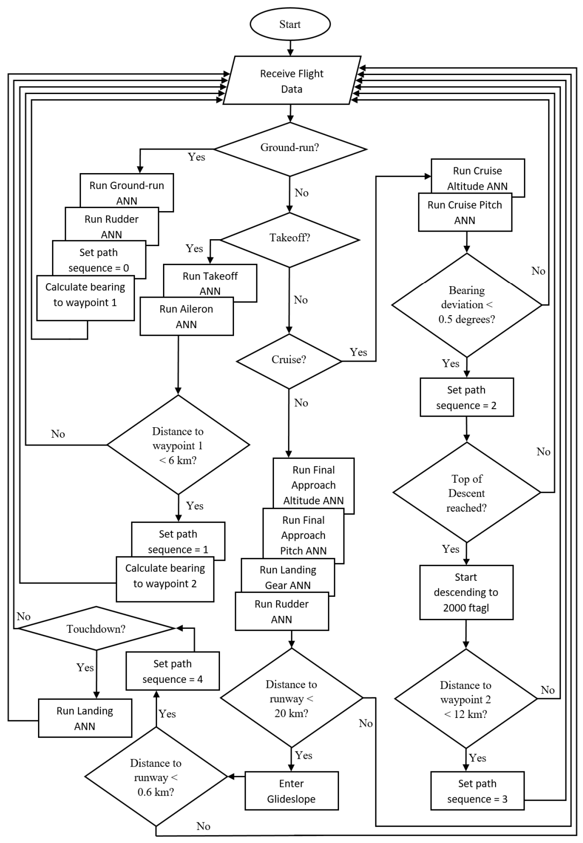

Fig. 9. A Flowchart illustrating the process which the Flight Manager program follows to decided which ANNs are to be used, and how to handle flight phases and navigation points transitions.

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.

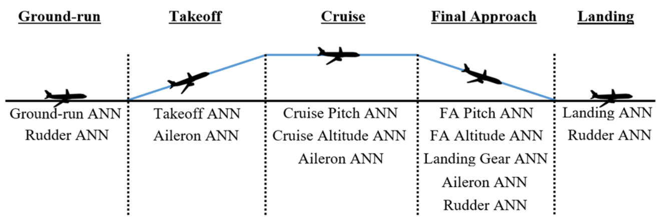

Fig. 10. The ANNs used during the different phases of the flight.

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.

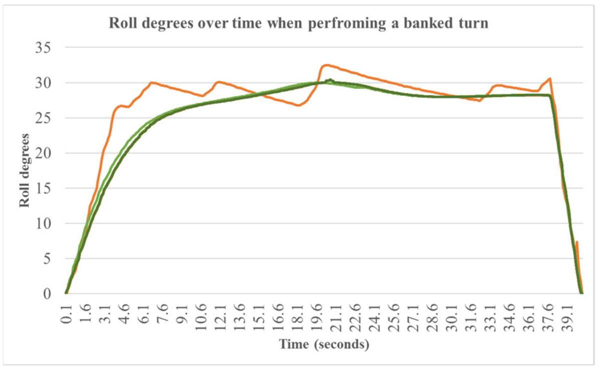

Fig. 11. (Banking turn and path line interception experiment). A comparison between the human pilot (orange line) and the 50 attempts by the IAS (overlapping lines) to perform a banked turn to change the aircraft's bearing by 145 degrees over a period of 40 seconds. The good fit of the generated model allowed the IAS to maintain a significantly steadier and consistent change of roll degrees compared to the human pilot.

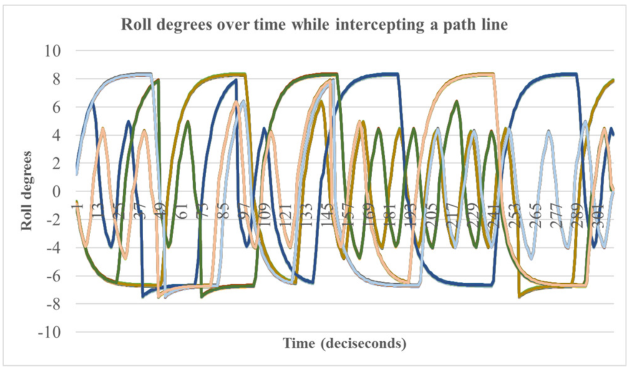

Fig. 12. (Banking turn and path line interception experiment). Smaller degrees of roll when banking continuously to intercept a path line. For the 50 attempts, the roll is below 9 degrees (suitable for small turns while intercepting) compared to a maximum of 31 (suitable for major bearing change) as Fig. 10 illustrates. Due to the significant consistency, and similarity of the testing scenarios (location and weather), most of the lines are overlapped.

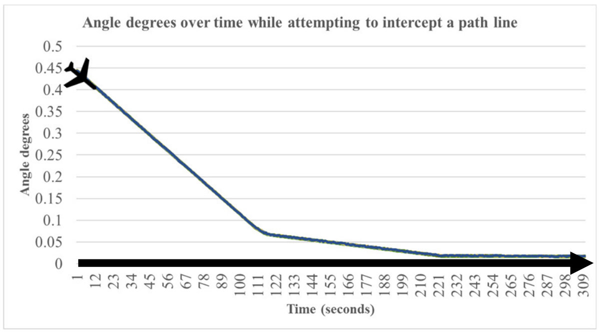

Fig. 13. (Banking turn and path line interception experiment). 50 attempts with strong consistency to gradually intercept and follow a path line (black arrow). The interception attempt is represented by the gradual decrease of the angle between the aircraft and the path line.

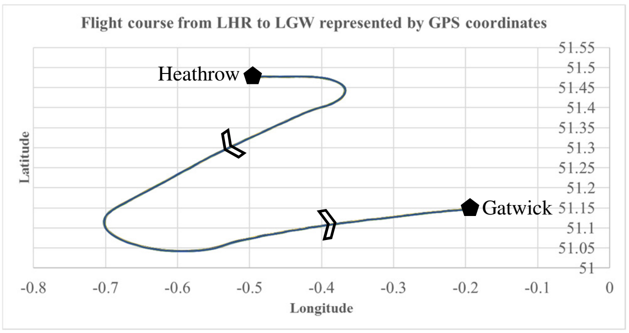

Fig. 14. (Banking turn and path line interception experiment). The 50 flight courses (overlapped lines) with strong consistency flown autonomously by the IAS, starting with takeoff from London Heathrow airport, and landing at Gatwick airport. Since the distance between the two airports is short for an 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 illustrate 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

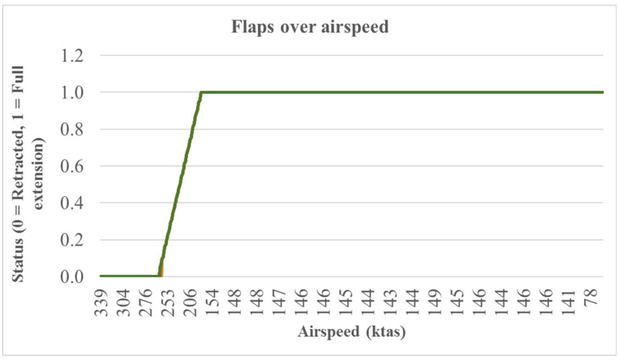

Fig. 15. (Final approach experiment). A comparison between the human pilot and 50 attempts by the IAS (overlapped lines) when extending flaps immediately after an airspeed of 260 (ktas) is reached. The behaviour of the human pilot and the IAS in the 50 attempts are significantly close with strong consistency.

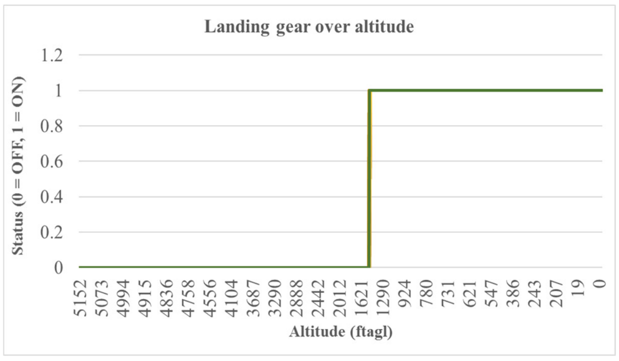

Fig. 16. (Final approach experiment). A comparison between the human pilot and 50 attempts by the IAS (overlapped lines) when engaging the landing gear immediately after an altitude of 1500 (ftagl) is reached. The behaviour of the human pilot and the IAS in the 50 attempts are significantly close with strong consistency.

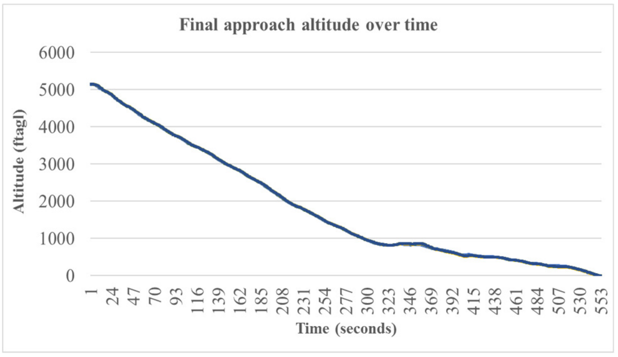

Fig. 17. (Final approach experiment). 50 attempts (overlapped lines) with strong consistency by the IAS to follow the final approach glideslope. The glideslope is adjusted by the IAS after descending to an altitude below 1000 (ftagl) to compensate for the additional drag generated by the landing gear.

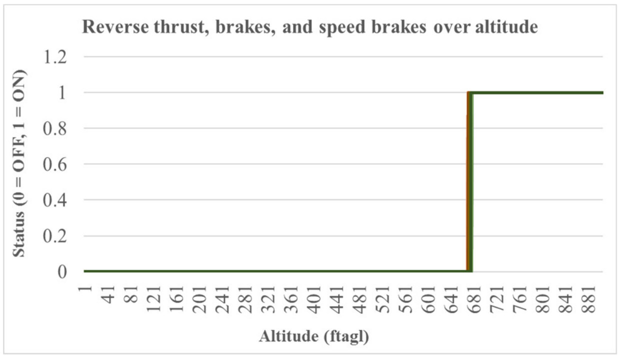

Fig. 18. (Landing experiment). A comparison between the human pilot and 50 attempts by the IAS (overlapped lines) when engaging reverse thrust, brakes, and speed brakes immediately after touchdown. The behaviour of the human pilot and the IAS in the 50 attempts are significantly close with strong consistency.

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 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. 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 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 Large Jets 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.