UNIVERSIDAD COMPLUTENSE DE MADRID
FACULTAD DE INFORMÁTICA

TESIS DOCTORAL

Aplicación de técnicas de soft computing al modelado y control de vehículos aéreos no tripulados

Soft computing techniques applied to modelling and control of unmanned aerial vehicles

MEMORIA PARA OPTAR AL GRADO DE DOCTOR

PRESENTADA POR

Jesús Enrique Sierra García

Directora

Matilde Santos Peñas

Madrid
Ed. electrónica 2019

UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE INFORMÁTICA

APLICACIÓN DE TÉCNICAS DE SOFT COMPUTING AL MODELADO Y CONTROL DE VEHÍCULOS AÉREOS NO TRIPULADOS

SOFT COMPUTING TECHNIQUES APPLIED TO MODELLING AND CONTROL OF UNMANNED AERIAL VEHICLES

TESIS DOCTORAL DE:

JESÚS ENRIQUE SIERRA GARCÍA

DIRIGIDA POR:

MATILDE SANTOS PEÑAS

Madrid, 2019

Ph. D. Dissertation

Aplicación de Técnicas de Soft Computing al Modelado y Control de Vehículos Aéreos no Tripulados

Soft Computing Techniques Applied to Modelling and Control of Unmanned Aerial Vehicles

Jesús Enrique Sierra García

Facultad de Informática

Universidad Complutense de Madrid

Madrid, 2019

Ph. D. Dissertation

Soft Computing Techniques Applied to Modelling and Control of Unmanned Aerial Vehicles

Title	Soft Computing Techniques Applied to Modelling and Control of Unmanned Aerial Vehicles
Author	Jesús Enrique Sierra García
Degree	Electronics Engineer & Telecommunication Engineer Universidad de Valladolid
Supervisors	Matilde Santos Peñas, UCM
University	School of Computer Science Universidad Complutense de Madrid
Program	Computer Science and Engineering

Madrid, 2019

Ph. D. Dissertation

Aplicación de Técnicas de Soft Computing al Modelado y Control de Vehículos Aéreos no Tripulados

Título	Aplicación de Técnicas de Soft Computing al Modelado y Control de Vehículos Aéreos no Tripulados
Autor	Jesús Enrique Sierra García
Grado	Ingeniero en Electrónica & Ingeniero de Telecomunicación Universidad de Valladolid
Supervisores	Matilde Santos Peñas, UCM
Universidad	Facultad de Informática Universidad Complutense de Madrid
Programa	Ingeniería Informática

Madrid, 2019

To my family and friends

Acknowledgements

I would like to express my deep gratitude to Professor Matilde Santos, my research supervisor, for her patient guidance, enthusiastic encouragement and useful critiques of this research work.

My deep and sincere gratitude to my family for their continuous and unparalleled love, help and support, this journey would not have been possible without them. Specially to my mother for instilling in me a work hard culture, and my wife for her wholehearted encouragement.

Finally, my gratitude to my bagpipe and flutes for helping me to find the solution when I was blocked.

Contents

List of Figures

List of Tables

List of Abbreviations

ADRC	Active Disturbance Rejection Control
AI	Artificial Intelligence
ANN	Artificial Neural Network
ANFIS	Adaptive Neuro-Fuzzy Inference System
ARMAX	Autoregressive Moving Average eXogenous
ARX	Autoregressive Exogenous
BJ	Box-Jetkins
CI	Computational Intelligence
CIFER	Comprehensive Identification from Frequency Responses
CMAC	Cerebellar Model Articulation Controller
DOBC	Disturbance Observer-Based Control
DOD	Department of Defence
ELM	Extreme Learning Machine
EP	Excitación Persistente
GLA	Generalized Learning Algorithm
IBSS	Iterative Bi-Section Shooting
IDS	Inverse Dynamics System
LQR	Linear-Quadratic Regulator
MIMO	Multi-Input Multi-Output
MISO	Multi-Input Single-Output
MLP	Multilayer Perceptron
MSE	Mean Squared Error
NARX	Non-Linear ARX
NASA	National Aeronautics and Space Administration
OE	Output Error
PDE	Partial Differential Equation
PEM	Prediction Error Methods
PID	Proportional-Integral-Derivative
RBF	Radial Basis Function
RBN	Radial Basis Networks
RC-Helicopter	Remote Controlled Helicopter
RELU	Rectified Linear Unit
ROA	Remotely Operated Aircraft
RPA	Remoted Piloted Aircraft
RPV	Remotely Piloted Vehicle
SISO	Single-Input Single-Output
SMC	Sliding Mode Control
SMDO	Sliding Mode Disturbance Observer
SNR	Signal-Noise Ratio
TC	Time Complexity
UAS	Unmanned Aircraft System
UAV	Unmanned Aerial Vehicle
USA	United States of America
UVS	Unmanned Vehicle System
VTOL	Vertical Take-off and Landing

Abstract

The use of UAVs (unmanned aerial vehicles), and specially quadrotors, is growing day by day, they are planned to be used in multitude of valuable applications: rescue, security, fire-fighting, agriculture, structure inspection, logistics, … In most of those tasks, the quadrotors are expected to be fully autonomous. All these applications and those to come, demand the design of efficient and robust models and controllers for those autonomous vehicles. However, this is not an easy task due to, among others: the randomness of the airstreams, the high non-linearity dynamics, the coupling between the internal variables, etc. These factors make the *Soft Computing* techniques (a field of the Artificial Intelligence), and among them specially the artificial neural networks and the fuzzy logic, a promising approach for the identification and control of these systems.

The learning ability of these intelligent strategies is especially useful when the model parameters change while the system is working and when the UAV is subject to noticeable disturbances. These issues are applicable in our case: in the quadrotor the total mass will undergo variations when it is performing logistic tasks and the effect of the wind will exert a high influence on its movement.

To tackle these problems in UAVs, this Thesis proposes the use of artificial neural networks, in same cases combined with fuzzy logic, to identify and control quadrotors. The developed models aim the reduction of the computational complexity of the UAVs models thanks to the hybridization of parametric and *Soft Computing* techniques; furthermore online learning has been applied to reject disturbances. Neuro-controllers have been combined to tackle wind and mass variation. The outcomes contribute with some improvements in the modelling and control of quadrotors field.

Keywords: Identification, Modelling, Parametric techniques, Neuro-fuzzy networks, Hybridization, Neuro-control, Adaptive control, Disturbances rejection, On-line learning, Neural networks, Unmanned aerial vehicles (UAV), Quadrotor

Resumen

El uso de UAVs (vehículos autónomos aéreos), y en concreto, de cuatrirrotores o drones, está creciendo de día en día, y se espera que se usen en multitud de aplicaciones: rescate, seguridad, lucha contra incendios, agricultura, inspección de estructuras, logística, … En la mayoría de estas tareas los cuatrirrotores deben actuar de una forma totalmente autónoma. Estas aplicaciones y las que están por llegar requieren el diseño de modelos y controladores eficientes y robustos para esos vehículos no pilotados. Sin embargo, esta no es una tarea sencilla debido, entre otras causas, a la aleatoriedad de los flujos de aire, la dinámica altamente no lineal del UAV, el acoplamiento entre sus variables internas, etc. Estos factores hacen que las técnicas de Soft Somputing (computación suave, una rama de la Inteligencia Artificial), y entre ellas concretamente las redes neuronales artificiales y la lógica fuzzy, sean un enfoque prometedor para la identificación y el control de estos sistemas.

La capacidad de aprendizaje de estas estrategias inteligentes es especialmente útil cuando los parámetros del modelo cambian mientras el sistema está en funcionamiento y cuando el UAV está sujeto a perturbaciones relevantes. Estos problemas son aplicables en nuestro caso: en el cuatrimotor la masa total sufrirá variaciones al realizar tareas logísticas y el efecto del viento ejercerá una gran influencia en su movimiento.

Para abordar esta problemática de los UAV, esta Tesis propone el uso de redes neuronales artificiales, en algún caso combinadas con lógica fuzzy, para la identificación y control de cuatrirrotores. Los modelos desarrollados persiguen la reducción de la complejidad computacional de los modelos de los UAVs gracias a la hibridación de técnicas paramétricas y del Soft Computing; además se ha aplicado aprendizaje on-line para el rechazo de perturbaciones. Se han combinado neuro-controladores para hacer frente a variaciones de masa y viento. Los resultados permiten contribuir con ciertas mejoras al campo del modelado y el control de cuatrirrotores.

Palabras Clave: Identificación, Modelado, Técnicas paramétricas, Redes neuro-difusas, Hibridación, Control neuronal, Control adaptativo, Rechazo de perturbaciones, Aprendizaje on-line, Redes neuronales, Vehículos aéreos no tripulados (UAV), Cuatrirrotores.

I Resumen de la Investigación

1 Introducción

Este capítulo recoge algunas ideas generales sobre UAVs (Unmanned Aerial Vehicles) (Sección 1.1) y las técnicas de soft computing (Sección 1.2), la motivación de la investigación (Sección 1.3), sus objetivos y ámbito (Sección 1.4), y las principales contribuciones de la Tesis (Sección 1.5).

1.1 VEHÍCULOS AÉREOS NO TRIPULADOS (UAVS)

El concepto de UAV se aplica a cualquier vehículo que este volando en el aire sin conductor a bordo y con cierta autonomía de movimientos [1], aunque puede aparecer con diferentes nombres como Vehículo Pilotado Remotamente (RPV), Avión Pilotado Remotamente (RPA), Avión Operado Remotamente (ROA), Helicóptero Controlado Remotamente (RC-Helicopter), Sistema de Aviación no Tripulado (UAS) y Sistema Vehicular no Tripulado (UVS). Los UAVs son definidos por el Departamento de Defensa de los EE. UU. como vehículos aéreos alimentados que no llevan a un operador humano, que usan fuerzas aerodinámicas para proporcionar la sustentación, pueden volar autónomamente o ser pilotados remotamente, pueden ser desechables o recuperables, y pueden llevar cargamento letal o no letal [2].

Los UAVs tienen múltiples aplicaciones civiles y militares. Por ejemplo, se emplean en agricultura para la monitorización de los cultivos, en medio ambiente para la prevención de incendios, en seguridad y protección civil para la vigilancia, en arquitectura y construcción para la fotografía y el análisis de estructuras, en operaciones de inspección de tendidos eléctricos y aerogeneradores, etc. Una de las más recientes aplicaciones es la vigilancia del tráfico en carreteras, autovías y autopistas.

Estas aplicaciones requieren el diseño de controladores eficientes y robustos. Sin embargo, el modelado y control de los UAVs no es algo sencillo. Su complejidad proviene de la aleatoriedad de los flujos de aire y de las fuerzas exógenas, la alta no-linealidad de su dinámica, el acoplamiento entre las variables, la incertidumbre en las medidas, etc. Estos factores hacen que las técnicas basadas en inteligencia artificial sean consideradas como un enfoque prometedor para la identificación y control de estos sistemas. De ahí que el diseño y el control de estos sistemas complejos no lineales siga despertando el interés de la comunidad científica [3-10].

Existe un amplio rango de tipos de UAVs, como se muestra a continuación [11].

Multirrotores

Los multirrotores son UAVs con más de un rotor en la parte más alta. Normalmente es el medio más barato de conseguir videos aéreos rápidamente y con calidad. La principal desventaja de los multirrotores es su limitada autonomía y velocidad, haciéndoles inadecuados para mapeado aéreo a gran escala, monitorización de gran autonomía, e inspecciones de largas distancias como oleoductos, carreteras y líneas de alta tensión. El modelo más usado es el cuatrirrotor, pero hay otros modelos con 6, 8, 10, 12 e incluso con más rotores. La Figura 1.1 muestra un ejemplo de multirrotor de 8 hélices [12].

Fig. 1.1.Un UAV multirrotor [12]

Alas-fijas

Los UAVs de alas fijas emplean alas estándar como las de los aviones para proporcionar sustentación. Por esta razón pueden cubrir mayores distancias, mapear zonas mucho más grandes, y volar durante más tiempo monitorizando puntos de interés. Además de su mayor eficiencia, también es posible utilizar motores de gasolina como fuente de energía, y de este modo muchos UAVs de alas fijas pueden estar en vuelo durante 16 horas o más. La principal desventaja de un avión de alas fijas es obviamente su incapacidad para girar sobre un punto, lo cual les descarta para cualquier trabajo de fotografía aérea de forma general. Esto también hace que el despegue y el aterrizaje sean muy complejos; dependiendo del tamaño pueden necesitar una pista o una catapulta para ponerles en el aire, o una pista, paracaídas o red para recuperarlos de forma segura. Un ejemplo de este tipo se muestra en la Figura 1.2 [13].

Fig. 1.2. Un UAV de alas fijas [13]

Helicópteros de una hélice:

Un UAV helicóptero de una hélice tiene solamente un rotor en la parte más alta y un motor en la cola para controlar la dirección. Se muestra un ejemplo en la Figura 1.3 [14].

Fig. 1.3. Helicóptero de rotor único [14]

Este sistema suele ser más eficiente que un multirrotor; además normalmente vienen provistos de un motor de gasolina para una mayor autonomía. Suelen ser muy útiles para transportar cargas pesadas como un LIDAR. Los principales defectos son: complejidad, coste, vibración y mayor peligrosidad debido a las hélices.

Modelos híbridos VTOL (Vertical Take-off and Landing) de alas fijas

Hoy en día se siguen diseñando nuevos modelos de estos tipos de UAVs. Por ejemplo, la Figura 1.4 muestra un VTOL diseñado recientemente por la NASA [15]. La principal característica es el aterrizaje y despegue vertical (VTOL). Este hecho permite reducir los requerimientos de espacio de aterrizaje/despegue al mínimo.

Fig. 1.4. Un UAV de alas fijas híbrido VTOL [15]

El UAV más conocido es el cuatrirrotor; por este motivo esta Tesis se centra en este modelo. Además el sistema es suficientemente complejo para ensayar las nuevas técnicas de modelado y control propuestas.

1.2 SOFT COMPUTING

El término Soft Computing fue propuesto por el inventor de la lógica difusa, Lotfi A. Zadeh: "Soft computing es un conjunto de metodologías que tienen como objetivo explotar la imprecisión e incertidumbre para alcanzar trazabilidad, robustez y soluciones de bajo coste. Sus principales constituyentes son la lógica difusa, el cálculo neuronal, y el razonamiento probabilístico…. El modelo de referencia para el soft computing es la mente humana" [16].

El soft computing no está definido de forma precisa. Está formado por diferentes conceptos y técnicas que permiten abordar y resolver algunas dificultades encontradas en problemas reales provenientes del hecho de que nuestro mundo parece ser impreciso, incierto y difícil de categorizar. Por ejemplo, la incertidumbre que se introduce al medir una cantidad debido a variaciones inherentes en el proceso de medida en sí mismo [17].

El soft computing también puede ser visto como la base del área de conocimiento de la Inteligencia Computacional (CI). La diferencia entre la inteligencia artificial tradicional (AI) y la inteligencia computacional es que la AI se basa en hard computing mientras que CI se basa en soft computing [17].

Durante el desarrollo de esta Tesis se han empleado las redes neuronales en el proceso de modelado y en el diseño de las estrategias de control, explotando principalmente la capacidad de aprendizaje on-line de estas técnicas. También se ha utilizado la lógica difusa en el modelado mediante la técnica ANFIS (para más detalles ver sección 2.2.3)

1.3 MOTIVACIÓN Y FORMULACIÓN GENERAL DEL PROBLEMA

En los últimos años están surgiendo nuevas y valiosas aplicaciones de los vehículos aéreos no tripulados en diferentes sectores como: defensa, seguridad, construcción, agricultura, entretenimiento, paquetería, etc. [18-21]. Estas y otras aplicaciones requieren el diseño de controladores eficientes y robustos.

Para analizar y controlar efectivamente un sistema físico es necesario encontrar un modelo matemático útil y fiable que describa su comportamiento. La importancia de tener un buen modelo crece con la complejidad del sistema. Existen muchas técnicas y métodos para alcanzar este objetivo, pero se pueden clasificar en dos grupos principalmente. El primero se basa en aplicar las ecuaciones físicas bien conocidas que gobiernan el comportamiento dinámico y usar este conocimiento para generar las ecuaciones diferenciales que lo representen. Esta metodología es directa cuando todas las variables están definidas, y las relaciones entre ellas son simples. Pero en muchas ocasiones es necesario tratar con ecuaciones demasiado complejas o incluso imposibles de obtener. En este caso, es necesario aplicar métodos algo más avanzados tales como la identificación de sistemas. Este otro grupo de técnicas está basado en el diseño de experimentos que generen datos informativos de manera eficiente, para después aplicar diferentes mecanismos de mapeado entre entradas y salidas. Estos procesos de ajuste actualizan iterativamente sus parámetros para sintonizar el modelo, así como para reducir su orden [22]. En esta última categoría se pueden encontrar, entre otros, los métodos paramétricos y las técnicas inteligentes proporcionadas por la disciplina del soft computing, como son las redes neuronales y neuro-difusas y los algoritmos genéticos [23]. Estos métodos son adecuados cuando se trata de sistemas complejos y fuertemente no lineales como a los que nos enfrentamos en este trabajo.

Debido a la capacidad de aprendizaje on-line, estas estrategias inteligentes resultan especialmente interesantes cuando los parámetros del modelo varían mientras el sistema está en funcionamiento. Por ejemplo, en el caso del UAV la masa total sufre variaciones cuando el

vehículo está realizando tareas logísticas, y también se ve muy influenciado por perturbaciones externas cambiantes como el viento.

Por todo esto, esta Tesis propone el uso de redes neuronales artificiales y lógica difusa para mejorar las técnicas actuales de modelado y control de UAVs cuatrirrotores. En esta línea se han realizado otros estudios anteriormente, pero aún quedan puntos que deben ser estudiados en profundidad, como por ejemplo: la reducción de la complejidad computacional de los modelos de UAVs gracias a la hibridación de las técnicas de modelado; o los beneficios del aprendizaje on-line en el rechazo de perturbaciones. Estos y otros temas han sido tenidos en cuenta en este trabajo, el cual puede contribuir a mejoras significativas en el modelado y el control de cuatrirrotores.

1.4 OBJETIVOS DE LA TESIS

El objetivo general de esta Tesis es desarrollar métodos de modelado avanzados y aplicar estrategias de control innovadoras para llevar a cabo un control inteligente de UAVs capaz de adaptarse a cambios en el sistema y en el entorno, y rechazar perturbaciones. Dentro del amplio rango de UAVs, esta Tesis se centra en el cuatrirrotor. Sin embargo, muchos de los resultados podrían ser generalizados a otros tipos de UAVs.

En este trabajo se han abordado los siguientes objetivos específicos:

- El estudio del estado del arte sobre el modelado y el control de cuatrirrotores, prestando especial atención a los trabajos previos basados en la aplicación de técnicas inteligentes.

Modelado:

- El análisis de la viabilidad de la hibridación de técnicas para mejorar las características de los modelos.
- La comparación del rendimiento de diferentes modelos matemáticos, considerando el error en los dominios del tiempo y de la frecuencia, así como su complejidad computacional.
- El estudio de la influencia de la partición de los datos en el rendimiento de los modelos obtenidos mediante redes neuronales.

Control:

- La propuesta y validación mediante simulación de una estrategia de control de UAVs basada en redes neuronales artificiales (neuro-control).
- Añadir capacidades de aprendizaje a la estrategia de control diseñada (neuro-control adaptativo), permitiéndola reaccionar frente a cambios en el entorno y en los parámetros del modelo.
- Estudiar la posibilidad de integrar las redes en una estrategia de control que permita rechazar algunas perturbaciones, como por ejemplo, las producidas por el viento.

1.5 PRINCIPALES CONTRIBUCIONES

Las contribuciones de esta Tesis se resumen en las conclusiones principales de este trabajo y pueden ser ordenadas en los siguientes desarrollos y publicaciones.

1.5.1 Aportaciones

Los resultados de esta Tesis incluyen el desarrollo de algoritmos de control y modelado, que han sido implementados en MATLAB [24]. Las aportaciones más importantes se resumen a continuación:

- Modelos de cuatrirrotor basados en redes neuronales.
- Hibridación de técnicas paramétricas y del soft computing para la identificación de sistemas.
- Estrategia de neuro-control adaptativo basado en modelo de planta inverso.
- Estimadores neuro-adaptativos para variaciones de la masa y la perturbación del viento.
- Controlador de seguimiento adaptativo basado en los estimadores neuro-adaptativos del punto anterior.

1.5.2 Publicaciones

Durante el desarrollo de la tesis doctoral se han publicado varios artículos en revistas especializadas, y contribuciones en congresos nacionales e internacionales. La mayoría de los

artículos se han obtenido como resultado directo de esta Tesis, siendo el doctorando el primer autor de todos ellos.

Artículos Publicados en Revistas Indexadas:

- Sierra. J.E., Santos, M. (2018). Modelling engineering systems using analytical and neural techniques: hybridization. Neurocomputing, 271, 70-83 **(Q1)**
- Sierra. J.E., Santos, M. (2019). Wind and payload disturbance rejection control based on adaptive neural estimators: application on quadrotors. Complexity, Article ID 6460156, 20 pages **(Q2)**

Artículos en Congresos:

- Sierra, J.E., Santos, M. (2013) Estudio comparativo de modelos de un vehículo aéreo obtenidos mediante técnicas analíticas y basadas en redes neuronales. Actas Multiconferencia CAEPIA'13 (XV Conferencia de la Asociación Española para la Inteligencia Artificial), 1270-1279. ISBN: 978-84-695-8348-7
- Sierra, J.E., Santos, M. (2013) Modelado de un vehículo aéreo no tripulado mediante aplicación conjunta de técnicas paramétricas y neuronales. Actas de las XXXIV Jornadas de Automática, 189-194, 2013. ISBN: 978-84-616-5063-7
- Sierra, J.E., Santos, M. (2014) Modelado de un vehículo aéreo no tripulado mediante combinación de técnicas paramétricas y neurodifusas. In Actas del XVII Español sobre Tecnologías y Lógica Fuzzy, 339-344. ISBN: 978-84-15688-76-1.
- Sierra, J. E., Santos, M. (2015). Adaptive neural control-oriented models of unmanned aerial vehicles. Advances in Intelligent Systems and Computing, (pp. 329-337). Springer International Publishing. (10th International Conference on Soft Computing Models in Industrial and Environmental Applications). ISBN 2194-5357.
- Sierra, J. E., Santos, M. (2017). Control de un vehículo cuatrirrotor basado en redes neuronales. Actas de las XXXVIII Jornadas de Automática, pp 431-436. ISBN: 978-84-16664-74-0.
- Sierra, J. E., Santos, M. (2019). Disturbances based adaptive neuro-control for UAVs: a first approach. Advances in Intelligent Systems and Computing (pp. 293-302). Springer, Cham (In The 13th International Conference on Soft Computing Models in Industrial and Environmental Applications). ISBN: 978-3-319-94119-6

2 Técnicas Utilizadas y Materiales

Este capítulo describe los materiales empleados durante la fase de modelado (Sección 2.1) y las diferentes técnicas analizadas para su aplicación al modelado y el control de los UAVs (Sección 2.2).

2.1 MATERIALES

Para el estudio del modelado se emplearon datos proporcionados por el Grupo de Ingeniería de Control del Comité Español de Automática (CEA) [25]. Los datos fueron capturados a partir de un vehículo aéreo no tripulado ARDRONE 2.0 de Parrot (Figura 2.1). Este sistema está hecho de fibra de carbono y plástico PA66. Está equipado con cuatro motores sin escobillas de alta eficiencia, alimentados por una batería de litio de 11 V 1000mAh, que le proporciona una autonomía de vuelo de 12 minutos. Pesa 380g sin carcasa y 420 g con carcasa. Tiene un acelerómetro de tres ejes y 2 giróscopos, uno para el ángulo de cabeceo y otro para el ángulo de alabeo, y un giróscopo de alta precisión para el ángulo de guiñada (yaw) [26].

Fig. 2.1. ARDRONE de Parrot [27]

Las señales de entrada/salida se representan en la Figura 2.2. La señal de cabeceo (pitch) y la velocidad en el eje X se muestran en azul, mientras que el alabeo (roll) y la velocidad en el eje Y se representan en verde. Como puede verse en la figura, las señales de entrada consisten en un tren de impulsos de diferentes amplitudes. El pitch y el roll aparecen en la figura superior y las velocidades en la figura inferior. Este tipo de señales se ha empleado tradicionalmente para extraer la máxima información posible de los sistemas.

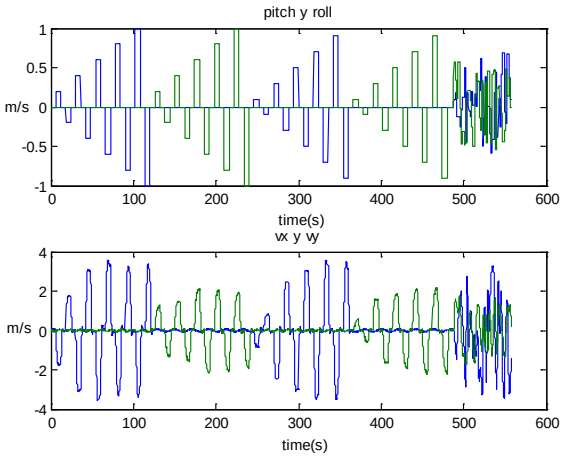

Fig. 2.2. Señales de entrenamiento de entrada/salida

Para extraer la máxima información posible de la planta, las señales de entrada empleadas durante la identificación deben poseer tanta potencia como sea posible, es decir un EP de grado elevado, para excitar el mayor número posible de frecuencias del sistema; su duración debe ser lo mayor posible, cuanto mayor es el número de datos de entrada menor será la varianza de los parámetros estimados; y no deben estar correlacionadas con la perturbación. Por otro lado, también se persigue que los experimentos sean amigables con la planta, para ello deben tener una duración tan corta como sea posible y producir la mínima perturbación de las variables controladas, es decir, introducir en las mismas una varianza baja y desviaciones pequeñas del punto de consigna.

En consecuencia, a la hora de escoger una señal de entrada debe llegarse a un compromiso entre facilidad de uso y capacidad de extracción de información. Las señales de la Figura 2.2 presentan un balance adecuado en estos indicadores para identificar el modelo del sistema.

2.2 TÉCNICAS UTILIZADAS

2.2.1 Modelos PEM (Prediction Error Methods)

Este tipo de técnica es muy útil para generar modelos cuanto el comportamiento del sistema se acerca al lineal. Aunque este no es nuestro caso, estas técnicas pueden usarse si se comprueba que el error es tolerable para cada aplicación específica.

Consideremos un sistema lineal invariante descrito por la siguiente ecuación diferencial:

$$y(t) = G(\theta, q)u(t) + H(\theta, q)e(t) \tag{2.1}$$

Donde $y(t)$ es la salida, $u(t)$ es la entrada, $G(\theta, q)$ es la función de transferencia de la planta y $H(\theta, q)$ es la función de trasferencia de la perturbación, q es el operador de retardo y $e(t)$ es el error de predicción de un paso. $G(\theta, q)$ y $H(\theta, q)$ vienen dadas por las expresiones [28]:

$$G(\theta, q) = \sum_{k=1}^{\infty} g(\theta, k)q^{-k} \qquad H(\theta, q) = 1 - \sum_{k=1}^{\infty} h(\theta, k)q^{-k} \tag{2.2}$$

Para obtener el predictor de un paso, asumamos que $e(t)$ es ruido blanco, por lo que su varianza está dada por $Var(e) = \lambda^2(\theta)I$. Entonces la salida $\hat{y}(\cdot, \theta)$ que minimiza $E(y - \hat{y})^T(y - \hat{y})$ tiene la forma:

$$\hat{y}(\cdot, \theta) = L_1(q, \theta)y + L_2(q, \theta)u \tag{2.3}$$

Las entradas y salidas pasadas son mapeadas para predecir la nueva salida.

$$y - \hat{y} = Gu + He - L_2u - L_1y \tag{2.4}$$

Teniendo en cuenta que $y = Gu + He$, manipulando los términos de la ecuación llegamos a:

$$y = (I - H^{-1})y + H^{-1}Gu + e \tag{2.5}$$

Por lo tanto,

$$y - \hat{y} = (I - H^{-1} - L_1)y + (H^{-1}G - L_2)u + e = z + e \tag{2.6}$$

Donde $E(y - \hat{y})^T(y - \hat{y})$ debe ser minimizado,

$$E(y - \hat{y})^T(y - \hat{y}) = E(z^Tz) + E(e^Te) \geq \lambda^2(\theta)I \tag{2.7}$$

El límite más bajo se alcanza cuando $z = 0$, entonces de (2.3) L_1 y L_2 debe ser:

$$L_1 = I - H^{-1} \qquad L_2 = H^{-1}G \tag{2.8}$$

Finalmente, el predictor de un paso resulta,

$$\hat{y}(t, \theta) = \left(1 - H^{-1}(q, \theta)\right)y(t) + H^{-1}(q, \theta)G(q, \theta)u(t) \tag{2.9}$$

Y el error de predicción de un paso es,

$$\varepsilon(t, \theta) = y - \hat{y} = H^{-1}(q, \theta)[y - G(q, \theta)]u \tag{2.10}$$

La base de las técnicas paramétricas es el paradigma del error de predicción mínimo. Dado $[y(t), u(t), t \leq N]$, encontrar $\widehat{\theta}$ (es decir la estimación de los parámetros θ) que minimiza

$$Var(\varepsilon) = \frac{1}{N}\sum_{t=1}^{N}\|\varepsilon(t,\theta)\|^2 \tag{2.11}$$

El modelo definido en (2.1) está basado en PEM (Prediction Error Methods), $G(q)$ y $H(q)$ son funciones racionales que vienen dadas por los coeficientes del numerador y el denominador, que son los parámetros del modelo.

En general, un modelo PEM puede ser expresado mediante la siguiente ecuación:

$$y(t) = \frac{B(q)}{A(q)F(q)}u(t-nk) + \frac{C(q)}{A(q)D(q)}e(t) \tag{2.12}$$

Donde $A(q), B(q), C(q), D(q)$ y $F(q)$ son polinomios:

$$\begin{aligned} A(q) &= 1 + a_1 q^{-1} + \cdots + a_{na} q^{-na} \\ B(q) &= b_1 q^{-1} + \cdots + b_{nb} q^{-nb} \\ C(q) &= 1 + c_1 q^{-1} + \cdots + c_{nc} q^{-nc} \\ D(q) &= 1 + d_1 q^{-1} + \cdots + d_{nd} q^{-nd} \\ F(q) &= 1 + f_1 q^{-1} + \cdots + f_{nf} q^{-nf} \end{aligned} \tag{2.13}$$

La estructura de un modelo PEM se define por el orden de los polinomios (na, nb, nc, nd, nf) y el retardo nk. Como hay cinco funciones polinomiales diferentes, es posible encontrar hasta 32 estructuras PEM diferentes. Para llevar a cabo este estudio, se han seleccionado tres de ellas de acuerdo con su importancia y simplicidad: ARX, ARMAX and OE.

ARX (Autoregressive with Exogenous Input)

Un modelo ARX de define por la ecuación diferencial [28]:

$$A(q,\theta)y(t) = B(q,\theta)u(t-nk) + e(t) \tag{2.14}$$

También puede ser expresado como,

$$\begin{aligned} &y(t) + a_1 y(t-1) + \cdots + a_{n_a} y(t-n_a) \\ &= b_1 u(t-1-nk) + \cdots + b_{n_b} u(t-n_b-nk) + e(t) \end{aligned} \tag{2.15}$$

Teniendo en cuenta las últimas ecuaciones, es sencillo darse cuenta que el vector de parámetros en este caso es $\theta = [a_1 \ \ldots \ a_{n_a} \ b_1 \ \ldots \ b_{n_b}]$

El acrónimo ARX proviene del inglés Autoregressive with Exogenous Input, ya que $A(q,\theta)y(t)$ es una autoregresión y $B(q,\theta)u(t-nk)$ es la contribución de la entrada exógena

u. En este caso, el problema de encontrar θ puede resolverse mediante un problema de regresión lineal. La correspondencia entre los polinomios y el modelo definido en (2.1) es:

$$G(q,\theta) = \frac{B(q,\theta)}{A(q,\theta)} \qquad H(q,\theta) = \frac{1}{A(q,\theta)} \tag{2.16}$$

ARMAX (Autoregressive Moving Average with Exogenous Input)

Un modelo ARMAX se define por la siguiente ecuación diferencial [28]:

$$A(q,\theta)y(t) = B(q,\theta)u(t-nk) + C(q,\theta)e(t) \tag{2.17}$$

También puede ser expresado como,

$$\begin{aligned} &y(t) + a_1 y(t-1) + \cdots + a_{n_a} y(t-n_a) \\ &\quad = b_1 u(t-1-nk) + \cdots + b_{n_b} u(t-n_b-nk) + \\ &\qquad c_1 e(t-1) + \cdots + c_{n_c} e(t-n_c) + e(t) \end{aligned} \tag{2.18}$$

Esta expresión es equivalente a:

$$\begin{aligned} &y(t) + a_1 y(t-1) + \cdots + a_{n_a} y(t-n_a) \\ &= b_1 u(t-1-nk) + \cdots + b_{n_b} u(t-n_b-nk) + v(t) \end{aligned} \tag{2.19}$$

Donde $v(t)$ es una media móvil de la señal de error $e(t)$ y está dado por:

$$v(t) = c_1 e(t-1) + \cdots + c_{n_c} e(t-n_c) + e(t) \tag{2.20}$$

Teniendo en cuenta las últimas ecuaciones, el vector de parámetros en este caso es $\theta = [a_1 \ \ldots \ a_{n_a} \ b_1 \ldots \ b_{n_b} \ c_1 \ldots \ c_{n_c}]$.

El acrónimo ARMAX proviene del inglés Autoregressive Moving Average with Exogenous Input, ya que $A(q,\theta)y(t)$ es una autoregresión y $B(q,\theta)u(t-nk)$ es la contribución de la entrada exógena u, y $v(t)$ es la media móvil de la señal de error. La correspondencia entre los polinomios y el modelo definido en (2.1) es:

$$G(q,\theta) = \frac{B(q,\theta)}{A(q,\theta)} \qquad H(q,\theta) = \frac{C(q,\theta)}{A(q,\theta)} \tag{2.21}$$

OE (Output Error)

El modelo está definido por la siguiente ecuación diferencial [28]:

$$y(t) = \frac{B(q,\theta)}{F(q,\theta)}u(t-nk) + e(t) \tag{2.22}$$

Donde

$$B(q) = b_1 q^{-1} + \cdots + b_{nb} q^{-nb}$$
$$F(q) = 1 + f_1 q^{-1} + \cdots + f_{nf} q^{-nf} \tag{2.23}$$

El significado del acrónimo OE es error de salida. Este nombre se explica por el hecho de que la fuente del error $e(t)$ es exactamente la fuente de perturbación, ya que $H(q,\theta) = 1$. En contraposición al último caso, el problema de encontrar θ debe resolverse como un problema de regresión no lineal. La correspondencia entre los polinomios y el modelo definido en (2.1) es:

$$G(q,\theta) = \frac{B(q,\theta)}{F(q,\theta)} \qquad H(q,\theta) = 1 \tag{2.24}$$

2.2.2 Redes Neuronales Aplicadas a la Identificación de Sistemas

Las redes neuronales artificiales son sistemas de procesado de información distribuida masivamente paralelizados, implementadas de forma hardware o software. Las redes neuronales básicas están compuestas por tres elementos: un conjunto de unidades de procesado simples idénticas o muy parecidas, altamente interconectadas, comúnmente llamadas neuronas, las cuales ejecutan un procesamiento local y están organizadas en una topología ordenada; un algoritmo de aprendizaje para adquirir conocimiento de su entorno; y finalmente, un algoritmo para usar el conocimiento adquirido [29]. La topología de la red y el tipo de neurona determinan el nombre de la red neuronal. En general, las redes pueden ser clasificadas en dos grandes grupos: redes neuronales con aprendizaje supervisado y con aprendizaje no supervisado. Para propósitos de identificación, las más interesantes son las de aprendizaje supervisado. Centrándonos en esta categoría, las topologías más sobresalientes son, entre otras: MLP (Multilayer Perceptron), RBF (Radial Basis Function) y CMAC (Cerebellar Model Articulation Controller) [30].

Algunas de las principales características de las redes neuronales son: mapeado de entradas-salidas no lineal complejo, adaptabilidad, capacidad de aprendizaje, arquitectura distribuida, tolerancia a fallos y analogía neurobiológica. Las dos primeras propiedades se explotan en el proceso de identificación. Como los sistemas reales son causales, el valor de la salida en el instante actual depende de los valores anteriores de las entradas y las salidas. Este hecho puede ser formalizado por la siguiente expresión:

$$y(k) = f(u(k), u(k-1), \dots, u(k-N), y(k-1), \dots, y(k-M)\ ) \qquad (2.25)$$

Donde k representa el instante de tiempo actual, y es la salida del sistema, u es la entrada y f es una función de mapeado normalmente no lineal. La estructura de la función f viene dada por la topología de la red y el tipo de neuronas. El comportamiento de esta función f es modulado por un conjunto de pesos, usualmente descritos por w. El proceso de aprendizaje de la rede modifica w, de tal forma que la salida del modelo es cada vez más parecida a la salida real debido a la aplicación de alguna regla de optimización. De esta forma, cuando el aprendizaje finaliza, la función f es una buena aproximación a la dinámica del sistema.

El algoritmo de aprendizaje supervisado es alimentado con un conjunto de datos de entrenamiento, donde se establece la relación entre las salidas y las entradas. Para cada dato objetivo (el valor de la señal de la salida), un vector de entrada (vector de regresión) es asignado. La correcta elección del algoritmo de aprendizaje es muy importante para alcanzar una velocidad apropiada e incluso para asegurar la convergencia. Se pueden encontrar varias clases de algoritmos: métodos de gradiente, búsqueda exhaustiva, búsqueda aleatoria, búsqueda genética, etc. Centrándonos en los métodos de gradiente, los más conocidos son: el gradiente escalonado, Newton, Levenberg-Marquardt [31] y el gradiente conjugado.

2.2.2.1 Perceptrón Multi-Capa (MLP)

Consiste en varias capas de neuronas conectadas en una única dirección hacia adelante, donde cada neurona de una capa está conectada con todas las neuronas de la capa siguiente. De esta forma, el grafo que representa la estructura de la red no posee ciclos. En cada capa puede emplearse una función de activación diferente. La conexión entre la neurona i y la neurona j (de dos capas consecutivas) tiene asociada un coeficiente de ponderación o peso, w_{ij}, y la neurona i posee un coeficiente umbral o bias b_i. La salida de la neurona i de un perceptrón viene dada por la expresión:

$$\mathrm{x_i} = \mathrm{f}(b_i + \sum_{j=1}^{N} w_{ij}\,\mathrm{x}_j) \qquad (2.26)$$

Donde f es la función de activación y N es el número de neuronas de la capa previa. Normalmente en la última capa se emplea una función de transferencia lineal. En las capas intermedias, también llamadas capas ocultas, se emplea a menudo la función sigmoide, la

tangente hiperbólica, la función RELU (Rectified Linear Unit), o el escalón, dependiendo del problema a resolver, aunque hay otras muchas funciones de activación posibles.

2.2.2.2 Redes de Función de Base Radial (RBF)

Se han empleado las redes de función de base radial (Radial Basis Function Networks o RBF) debido a su conocida aptitud para la aproximación de funciones; este hecho fue probado por Park en 1991 [32]. La topología de la red se muestra en la Figura 2.3.

Este tipo de red tiene tres capas: la capa de entrada que recibe las señales del exterior, la capa oculta que está conectada con la capa de entrada y donde cada elemento se asocia con una función de base radial, y la capa de salida donde los resultados de las funciones de activación de la capa oculta se ponderan linealmente [33].

La salida de cada elemento de la capa oculta se calcula como la distancia entre el patrón de entrada x(n) y el centro del cluster c_i ponderado por d_i. Este valor es aplicado a la función de base radial de acuerdo con la siguiente expresión:

$$z_i(n) = \Phi\left(\frac{\sqrt{\sum_{j=1}^{N}\left(x_j(n) - c_{ij}\right)^2}}{d_i}\right), i = 1, \ldots M \tag{2.27}$$

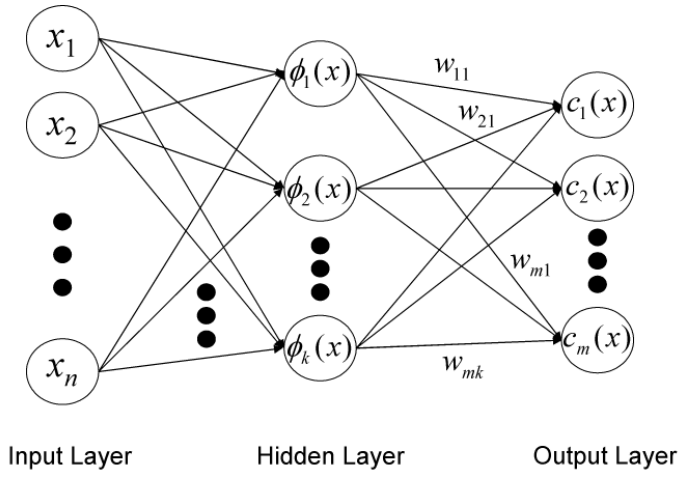

Fig. 2.3. Arquitectura de la Red Neuronal de Funciones de Base Radial

Donde N es el número de elementos del patrón de entrada x(n) y la función de base radial está definida por:

$$\Phi(n) = e^{-n^2} \tag{2.28}$$

Los valores de salida de la capa oculta son linealmente ponderados para la generar la salida final de la red:

$$y_k(n) = \sum_{i=1}^{M} w_{i,k} z_i(n) + \mu_k, k = 1, \dots K \tag{2.29}$$

Donde el número de neuronas de la capa oculta es M, K es el número de nodos de la capa de salida, $w_{i,k}$ son los pesos de la capa de salida, y μ_k son los umbrales de activación.

2.2.2.3 Redes de Regresión Generalizada

Es muy parecida a la Red de Base Radial, la diferencia se encuentra en la capa de salida donde se sustituye la ponderación lineal de la expresión (2.29) por una función de ponderación producto normalizada [34].

$$y_k(n) = \frac{\sum_{i=1}^{M} w_{i,k} z_i(n)}{\sum_{i=1}^{M} z_i(n)}, k = 1, \dots K \tag{2.30}$$

Donde el número de neuronas de la capa oculta es M, K es el número de nodos de la capa de salida y $w_{i,k}$ son los pesos de la capa de salida.

2.2.3 ANFIS

La técnica ANFIS (Adaptive Neural Fuzzy Inference System) ha sido ampliamente utilizada en identificación [35]. La Figura 2.4 muestra la estructura de capas que componen la topología ANFIS.

Está basada en la generación de un árbol de decisión para clasificar datos entre p^n modelos de regresión, donde el número de entradas es n y el número de particiones de cada variable es p.

Estas redes están compuestas por 5 capas. En la capa de entrada, también llamada capa de valor, las entradas son fuzzificadas. Esta capa está compuesta por p^n nodos, y la salida de cada nodo viene dada por:

$$O_{1,i} = \mu_{A,j}(x_k), i = 1, \dots p^n, j = 1, \dots p, k = 1, \dots n \tag{4.31}$$

Donde A es una etiqueta lingüística (pequeño, grande, …), y la función μ está dada por:

$$\mu_A(x_k) = \frac{1}{1 + \left|\frac{x_k - c_i}{a_i}\right|^{2b}} \tag{4.32}$$

Con las salidas de los nodos en la primera capa se disparan las reglas difusas de la segunda capa (capa de reglas), usando alguna norma (mínimo, producto, …):

$$O_{2,i} = w_i = T(\mu_{A,j}, \mu_{B,j}, \ldots), i = 1, \ldots p^n, j = 1, \ldots p \tag{4.33}$$

En la capa 3 (capa de normalización), las salidas de la capa 2 se normalizan:

$$O_{3,i} = \overline{w}_i = \frac{w_i}{\sum_{i=1}^{p^n} w_i}, i = 1, \ldots p^n \tag{4.34}$$

En la capa 4 (capa de funciones), las salidas de la capa 3 (w_i), ponderan los resultados de la función lineal de su nodo, generando las reglas de salida:

$$O_{4,i} = \overline{w}_i f_i = \overline{w}_i (p_{1,i}\, x_i + \cdots\, p_{n,i}\, x_n), i = 1, \ldots p^n \tag{4.35}$$

Donde $p_k, k = 1, .. n$ son los parámetros consecuencia. Al final, en la capa de salida (capa 5) se lleva a cabo una suma ponderada de las salidas de la capa 4:

$$O_{5,i} = \sum_{i=1}^{p^n} \overline{w}_i f_i = \frac{\sum_{i=1}^{p^n} w_i f_i}{\sum_{i=1}^{p^n} w_i} \tag{4.36}$$

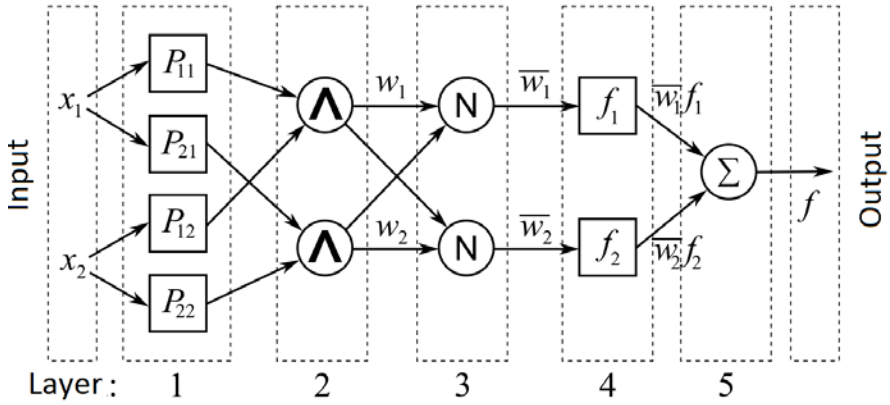

Fig. 2.4. Estructura de capas de la topología ANFIS [35]

3 Modelado de Cuatrirrotores

Este capítulo comienza describiendo el estado del arte citando las principales contribuciones de los diferentes autores en el modelado de cuatrirrotores a partir de técnicas analíticas, neuronales y neurodifusas (Sección 3.1). Después se presenta brevemente un modelo analítico del cuatrirrotor (Sección 3.2). En las secciones 3.3 y 3.4 se expone el procedimiento para la obtención de modelos neuronales y neuro-difusos empleado durante la Tesis. Finalmente se presenta la nueva técnica de hibridación desarrollada durante esta Tesis (Sección 3.5).

3.1 ESTADO DEL ARTE

En los últimos años la investigación y el desarrollo de vehículos aéreos no tripulados ha recibido una gran atención en la comunidad científica [3-10]. Centrándonos en el modelado, se pueden encontrar diferentes estrategias en la literatura, desde modelos analíticos basados en las ecuaciones físicas que representan el comportamiento dinámico del sistema hasta modelos inteligentes basados en técnicas de soft computing [36].

Uno de los primeros modelos analíticos del comportamiento dinámico de un helicóptero fue propuesto por Gessow y Myers en 1952 [37]. Desde entonces se han realizado diferentes aportaciones por otros autores, comenzando por los helicópteros, como el trabajo de Bramwell en 1976 [38] y después ampliándose para cualquier tipo de UAV. Modelos analíticos de helicópteros también aparecen en Cai [39], Budiyono [40] y El-Saadany [41]. Del Cerro [42] presenta un modelo híbrido, analítico y empírico, donde los parámetros se calculan mediante algoritmos genéticos. Un estudio muy completo de diferentes modelos matemáticos y estrategias de control para diferentes tipos de vehículos aéreos aparece en Castillo et al. [43].

Entre estas aproximaciones matemáticas, varios autores obtienen modelos de diferentes UAVs mediante la aplicación de técnicas PEM. Por ejemplo, Wu et al. [44] aplica ARX para identificar el modelo de un micro-vehículo aéreo en vuelo basándose en la evaluación de las medidas de la orientación del UAV. Morris et al. [45] y Cai et al. [39] usan PEM basados en el espacio de estados para identificar el modelo del vehículo completo o un subsistema del mismo, como por ejemplo la variación del ángulo de *yaw*. La propuesta de Mettler [46] es ampliamente citada por otros autores como un modelo complejo pero efectivo. Otro modelo paramétrico de un helicóptero es obtenido aplicando un método en el dominio de la frecuencia, obtenido por la Armada de los EE. UU. (Estados Unidos) y la NASA, conocido como CIFER (Comprehensive

Identification from Frequency Responses) [47]. Aunque CIFER fue desarrollado específicamente para helicópteros, se ha empleado satisfactoriamente en un amplio rango de vehículos de alas fijas y alas con hélices, y también en aplicaciones aéreas no convencionales [48].

Otros autores también utilizan estructuras paramétricas clásicas para identificar el modelo, aunque después aplican estrategias avanzadas para ajustar los valores de los parámetros. Algunos de ellos aplican técnicas de soft computing para hacer el ajuste. Por ejemplo, Ahmad et al. emplean el modelo paramétrico NARX (Nonlinear ARX) y aplican redes neuronales para ajustar los parámetros [49]. Otro ejemplo es el de Cerro et al. [42] donde el ajuste de los parámetros es realizado mediante algoritmos genéticos.

Hashimoto et al. [50] comparan el rendimiento de la identificación de modelos de helicópteros no tripulados obtenidos mediante las estructuras PEM: ARX, ARMAX, OE y BJ. En nuestro estudio también hemos aplicado ARX, ARMAX, OE y BJ pero, desafortunadamente, Hashimoto et al. no proporcionan resultados cuantitativos, por lo que no ha sido posible comparar nuestros resultados con sus modelos. Yuan [51] también aplica el método OE para estimar el modelo de un helicóptero no tripulado. Manai [52] emplea la estructura OE para la identificación de un UAV con el propósito de diseñar controladores.

Algunos trabajos teóricos rigurosos han probado que, incluso con una sola capa oculta, las redes neuronales pueden utilizarse como aproximadores universales de funciones [53]. De ahí que se empleen comúnmente para modelar sistemas no lineales como por ejemplo los UAVs.

Las redes MLP y RBF son las más usadas para simular el pitch, el roll, el yaw y la posición de estos vehículos [54-61]. También se emplea una RBF en [62]. Las redes neuronales basadas en modelos no lineales autorregresivos con entradas exógenas (modelos NARX) se han adoptado debido a su buen comportamiento simulando especialmente sistemas complejos no lineales [63].

En [64] se emplean tres métodos para obtener los parámetros de un modelo cuatrirrotor: el Iterative Bi-Section Shooting method, IBSS, una red neuronal artificial, y un método híbrido ANN-IBSS. En este artículo, al igual que en nuestro caso, el método híbrido proporciona mayor precisión que los métodos aplicados individualmente. Otra técnica de hibridación es propuesta por Jin en [65], en este caso combinando RBF y ARX. Puttige [66] resuelve el modelado con una arquitectura multi-red y la combinación de métodos off-line y on-line.

La lógica difusa también se ha aplicado en la identificación de UAVs, pero en menor medida. Hay algunos trabajos que pueden ser resaltados, como el de Salman, quién obtiene el modelo del UAV aplicando modelado de espacio de estados y lógica difusa, y después los compara [67]. Encontramos otros ejemplos en los que se emplean redes neuro-difusas en el modelo de sistemas complejos no lineales, como en [68-70].

3.2 MODELO ANALÍTICO

Un vehículo cuatrirrotor está compuesto de cuatro brazos perpendiculares, cada uno de ellos con un motor y una hélice (Figura 3.1). Los cuatro motores proporcionan el control de la sustentación y de la dirección.

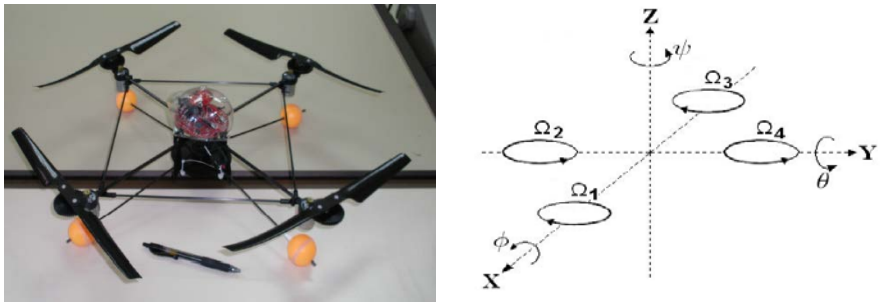

Fig. 3.1. Un vehículo cuatrirrotor (izquierda) y el sistema de coordenadas del UAV (derecha)

La posición absoluta del UAV se describe por tres coordenadas, (x, y, z), y la orientación viene dada por los tres ángulos de Euler (ϕ, θ, ψ). Se deben cumplir ciertas restricciones en los ángulos de Euler para que no se produzca un efecto indeseado conocido como el "gimbal lock", es decir la pérdida de uno de los grados de libertad cuando dos de los ejes de rotación están alineados. Estas restricciones son: $(-\pi \leq \psi < \pi)$ para el ángulo de guiñada (yaw), $(-\pi/2 \leq \phi < \pi/2)$ para el ángulo de balanceo (roll), y $(-\pi/2 \leq \theta < \pi/2)$ para el ángulo de inclinación (pitch), todos ellos en radianes. Estas restricciones no serían necesarias en un modelo basado en cuaterniones.

El sistema se basa en dos pares de hélices opuestas, (1, 3) y (2, 4) (Figura 3.1, derecha). Para mantener el sistema balanceado, dos motores giran en el sentido de las agujas del reloj, mientras los otros dos giran al contrario. El incremento de la velocidad del motor 3 respecto del 1 produce un pitch positivo ($\theta>0$), mientras que incrementar la velocidad del 4 respecto del 2 proporciona un roll positivo ($\phi>0$). El incremento de las velocidades de los motores 1+3 respecto de los motores 2+4 proporciona un yaw positivo ($\psi>0$).

Usando el método de Newton-Euler, la dinámica angular del sistema se representa [71]: (2.1):

$$\tau = J\dot{\omega} + \omega \times J\omega \qquad (3.1)$$

$$J = \begin{pmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{pmatrix} \qquad (3.2)$$

Donde τ es el vector de par en los tres ejes en N·m, J es el tensor de inercia en Kg·m^2 (3.2), ω es el vector de velocidades angulares en rad/s, and × representa el producto vectorial.

La dinámica traslacional viene dada por (3.3):

$$m\dot{v} = RT - mge_3 \qquad (3.3)$$

Donde m es la masa del sistema en Kg, R es la matriz de rotación adimensional, g es la aceleración de la gravedad en m/s^2, T es un vector de fuerzas en N, y $e_3 = [0,0,1]^T$ es un vector unitario que describe la orientación del cuatrirrotor.

Los vectores τ (3.4) y T (3.5) vienen dados en función de las velocidades de las hélices:

$$\tau = \begin{pmatrix} bl(\Omega_4^2 - \Omega_2^2) \\ bl(\Omega_3^2 - \Omega_1^2) \\ d(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 - \Omega_3^2) \end{pmatrix} \qquad (3.4)$$

$$T = \begin{pmatrix} 0 \\ 0 \\ b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \end{pmatrix} \qquad (3.5)$$

En las ecuaciones (3.4-3.5), b es el coeficiente de empuje en N·s^2, d es el coeficiente de arrastre, l es la longitud de cada brazo en m, y $\Omega_1, \dots, \Omega_4$ son las velocidades en rad/s de los motores 1 a 4, respectivamente.

Para simplificar los cálculos, en vez de usar la velocidad de los motores, es posible definir un conjunto de señales de control:

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -1 & 0 & 1 \\ -1 & 0 & 1 & 0 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} \qquad (3.6)$$

Esta matriz es invertible, así que es posible generar referencias de velocidad para los motores desde un conjunto de señales de control.

Finalmente, de las ecuaciones 3.1 a 3.6, se deriva el siguiente sistema de ecuaciones:

$$\ddot{\phi} = \dot{\theta}\dot{\psi}\,(I_y - I_z)/I_x + (lb/I_x)u_2 \qquad (3.7)$$

$$\ddot{\theta} = \dot{\phi}\dot{\psi}\,(I_z - I_x)/I_y + (lb/I_y)u_3 \qquad (3.8)$$

$$\ddot{\psi} = \dot{\phi}\dot{\theta}\,(I_x - I_y)/I_z + (d/I_z)u_4 \qquad (3.9)$$

$$\ddot{X} = -(sin\theta cos\phi)\,(b/m)u_1 \qquad (3.10)$$

$$\ddot{Y} = (sin\phi)\,(b/m)u_1 \qquad (3.11)$$

$$\ddot{Z} = -g + (cos\theta cos\phi)\,(b/m)u_1 \qquad (3.12)$$

3.3 MODELOS NEURONALES

Para comprobar la eficacia de las técnicas neuronales, durante el desarrollo de la Tesis se realizó un estudio comparativo de modelos obtenidos mediante diferentes técnicas PEM y redes neuronales con diferentes estructuras y configuraciones. Los resultados preliminares se publicaron inicialmente en [72]. En concreto se compararon los modelos PEM: ARX, OE y basados en el espacio de estados con modelos obtenidos a partir de redes neuronales artificiales con las siguientes estructuras: perceptrón multicapa, red de base radial y redes de regresión generalizada (para más detalles sobre estas estructuras consultar el Capítulo 2). Para comparar los resultados se empleó el MSE y el MSE de la respuesta al escalón estimada.

Para realizar el estudio, el conjunto de datos de entrada de la red se formó con la señal de entrada en el instante actual $pitch(t)$ y los valores en algunos instantes de muestreo anteriores, $pitch(t - Ki)$ donde $i = 1 \ldots 100/K$ para diferentes valores de K. El conjunto de datos objetivo se generó con los valores de la señal de salida en el instante actual $vx(t)$ (velocidad en el eje X). Se emplearon 5000 muestras de las señales, el 60% se usó en el entrenamiento y el resto para la simulación. Para obtener más detalles sobre los datos de entrenamiento puede consultarse la Sección 2.1. Tanto gráficamente como cuantitativamente se comprueba que mediante las redes neuronales se obtienen mejores resultados que con los modelos PEM.

En otro trabajo posterior [73] se analiza la utilidad del aprendizaje on-line de las redes neuronales artificiales para la construcción de modelos de UAVs. En este caso el conjunto de datos de entrada se construye con las señales pitch, roll y la velocidad en el eje X para el modelo de la velocidad en el eje X (v_x), es decir $[roll(t-i), pitch(t-i), v_x(t-i)] \longrightarrow v_x(t)$ y con las señales pitch, roll y la velocidad en el eje Y (v_y) para el eje Y $[roll(t-i), pitch(t -$

$i), v_y(t-i)] \longrightarrow v_y(t)$, donde $i = 1 \ldots N$ con diferentes valores de N en cada señal, N_θ para la señal *roll*, N_Φ para la señal *pitch*, N_{vx} para v_x y N_{vy} para v_y. Se comprueban diferentes combinaciones de $\{N_\theta, N_\Phi, N_{vx}\}$ y $\{N_\theta, N_\Phi, N_{vy}\}$ buscando las que proporcionan menor MSE. El objetivo de este estudio es contrastar las ventajas del aprendizaje on-line respecto del aprendizaje off-line. En efecto, se observa como las redes entrenadas con aprendizaje off-line son mucho más sensibles al conjunto de datos de entrenamiento; en cambio, con el aprendizaje on-line, como la red aprende con el paso del tiempo se reduce este sesgo.

3.4 MODELOS NEURO-DIFUSOS

Tras la constatación de la idoneidad de las redes neuronales para el modelado de UAVs se estudió si las redes neuro-difusas también podían proporcionar buenos resultados. Para ello se obtuvieron diferentes modelos mediante las técnicas ARX (PEM), redes de base radial y ANFIS (Adaptive Neuro-fuzzy Inference System). Los resultados se publicaron en [74]. La estructura ANFIS se detalla en la Sección 2.2.3 de la Tesis.

En este caso, para realizar el estudio, el conjunto de datos de entrada de la red se formó con la señal de entrada en el instante actual, $pitch(t)$, y valores en los 5 instantes de muestreo anteriores, $pitch(t - iTs)$ donde $i = 1 \ldots 5$, y Ts es tiempo de muestreo. El conjunto de datos objetivo se generó con los valores de la señal de salida en el instante actual $vx(t)$. En este artículo también se amplía el estudio para recoger la relación $roll \longrightarrow vy$.

La configuración del método ARX empleada fue $na = 10$, $nb = 10$ y $tk = 1$, donde na es el orden del numerador, nb el orden del denominador y tk el número de retardos de la señal de entrada. El número de neuronas de la capa oculta de la red de base radial se limitó a 30. En el caso del ANFIS, se asignó un número de particiones $p = 2$, con lo que se obtuvo un conjunto de 2^6=64 reglas. Del mismo modo que en el estudio previo, se emplearon 5000 muestras de las señales, donde el 60% se usaron en el entrenamiento y el resto en la validación. Otra aportación nueva en este estudio fue la utilización del MSE de la respuesta en frecuencia estimada, incluyendo de esta forma el comportamiento frecuencial en el análisis comparativo.

3.5 MÉTODO PROPUESTO: HIBRIDACIÓN DE TÉCNICAS PARAMÉTRICAS Y DEL SOFT COMPUTING

El modelado es una de las aproximaciones más importantes para trabajar con sistemas complejos. Una representación del sistema nos permite analizar su comportamiento y conseguir un conocimiento profundo sobre él. En ingeniería de control, la disponibilidad de los modelos de simulación es esencial para diseñar leyes de control y poder testearlas sin dañar los sistemas reales. Por lo tanto, uno de los pasos más útiles para controlar de forma efectiva un sistema físico es obtener modelos útiles y fiables que describan su comportamiento.

La importancia de tener un buen modelo crece con la complejidad del sistema. Por un lado, el modelo debe acercarse tanto como sea posible a la realidad pero, por otro lado, el modelo debería mantenerse tan simple como sea posible para reducir el coste computacional. Un coste computacional bajo es un factor habilitante clave para poder correr el modelo en plataformas hardware de propósito general de bajo coste. La idea subyacente de la propuesta de modelado híbrido es obtener modelos de sistemas complejos manteniendo un equilibrio adecuado entre precisión y recursos hardware necesarios para su ejecución.

Una vez que la técnica de modelado es seleccionada, el objetivo final es obtener modelos realistas y realizables. El realismo está directamente relacionado con la precisión del modelo. Por lo tanto, una de las premisas en el proceso de diseño debe ser reducir el error tanto como sea posible. Por otro lado, la factibilidad del modelo está unida a la simplicidad y la eficiencia. En otras palabras, los modelos más complejos son normalmente menos realizables y eficientes. Normalmente, estos son conceptos antagonistas; las mejoras en la precisión suelen causar aumentos de la complejidad y reducción de la factibilidad. De igual forma, los modelos más realistas suelen tener un mayor coste computacional.

En este sentido, las técnicas paramétricas basadas en PEM normalmente proporcionan modelos sencillos, pero con una baja precisión. Por otro lado, los métodos basados en técnicas inteligentes, como redes neuronales, proporcionan modelos que se ajustan mejor al comportamiento real de los sistemas, pero normalmente demandan mucha más memoria y capacidad de procesamiento. Por lo tanto, el mejor método será aquel que tenga la habilidad de unir precisión y simplicidad, combinándolas.

Esta idea sirvió de inspiración para el desarrollo del método que se describe a continuación. La precisión alcanzada por la aplicación independiente de técnicas de modelado puede ser suficiente para algunas aplicaciones, pero es posible que necesitemos mejorar su rendimiento. Para hacer esto, una primera opción es intentar modificar los parámetros de configuración del

modelo que fueron seleccionados inicialmente. Por ejemplo: los polinomios en las técnicas paramétricas, el tamaño de la capa de entrada de las redes neuronales, el número de neuronas, o el número de particiones en el método ANFIS. Cuando esto no funciona o funciona, pero el incremento del coste computacional es inaceptable, las técnicas híbridas pueden jugar un papel importante.

El método híbrido propuesto se compone de una topología más un proceso de entrenamiento. La topología del método híbrido propuesto se muestra en la Figura 3.2. Tal y como se puede apreciar, se basa en una aplicación en cascada de una técnica paramétrica y una técnica de soft computing. Un conjunto de señales de entrada, en el ejemplo de la figura sólo la señal $pitch$ en un caso, y $roll$ en otro caso, alimentan a un modelo paramétrico PEM. El modelo PEM genera una única salida que sirve de entrada a la técnica de soft computing. El modelo PEM se emplea para aproximar el comportamiento lineal del sistema y la técnica de soft computing para el ajuste fino de las no linealidades. Es decir, realizamos un ajuste cualitativo o heurístico de una técnica paramétrica.

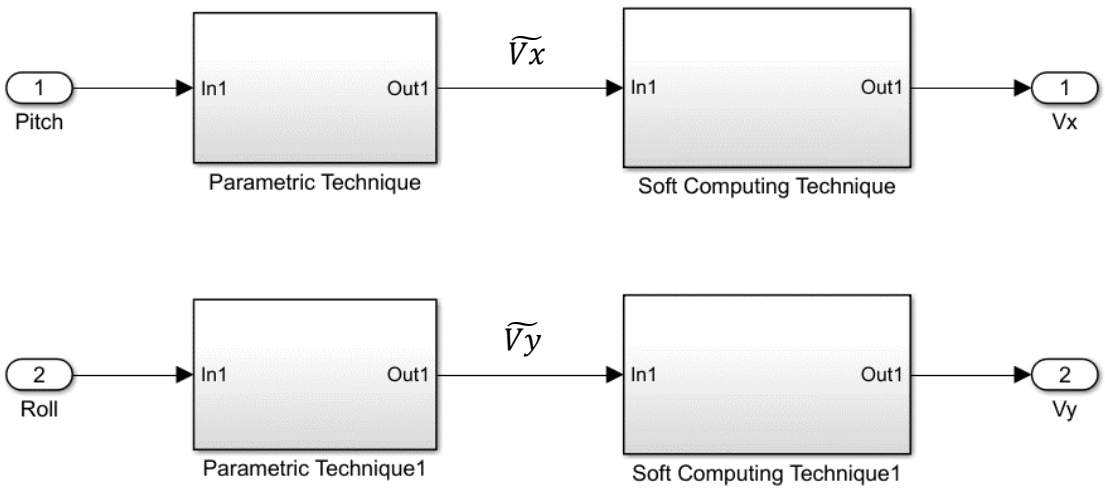

Fig. 3.2. Conexión de técnicas en cascada

El proceso para obtener un modelo completo se compone de diferentes pasos:

1. **Identificación de entradas dominantes:**

   Solamente es necesario este paso cuando trabajamos con un sistema MIMO o MISO. Se somete al sistema a un escalón en sus entradas para identificar la entrada dominante en cada salida. Por ejemplo, en el caso del UAV (sistema MIMO), la Figura 3.3 muestra la respuesta al escalón del UAV, para las velocidades vx (fila superior)

y vy (fila inferior), respecto de las señales $pitch$ (columna izquierda) y $roll$ (columna derecha).

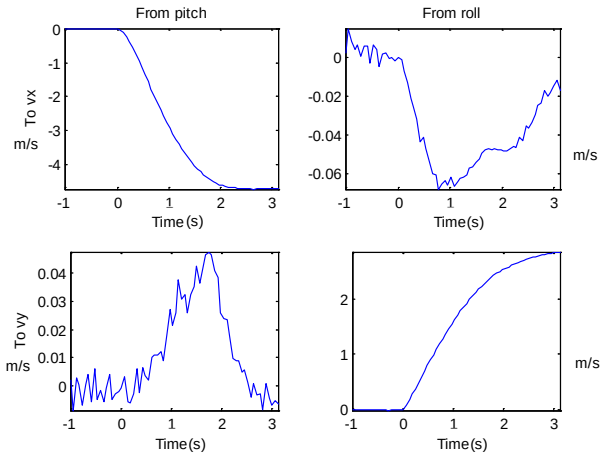

Fig. 3.3. Respuesta al escalón del sistema MIMO

En la figura anterior es posible observar que la ganancia estacionaria de los elementos de la diagonal principal es mucho mayor que la del resto de los elementos. Este hecho indica que el sistema puede ser parcialmente desacoplado y, por tanto, el sistema MIMO puede convertirse en dos sistemas SISO. Considerando esta simplificación, nos centraremos en la influencia del $pitch$ en la velocidad vx y el $roll$ en la velocidad vy. Por esta razón, en la Figura 3.2 aparece el modelado en paralelo de los sistemas $pitch \rightarrow vx$ y $roll \rightarrow vy$.

2. **Identificación de modelo PEM**:

Empleando datos de entrada/salida reales del sistema, por ejemplo en la Figura 3.2 la entrada es el $pitch$ y la salida es vx, se encuentra el modelo PEM que mejor se aproxima a la respuesta del sistema. Para ello, para cada posible estructura PEM (ARM, ARMAX, ARIMAX, BJ, OE, …) se generan tantos modelos como posibles combinaciones haya de parámetros de configuración. Por ejemplo, la estructura ARX tiene 2 parámetros de configuración: Na y Nb. Las primeras 100 configuraciones se generan obteniendo las diferentes combinaciones de Na y Nb, siendo Na={1,…10} y Nb={1….10}. Para cada estructura PEM se escoge la configuración del modelo que mejor se ajuste al comportamiento del sistema. Finalmente se escoge la mejor estructura PEM.

3. **Simulación de la salida del modelo PEM:**

   La salida esperada del sistema es simulada con el modelo paramétrico aplicando un conjunto de señales conocidas previamente. En el ejemplo de la Figura 3.2 a la salida del modelo PEM se obtienen las señales $\widetilde{vx}$ y $\widetilde{vy}$. Estas son las salidas que esperaríamos obtener en el sistema si fuera puramente lineal.

4. **Entrenamiento de la técnica de soft computing**: La técnica de soft computing es entrenada usando el resultado de la simulación obtenida en el punto anterior como conjunto de datos de entrada, y los datos reales de salida conocidos como conjunto de datos de salida. En el ejemplo de la Figura 3.2, la entrada sería la señal estimada $\widetilde{vx}$ y la salida la señal real vx para uno de los modelos, y $\widetilde{vy}$ y vy para el otro modelo.

Para llegar hasta este método se comenzó estudiando la posibilidad de combinar las técnicas PEM y neuronales, en paralelo y en cascada. Los resultados iniciales se publicaron en [75] Como se observó un mejor comportamiento en la combinación en cascada, a partir de ese momento se emplea esta arquitectura. Más tarde se comprueba que la arquitectura propuesta también es válida para la combinación de técnicas PEM y neuro-difusas, los resultados se publican en [74]. El método se consolida en [76], donde además se constata que es capaz de mejorar la precisión y reducir la complejidad computacional simultáneamente.

3.5.1 Resultados

Para comprobar la validez de la propuesta, durante el desarrollo de la tesis se obtuvieron diferentes modelos tanto aplicando las técnicas paramétricas, neuronales y neurodifusas individualmente, como combinándolas a través de la técnica de hibridización propuesta. La Figura 3.4 muestra las mejores combinaciones de técnicas. Para cada eje, la línea azul oscura es la salida del sistema real, las líneas verde, roja, azul clara, púrpura y amarilla representan los valores de la señal de salida obtenida por los modelos ARX-(10,10,1), RBF-(30,200), ANFIS-(8,2), ARX-(10,10,1)-RBF-(5,50), and ARX-(10,10,1)-ANFIS(5,2), respectivamente. Como podíamos esperar las señales obtenidas sólo con los modelos ARX son las peores para ambos ejes. Por otro lado, los métodos híbridos, ARX-RB y ARX-ANFIS, proporcionan mejor rendimiento que las técnicas aplicadas individualmente, incluso aunque la técnica esté configurada

con los mejores parámetros encontrados (el mayor tamaño de la capa de entrada y el mayor número de neuronas en la capa oculta).

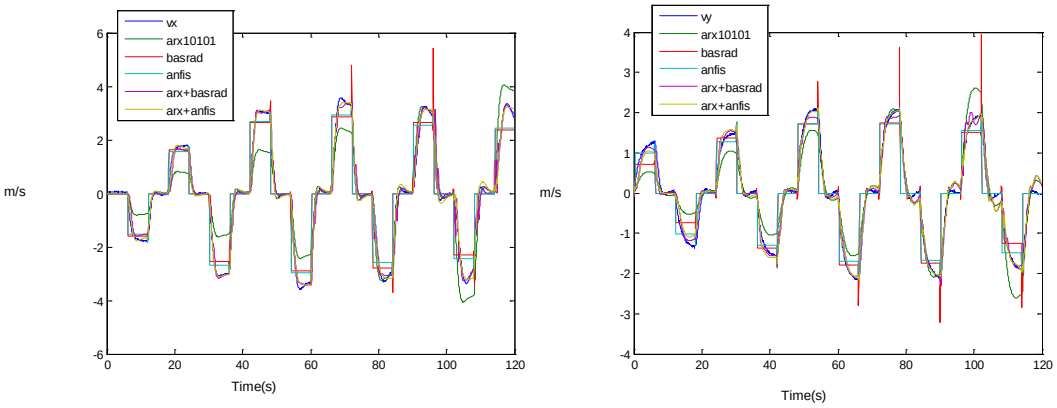

Fig. 3.4. Comparación de velocidad en el eje x (izquierda) y en el eje y (derecha) obtenidas con diferentes métodos

En la Tabla 3.1 hay tres franjas, cada una de ellss separadas del resto con una línea horizontal. El primer grupo de filas muestra los resultados obtenidos mediante la aplicación individual de las técnicas seleccionadas. El segundo grupo de filas presenta los resultados de los modelos generados por la hibridación de las técnicas inteligentes y paramétricas. El tercer grupo también muestra el resultado de las técnicas híbridas, pero en este caso con un ligero aumento de la complejidad de la configuración; los parámetros de la red neuronal cambian de (5,30) a (5,50) o a (10,50). Para cada columna y para cada fila principal, los mejores resultados se han marcado en negrita, y para cada columna los mejores resultados se han subrayado.

La última columna de la Tabla 3.1 representa la complejidad temporal (TC). Analizando este indicador junto con el MSE_C, es interesante puntualizar las diferencias entre las técnicas provenientes del soft computing, tales como RB-(30,200), y su correspondiente método híbrido (ARX-RB). En este caso, TC=38136 frente a TC: 2356.6. Por un lado, RB es más preciso que la técnica paramétrica ARX (0.0995 vs. 0.1746 MSE_C), pero es mucho más compleja en términos de tiempo computacional (38136 vs. 91.6). Sin embargo, con la hibridación en ARX-RB la precisión es ahora 0.0170 mientras que el TC es reducido a 2356.6.

En general, (Tabla 3.1) tal y como se preveía la técnica híbrida propuesta proporciona mejores resultados que las técnicas aplicadas individualmente, obteniendo importantes reducciones en el tiempo computacional (mayor simplicidad) sin sacrificar la precisión.

Tabla 3.1. Comparación en términos de MSE y TC para diferentes técnicas y combinaciones

Técnica	v_x				v_y				TC
	MSE_T	MSE_S	MSE_F	MSE_C	MSE_T	MSE_S	MSE_F	MSE_C	
ARX10,10,1	0.5056	**0.0122**	**0.0060**	0.1746	0.1326	0.0024	**0.0077**	0.0476	**91.6**
OE6,6,1	0.2915	0.0418	0.0352	0.1228	0.0820	**0.0010**	0.0085	**0.0305**	54.56
RB-(5,30)	0.5097	1.1919	3.4022	1.7013	0.2345	0.3796	1.5013	0.7051	1497.9
RB-(30,200)	**0.0240**	0.0171	0.2574	**0.0995**	**0.0159**	0.0145	0.2729	0.1011	38136
ANFIS-(5,2)	0.4816	1.1807	6.1789	2.6137	0.2106	0.3765	2.1761	0.9211	3743.8
ANFIS-(8,2)	0.3363	0.8236	4.9864	2.0488	0.1531	0.2731	1.7381	0.7214	36276
ARX-ANFIS(5,2)	**0.0145**	**0.0087**	**0.0252**	***0.0161***	**0.0100**	0.0057	0.0198	0.0118	3835.4
OE-ANFIS(5,2)	0.0295	0.1356	0.1433	0.1028	0.0103	0.0030	0.0281	0.0138	3798.4
ARX-RB(5,30)	0.0572	0.0172	0.0774	0.0506	0.0144	**0.0015**	**0.0059**	**0.0073**	1589.5
OE-RB(5,30)	0.0429	0.1163	0.0848	0.0813	0.0127	0.0039	0.0162	0.0109	**1552.5**
ARX-RB(5,50)	0.0125	**0.0031**	0.0355	**0.0170**	0.0068	0.0036	0.0108	0.0071	**2356.6**
OE-RB(5,50)	0.0286	0.1114	0.0754	0.0718	0.0071	0.0007	0.0110	0.0063	**2356.6**
ARX-RB(10,50)	**0.0103**	0.0040	**0.0340**	0.0287	0.0059	0.0022	0.0108	0.0063	3764.1
OE-RB(10,50)	0.0270	0.0863	0.0445	0.0526	**0.0057**	**0.0004**	**0.0078**	***0.0046***	3727.1

La mayoría de los trabajos anteriores presentados en la sección del estado del arte no aportan resultados cuantitativos comparables a los de la Tabla 3.1. Sólo en [56] se aportan datos MSE para una variable de velocidad. En este caso, Puttige plantea 3 modelos neuronales para identificar un UAV de alas fijas, con el modelo entrenado offline obtiene un MSE de 1.64 en la velocidad, con el modelo online 0.0534 y un MSE de 0.0088 con el modelo combinado. En uno de los trabajos desarrollados durante la tesis [73], también se observa como los modelos con aprendizaje online proporcionan mejores resultados que los modelos con aprendizaje offline.

4 Control de Cuatrirrotores

Este capítulo describe los conceptos y métodos usados para controlar el cuatrirrotor, así como los principales resultados obtenidos. La Sección 4.1 cita alguna de las principales contribuciones de diferentes autores en el control de cuatrirrotores a partir de técnicas neuronales. La Sección 4.2 se centra en el método de control propuesto basado en redes neuronales artificiales. Después se exponen los modelos de perturbaciones empleados (Sección 4.3). Finalmente se estudia el nuevo esquema de control anti-perturbaciones desarrollado durante la Tesis (Sección 4.4).

4.1 ESTADO DEL ARTE

Las redes neuronales han sido ampliamente utilizadas para el control de UAVs. Uno de los primeros artículos reseñables es el de Kim y Calisse [77-78], quienes presentan el desarrollo teórico de una arquitectura de control adaptativo basada en redes neuronales. Después de ellos, pueden localizarse muchos autores que utilizan técnicas basadas en redes neuronales para el control de UAVs. Mo y Farid recogen un extenso y reciente estudio del estado del arte sobre control inteligente adaptativo aplicado a UAVs [79]. A continuación, se citan algunos trabajos remarcables en este área.

Efe entrena una red neuronal para proporcionar los coeficientes de un filtro FIR, que aproxima la respuesta de un controlador PI $^{\lambda}$ D $^{\mu}$ con parámetros y órdenes de integración y derivación variables [80]. El filtro FIR consigue una respuesta similar al $PI^{\lambda}D^{\mu}$ pero se consigue reducir el coste computacional. En [81] se emplea un controlador neuronal jerárquico para controlar micro-cuatrirrotores. En [82] se propone un controlador neuronal adaptativo para estabilizar el cuatrirrotor frente a errores en el modelado y perturbaciones externas. El mismo objetivo se plantea en [83], pero resolviéndolo de forma diferente; en este caso se emplea un sistema neuro-difuso construido mediante un sistema de inferencia borroso Takagi-Sugeno (T-S), donde la sección consecuencia se construye con un conjunto de redes neuronales diferenciales. En [84] se emplea una red de una capa oculta basada en ELM (Extreme Learning Machine) para aproximar la dinámica no modelada. De forma similar en [85] también se emplean redes neuronales para aproximar las incertidumbres en la dinámica del UAV.

En varios artículos se emplean las redes como observadores. En [86] se emplea un observador neuronal para estimar la velocidad traslacional y angular del UAV. En [87] los errores de

control se estiman mediante un observador de estado neuronal. En [88-89] se propone un observador de modo deslizante neuronal para la reconstrucción de estados. Este observador tiene la misma estructura que un observador de modo deslizante simple, pero para reducir la sensibilidad al ruido se añade una red neuronal artificial como un término corrector. Estos observadores pueden ser especialmente útiles para el desarrollo de controladores anti-perturbaciones, como veremos a continuación.

En líneas generales hay dos estrategias de diseño de controladores respecto a su comportamiento frente a perturbaciones: algunos reguladores se diseñan sin considerar las perturbaciones, y después estas se introducen en diferentes puntos del sistema de control para examinar su influencia y comprobar su robustez por el contrario, existe otra categoría de controladores que están especialmente diseñados para reaccionar frente a las perturbaciones atenuándolas o rechazándolas.

En este sentido, en [90] se presenta un estudio sobre estrategias de control anti-perturbaciones. Guo identifica dos grupos de controladores anti-perturbaciones: los métodos de atenuación de perturbaciones (tales como la teoría de control estocástico y la teoría de control robusto) y los esquemas de rechazo de perturbaciones. Los métodos de rechazo de perturbaciones a su vez incluyen: control de modelo interno, teoría de regulación de salida, rechazo de perturbaciones activo (ADRC), el control de modelos embebidos, y los controladores basados en observador de perturbaciones (DOBC). La Figura 4.1 muestra como el observador de perturbaciones puede introducirse en un lazo de control. Para los sistemas lineales, el control del modelo interno puede ser utilizado para compensar la entrada de perturbación con condiciones de estabilidad neutras. La clásica teoría de regulación de salida es aplicable a las perturbaciones descritas con modelos exógenos [91-93]. Son difíciles de obtener métodos de compensación de perturbaciones fiables cuando el modelo o los parámetros exógenos del modelo son desconocidos. Para un sistema no lineal con perturbaciones se propuso un método de control adaptativo con modelo interno, donde se aplicó control adaptativo al problema de la regulación de salida en [94-96]. Así mismo, se han empleado métodos de rechazo de perturbaciones basados en la estimación adaptativa con modelos inciertos o parámetros desconocidos [97-98]. Para la teoría de regulación de salida no lineal, la solución se basa en las ecuaciones diferenciales parciales no lineales de Francis–Isidori–Byrnes (PDE) [99]. Por otra parte, la teoría del ADRC tiene muy buenas perspectivas para aplicaciones en ingeniería, pero aún está pendiente de una prueba teórica rigurosa de su estabilidad [100].

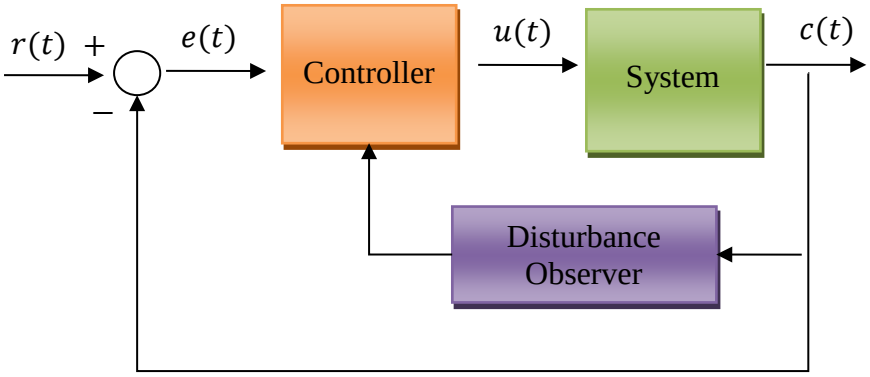

Fig. 4.1 Lazo de control con observador de perturbaciones

Si nos centramos en el control de cuatrirrotores, hay pocos trabajos que estudien el efecto de la variación de la carga en la dinámica del cuatrirrotor y la tengan en cuenta. En [101] un control adaptativo se diseña para mitigar el impacto de la variación de parámetros estimándolos bajo unas condiciones de rendimiento prefijadas. En [102] se emplea mínimos cuadrados y métodos de gradiente para estimación de parámetros adaptativa, los cuales se usan para adaptar las salidas del control a la masa y la inercia actual del UAV. Recientemente Wang [103] aplica de nuevo la misma estrategia para estimar las variaciones en la carga y el efecto del viento.

Otros estudios se centran en el rechazo de las perturbaciones del viento. En [104] se emplean observadores basados en Lyapunov para estimar las perturbaciones producidas por fuerzas externas. En [105] se propone una estrategia de control basada en modo deslizante y técnicas de control adaptativo para lidiar con condiciones de viento variables tanto rápidas como lentas. En [106] se presenta un controlador de orientación predictivo de modelo conmutable para un UAV sujeto a perturbaciones atmosféricas. En [107] se diseña un controlador de realimentación de estado adaptativo no lineal para control de sustentación y par, que garantiza la convergencia global del seguimiento de trayectorias en la presencia de perturbaciones de viento constantes. En [108] un control de modo deslizante basado en un observador de perturbaciones de modo deslizante (SMC-SMDO) se emplea para diseñar un controlador de vuelo robusto para un cuatrirrotor pequeño.

Los trabajos anteriores se han centrado principalmente en las variaciones de la carga o en el rechazo de la influencia del viento. Sólo se ha encontrado un estudio reciente que intente resolver ambos problemas al mismo tiempo [103]. Este enfoque ha sido abordado durante el desarrollo de la tesis, como veremos más adelante, y los resultados fueron publicados en [109].

4.2 MÉTODO PROPUESTO: CONTROLADOR NEURONAL ADAPTATIVO

La habilidad de las redes neuronales para aproximar funciones permite que sean utilizadas en el diseño de controladores. Hay diferentes estrategias de control basadas en redes neuronales: control directo, control inverso, aprendizaje indirecto, aprendizaje generalizado, aprendizaje especializado… Una clasificación sistemática se presenta en [110]. Durante el desarrollo de la Tesis se empleó una variante del algoritmo de aprendizaje generalizado (GLA) para controlar el cuatrirrotor [111]. La Figura 4.2 muestra la estructura del GLA.

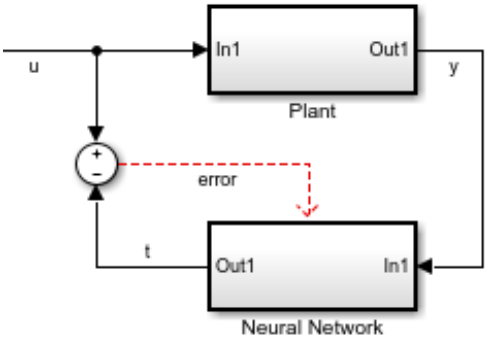

Fig. 4.2.Algoritmo de aprendizaje generalizado

En el aprendizaje generalizado [112], un conjunto de comandos, denotado $\{u\}$ se emplea para dirigir al sistema, y se obtiene un conjunto de trayectorias resultantes $\{y\}$. La red neuronal artificial (en la literatura científica a veces se le llama Sistema Dinámico Inverso, Inverse Dynamics System o IDS) recibe $\{y\}$ como entrada y genera un conjunto de comandos referencia $\{t\}$. El objetivo del aprendizaje generalizado es minimizar los errores cuadráticos entre $\{t\}$ y $\{u\}$. Después de entrenar la red neuronal, si una entrada real $\{y'\}$ está suficientemente cerca de una trayectoria del conjunto $\{y\}$, el controlador debería poder devolver el comando $\hat{t}$ adecuado, haciendo que el movimiento actual $\hat{y}$ siga a y'.

La variación propuesta respecto del GLA es el refinamiento on-line junto con la combinación con un PID. El primer paso es la aplicación del GLA para entrenar off-line la red neural con el objetivo de identificar la inversa de la planta (Figura 4.2). Una vez que la red es entrenada off-line, se coloca en cascada un controlador PID. Entonces, la configuración de la red es refinada on-line. Para hacerlo, en cada intervalo de control dos procesos son secuencialmente aplicados a la red (primero la simulación, después el aprendizaje on-line):

1. **Simulación:** La salida del PID, $PID_{OUT}(t_i)$, alimenta una de las entradas de la red neuronal artificial; el resto de las entradas son valores pasados de la planta,

$PLANT_{OUT}(t_i - j * Ts)$. La red genera la entrada de control, u_1, la cual es la entrada de la planta (Figura 4.3, conmutador en la posición superior).

$$u_1(t) = f_{NET}\left(PID_{OUT}(t), PLANT_{OUT}(t - i * Ts), par_{NET}(t - Ts)\right) \quad i = 1 \dots (N_{inputs} - 1) \quad (4.1)$$

2. **Aprendizaje On-line:** La red neuronal se entrena de nuevo con la salida actual y previas de la planta, para generar la salida de control, u_1, obteniendo los nuevos parámetros de configuración par_{NET}. El conjunto de datos de entrada de la red se construye con los valores anteriores de la planta, $PLANT_{OUT}(t_i - j * Ts)$. El conjunto de datos de salida es el valor actual de la entrada de la planta $u_1(t_i)$ (Figura 4.3, conmutador en la posición inferior)

$$par_{NET}(t) = f(PLANT_{OUT}(t - i * Ts), u_1(t), par_{NET}(t - Ts)) \quad i = 0 \dots (N_{inputs} - 1) \quad (4.2)$$

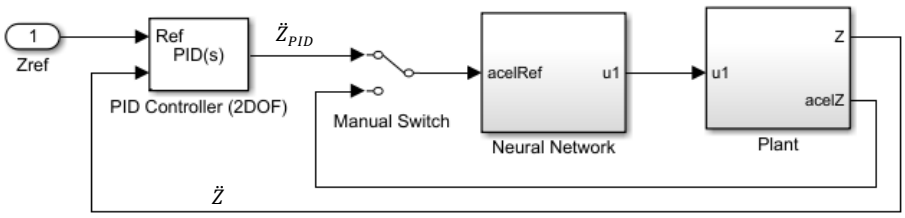

Fig. 4.3. Estrategia de control de altitud neuro adaptativa

En la Figura 4.3 se muestra una configuración específica del método propuesto aplicándolo al control de altitud. Los UAVs normalmente vienen provistos con acelerómetros, por lo que asumimos que la aceleración en el eje z estará disponible ($\ddot{Z}$). La red debe poder simular la señal de control u_1 usando las medidas de aceleración en Z. Gracias a la red neuronal artificial, el PID no necesita incluir la ganancia de la planta. La red puede aprender la ganancia de la planta y trabajar con ella. En otras palabras, con este método no es necesario conocer los parámetros del sistema para poder controlarlo [111].

Otra característica a remarcar de nuestra propuesta es la capacidad de aprendizaje on-line. Ésta permite a la red aprender de forma continua mientras el sistema está en funcionamiento y adaptarse a las variaciones con el paso del tiempo. Esto es extremadamente útil cuando modelamos sistemas sujetos a cambios, como por ejemplo, sistemas que funcionen al aire libre con condiciones meteorológicas cambiantes o sujetos a acciones externas que puedan modificar los parámetros del sistema. Ambas condiciones se pueden aplicar al control de cuatrirrotores. Las redes neuronales artificiales normalmente son entrenadas off-line. Durante este proceso sus

parámetros internos son ajustados para reducir el MSE entre la salida real de la red y los datos de salida del conjunto de datos de entrenamiento. A partir de ese momento, se presentan nuevos datos de entrada-salida a la red y sus parámetros son ligeramente actualizados para mantener el MSE bajo con el paso del tiempo. Este último proceso es llamado normalmente adaptación o aprendizaje on-line

En [111] se comprueba la robustez del método frente a cambios en la masa y una perturbación externa de tipo escalón en el control de la altitud. Más adelante se comprueba la validez del método para el control simultáneo de los ángulos de Euler y la altitud. Además la perturbación escalón es sustituida por un modelo más realista de una perturbación producida por el viento. Los resultados se publicaron en [113]. En estos artículos no se plantea una estrategia específica anti-perturbaciones, se plantea una estrategia de neuro control adaptativo y se comprueba su robustez frente a perturbaciones y variaciones en el modelo. En la Sección 4.4 se presentará el método desarrollado durante la Tesis para el rechazo específico de perturbaciones.

4.2.1 Resultados

En la Figura 4.4 se representan algunos resultados de la aplicación de este método en el control de altitud y control de los ángulos de Euler. En la parte izquierda de la figura se muestra el resultado del control de altitud (línea azul), la línea verde es la referencia. La parte derecha recoge un ejemplo de control de ángulos de Euler, donde la línea roja es el roll, la amarilla es el pitch, la morada es el yaw, y la azul la referencia.

En ambos casos los controladores son entrenados off-line desde t=0 s hasta t=2 s. Por esta razón la referencia es cero en ese intervalo. Las señales de control usadas para entrenar la red producen cambios en la altitud y en la orientación durante este intervalo que se reflejan en la figura. La fase de control comienza en t=2 s; en este instante la referencia de altitud se fija a 5 m y los ángulos de Euler a $\pi/4$ radianes. En las figuras se puede observar cómo el control es capaz de estabilizar las señales en torno a la señal de referencia.

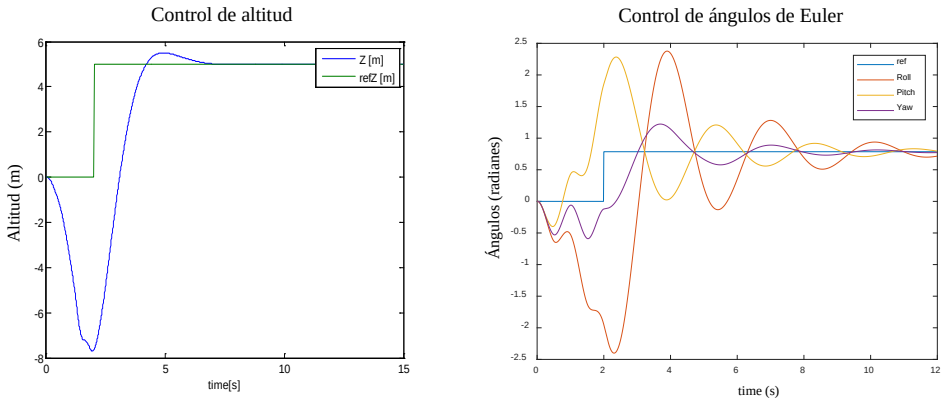

Fig. 4.4. Control de altitud (izquierda) y control de ángulos de Euler (derecha)

En otro experimento se comprobó la robustez de los controladores frente a las variaciones en la masa del sistema. El rendimiento del controlador de altitud con y sin aprendizaje on-line es comparado en la Tabla 4.1. Se aprecia como el sobreimpulso y error estacionario se reducen claramente con el aprendizaje on-line. El tiempo de establecimiento también se reduce ligeramente. Como era de esperar, el aprendizaje hace un poco más lenta la respuesta. La respuesta ofrecida por el controlador propuesto es significativamente mejor que la proporcionada por el PID.

Tabla 4.1 Rendimiento de los controladores cuando la masa varia (control de altitud)

Parámetro	Sin on-line learning	Con on-line learning	Solo con PID	Wang(2014)[102]
Tiempo de subida (s)	**2.05**	2.24	2.27	1
Tiempo de establecimiento (s)	9.42	**7.44**	8.56	8
Sobreimpulso (%)	9.66	**2.26**	34.04	10
Error estacionario	5.31	**0.015**	**0.015**	0.02

En la Tabla 4.1 también se han incluido los datos del experimento de Wang [102]. Estos datos deben comparase con cautela puesto que en su artículo se emplea un modelo de UAV diferente y un incremento de masa diferente.

4.3 MODELOS DE CAMBIO DE MASA E INFLUENCIA DEL VIENTO

En esta Tesis la variación de la carga y la influencia del viento fueron consideradas perturbaciones durante el diseño del controlador. La variación de la carga se simula mediante la adición del término $dist_m$ en las ecuaciones 3.10 a 3.12, resultando las ecuaciones 4.3 a 4.5.

$$\ddot{X} = -(sin\theta cos\phi)\,(b/(m * dist_m))u_1 \qquad (4.3)$$

$$\ddot{Y} = (sin\phi)\,(b/(m * dist_m))u_1 \quad (4.4)$$

$$\ddot{Z} = -g + (cos\theta cos\phi)\,(b/(m * dist_m))u_1 \quad (4.5)$$

Dos diferentes variaciones de carga se han tenido en cuenta: un perfil escalón (4.6), y un perfil de variación sinusoidal (4.7).

$$dist_m(t) = 2 + step(t-4) \quad (4.6)$$

$$m(t) = m_{model}\left(step(t) - step(t-4)\right) + step(t-4)\left(m_{model} + \frac{m_{model}}{2}\sin(\frac{2\pi}{8}t)\right) \quad (4.7)$$

Para modelar la influencia del viento se añaden nuevos términos a la aceleración del sistema, $dist_wX$, $dist_wY$, $dist_wZ$, dados por las ecuaciones 4.9 a 4.11. El viento externo se asume en la dirección del movimiento [114]:

$$v_w(Z) = v_{w(Z=20)} \cdot \frac{\log(Z/C)}{\log(20/C)} \quad (4.8)$$

$$dist_wX = sgn(v_w(Z)) \cdot \rho_{air} \cdot A_x \cdot Cd \cdot \left(\dot{X} - 2 * v_w(Z)\right)^2/(2m) \quad (4.9)$$

$$dist_wY = sgn(v_w(Z)) \cdot \rho_{air} \cdot A_y \cdot Cd \cdot \left(\dot{Y} - 2 * v_w(Z)\right)^2/(2m) \quad (4.10)$$

$$dist_wZ = sgn(v_w(Z)) \cdot \rho_{air} \cdot A_z \cdot Cd \cdot \left(\dot{Z} - v_w(Z)\right)^2/(2m) \quad (4.11)$$

Donde $v_{w(Z=20)}$ es la velocidad del viento a 20 m de altitud en m/s, $v_w(Z)$ es la velocidad del viento a la altitud Z en m/s, C es una constante relacionada con la fase de vuelo (durante los experimentos se fija su valor a 1.5), ρ_{air} es la densidad del aire en Kg·m3, A_x, A_y y A_z son las áreas efectivas del cuatrirrotor expuestas a cada componente del viento en m2, Cd es el coeficiente de arrastre respecto del viento, $\dot{X}$, $\dot{Y}$, $\dot{Z}$ son las velocidades en los ejes X, Y y Z en m/s, y sgn denota la función signo.

La velocidad del viento es simulada por un escalón con ruido blanco gaussiano en t = 4 s. El SNR entre la media del viento y el ruido es 10 dB. La media de la velocidad del viento es 12 m/s en el eje Z y 24 m/s en los ejes X e Y. Estos valores corresponden con los números 6 y 9 en la escala de Beaufort (brisa fuerte y viento fuerte) [115].

La Figura 4.5 muestra un ejemplo de la perturbación del viento en el eje x (izquierda) y eje y (derecha). La perturbación simulada se representa en azul y la perturbación estimada por el controlador en rojo.

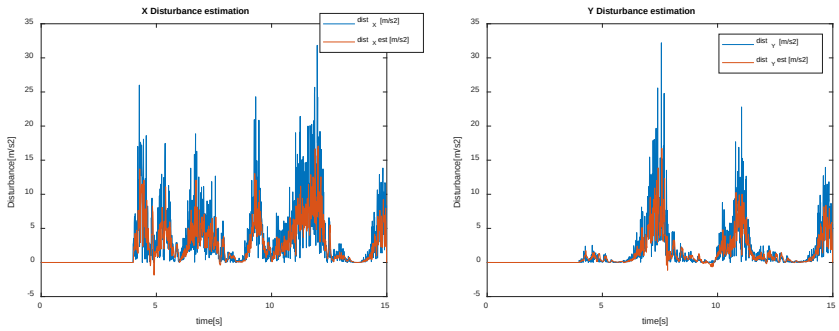

Fig. 4.5 Estimación de la perturbación del viento en el eje X e Y

4.4 MÉTODO PROPUESTO: CONTROLADOR BASADO EN NEURO ESTIMADORES ADAPTATIVOS

Durante el desarrollo de esta Tesis, como una evolución al controlador neural presentado en la Sección 4.2, se propuso el diseño de una estrategia de control anti-perturbaciones basada en redes neuronales para hacer frente a perturbaciones externa como cambios en la carga y viento, que alteran de forma relevante el comportamiento del cuatrirrotor. Los resultados se publicaron en [109]. El objetivo final es mejorar el seguimiento de trayectorias del UAV, así como el rendimiento del sistema frente a perturbaciones. La estrategia de control empleada se enmarca en la categoría del control basado en observadores de perturbaciones (DOBC), citada en la Sección 4.1. En nuestro caso se emplean observadores neuro-adaptativos para estimar la carga y la perturbación del viento.

El esquema de control de trayectorias para hacer frente a las variaciones en la carga y a las perturbaciones del viento propuesto se muestra en la Figura 4.6. Las entradas de control del UAV son las cuatro señales de control, u_i, $i = 1,..,4$, las cuales representan la potencia de los rotores. Se diseñan cuatro controladores principales para obtener estas entradas del modelo, los de las coordenadas X, Y, y Z, y otro para el ángulo de Yaw. Esto se debe a que el pitch y el roll se emplean para seguir las coordenadas de referencia $Xref$ e $Yref$. El control de Z se lleva a cabo mediante la señal de control u_1. El controlador de Y genera la referencia para el $Roll$, y el seguimiento del $Roll$ se realiza mediante la señal de control u_2. De la misma forma, el controlador de X genera los valores de referencia del ángulo de $Pitch$, y la señal de control u_3 estará a cargo de conseguir este valor. Por otro lado, la señal de control u_4 se usa para estabilizar el ángulo de Yaw en torno a cero. El objetivo de los PIDs de la Figura 4.6 es generar las referencias de aceleración que hagan que el error de orientación $(\phi_r - \phi, \theta_r - \theta, \psi_r - \psi)$ y el error de seguimiento $(X_r - X, Y_r - Y, Z_r - Z)$ converjan a cero. El resto de los controladores se emplean para compensar las no-linealidades del sistema. Los estimadores neuro adaptativos

alimentan las entradas de los controladores de X, Y y Z para compensar la influencia de la carga y el viento.

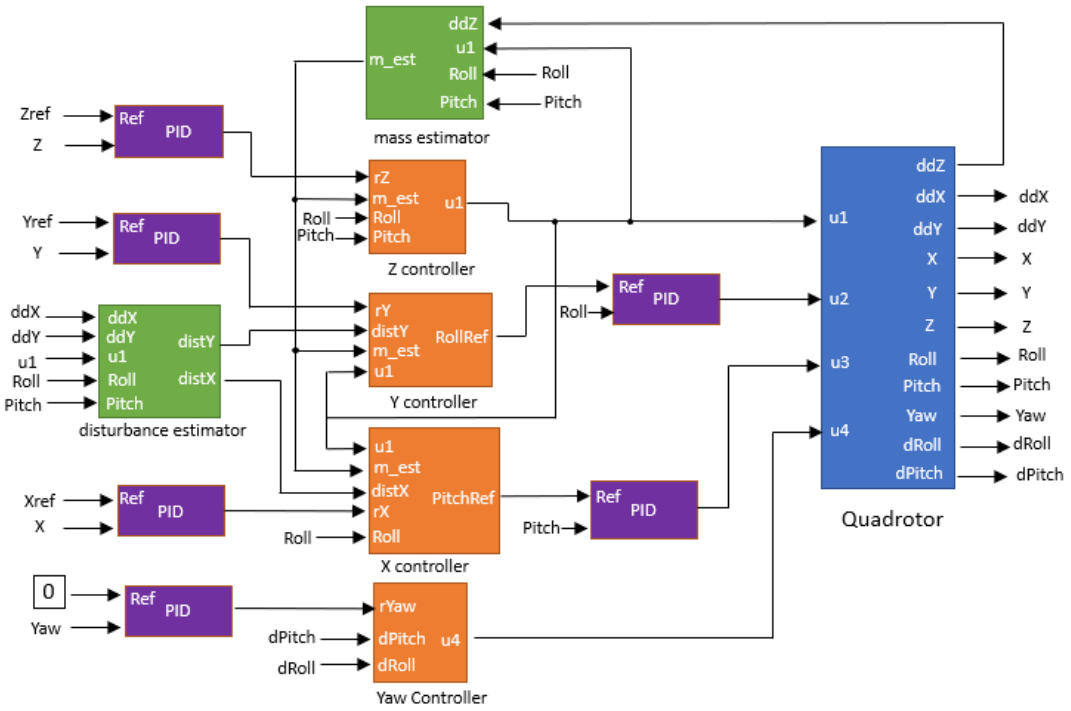

Fig. 4.6.. Sistema de control con estimadores de masa y perturbaciones

Aunque la estimación de la masa (m_{est}), la perturbación en el eje x ($distX_{est}$) y la perturbación en el eje y ($distY_{est}$) se hayan implementado en diferentes bloques, la estructura de todos ellos es la misma. Las diferencias provienen de los parámetros de configuración y de las señales de entrada y salida. Por lo tanto, pueden ser explicadas conjuntamente.

El estimador se basa en una red neuronal artificial con aprendizaje on-line. De ahí que haya una red neuronal específica para modelar cada una de las estimaciones consideradas. La Figura 4.7 representa su estructura genérica.

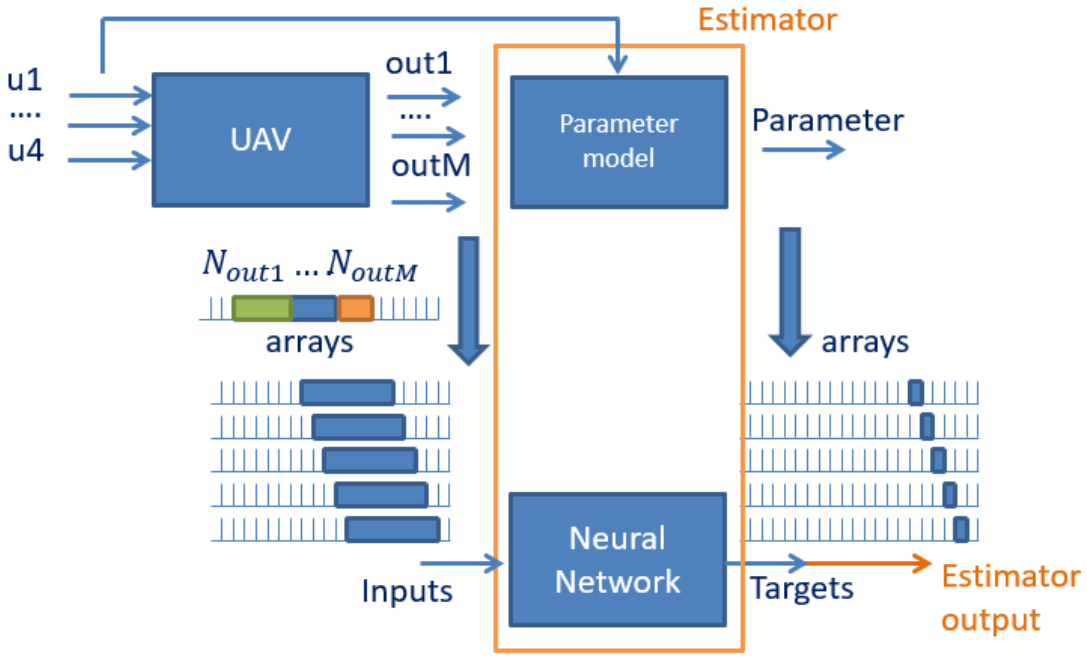

Fig. 4.7.. Estructura y configuración del estimador neuronal

El modelo paramétrico recibe las entradas y salidas del UAV. La salida del modelo se emplea como salida objetivo de la red neuronal durante el entrenamiento. La salida del estimador propuesto es siempre la salida de la red. El modelo paramétrico se necesita ya que estamos empleando redes neuronales artificiales supervisadas y, en éstas, las entradas y sus correspondientes salidas deben ser conocidas para poder realizar el entrenamiento.

Se emplea un modelo paramétrico para la masa, otro para la perturbación del viento en el eje X y otro para la perturbación del viento en el eje Y.

Cada elemento del conjunto de datos de entrenamiento está compuesto por:

- Objetivo: Valor del parámetro (masa o perturbación) a estimar calculado con el modelo en el instante t_i
- Entradas de la red: Para cada salida i, los N_i valores anteriores a t_i son recogidos y puestos en forma de vector.

Un ejemplo teórico sería el siguiente: En t_i=10 y para M=3 salidas con una configuración Nout1=3, Nout2=2, Nout3=1:

- Objetivo: param(10)
- Entradas: [out1(7), out1(8), out1(9), out2(8), out2(9), out3(9)]

Si el parámetro no puede ser calculado (división por cero, raíz cuadrada de un número negativo u otra singularidad) ese elemento no es incluido en el conjunto de datos de entrenamiento. En el caso de que no sea posible el cálculo del parámetro, la red neuronal es especialmente útil para estimar las perturbaciones. En estas situaciones, se puede decir que la red genera nuevo conocimiento.

La configuración específica de cada estimador empleado en la Figura 4.7, es la siguiente:

Estimación de masa:

- Salida esperada de la red: La masa es aproximada con el modelo

$$\widetilde{m}(t_i) = \frac{\ddot{Z}(t_{i-1})+g}{u_1(t_{i-1})cos\theta_{i-1}cos\phi_{i-1}} \qquad (4.12)$$

- Entradas de la red:
  - Aceleración en el eje Z, $\ddot{Z}(t_{i-1})$
  - Coseno de roll y pitch multiplicados ($cos\theta_{i-1}cos\phi_{i-1}$)

Estimación de perturbación en el eje X:

- Salida esperada de la red: La perturbación es aproximada con el modelo:

$$\widetilde{distX}(t_i) = -(sin\theta_{i-1}cos\phi_{i-1})\frac{bu_1(t_{i-1})}{m_{est}(t_{i-1})} - \ddot{X}(t_{i-1}) \qquad (4.13)$$

- Entradas de la red:
  - Aceleración en el eje X: $\ddot{X}(t_{i-1})$
  - Ángulo de roll: θ_{i-1}
  - Ángulo de pitch: ϕ_{i-1}

Estimación de perturbación en el eje Y:

- Salida esperada de la red: La perturbación es aproximada con el modelo:

$$\widetilde{distY}(t_i) = sin\phi_{i-1}\frac{bu_1(t_{i-1})}{m_{est}(t_{i-1})} - \ddot{Y}(t_{i-1}) \qquad (4.14)$$

- Entradas de la red:
  - Aceleraciones en el eje Y: $\ddot{Y}(t_{i-1})$

- Ángulo de roll: θ_{i-1}
- Ángulo de pitch: ϕ_{i-1}

4.4.1 Resultados

La Figura 4.8 muestra, a la izquierda, la trayectoria de referencia en azul, la trayectoria obtenida con el controlador con estimadores neuro-adaptativos en rojo, y sin ellos en amarillo (solo PID). A la derecha se presenta el error de seguimiento (en rojo con estimación adaptativa y en azul sin ella). En esta figura se estudia la robustez frente al efecto del viento.

Es posible observar como los resultados con estimadores neuro adaptativos son mucho mejores. También puede verse como el error de seguimiento es prácticamente el mismo hasta t = 4 s; en ese instante se introduce el efecto del viento, y el error sin los estimadores neuro adaptativos se incrementa significativamente.

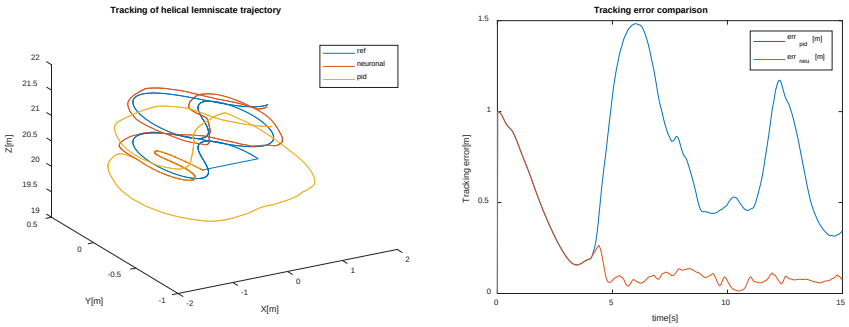

Fig. 4.8. Trayectoria lemniscata helicoidal con perturbación de viento (izquierda) y su error de seguimiento (derecha)

El controlador ha sido extensivamente probado mediante simulación para diferentes trayectorias con cargas y perturbación de viento. Los resultados numéricos (MSE) para el caso del viento se resumen en la Tabla 4.2 para las diferentes trayectorias y para cada coordenada, con el estimador neuronal (neuro) y sin él (PID). Las últimas columnas muestran el error absoluto. Los mejores resultados para cada componente y trayectoria están marcados es negrita.

Tabla 4.2: Comparación del MSE del error de seguimiento con perturbaciones de viento.

Trayectoria	MSE_X		MSE_Y		MSE_Z		MSE_T	
	Neuro	PID	Neuro	PID	Neuro	PID	Neuro	PID
Lineal	0.0707	0.0707	0.0708	0.0708	**1.6163**	2.1214	**0.6730**	1.0130
Circular	**0.2805**	0.8612	**0.0739**	0.2596	**1.6186**	2.2711	**0.7591**	1.6014
Helicoidal	**0.0951**	0.4850	**0.0042**	0.2009	**0.0048**	0.2847	**0.2181**	0.8666
Helicoidal Cíclica	**0.1053**	0.3435	**0.0132**	0.2000	**0.0069**	0.2596	**0.2746**	0.8084
Lemniscata	**0.0911**	0.1928	**0.0011**	0.0335	**0.0005**	0.3036	**0.1748**	0.6008
Lemniscata Helicoidal	**0.0940**	0.2614	**0.0032**	0.0695	**0.0010**	0.2460	**0.2015**	0.6903

En la Tabla 4.2 es posible observar como el controlador con estimadores neuronales adaptativos proporciona igual o menor error de seguimiento para todas las trayectorias aplicadas. El máximo error de seguimiento se obtiene con la trayectoria circular debido al alto error de las condiciones iniciales.

5 Conclusiones y Trabajos Futuros

5.1 CONCLUSIONES

- En esta Tesis se han generado diferentes modelos de vehículos cuatrirrotores aéreos no tripulados (UAVs) aplicando métodos paramétricos, basados en técnicas del soft computing, y con hibridación de técnicas. Comparando los resultados, se ha probado que las estrategias híbridas que combinan técnicas paramétricas y del soft computing proporcionan modelos cuya relación precisión-complejidad es mejor que las técnicas aplicadas de forma individual. Para llevar a cabo la comparación entre los modelos se generaron indicadores específicos de error en los dominios del tiempo y de la frecuencia. La complejidad computacional fue analizada a través de la descomposición de las técnicas individuales en sus operaciones elementales.

- Además, se ha analizado cómo el conjunto de datos de entrenamiento influye en el modelo obtenido cuando se aplican redes neuronales con aprendizaje supervisado. De hecho, las redes neuronales con aprendizaje off-line tienen la desventaja de ser muy sensibles a la partición del conjunto de datos de entrenamiento. Por el contrario, se ha observado que las redes neuronales adaptativas con aprendizaje on-line son mucho más robustas ya que están aprendiendo continuamente de los datos reales del sistema. Por tanto, no es necesario seleccionar ningún conjunto de datos para entrenar la red y no aparece sesgo derivado del entrenamiento.

- En una primera aproximación al diseño de controladores anti-perturbaciones se propuso un neuro-controlador adaptativo. Para ello, se aplicó una red neuronal MLP con un algoritmo de aprendizaje generalizado modificado al que se le añadió aprendizaje on-line. Los resultados de simulación validaron la efectividad de este controlador, incluso con perturbaciones como la variación de la masa y el viento. Además, el aprendizaje on-line de la red mejoró la robustez del controlador, reduciendo los efectos de las perturbaciones en la respuesta del sistema.

- Por último, se diseñó una estrategia de control anti-perturbaciones basada en neuro estimadores adaptativos. En la arquitectura propuesta las ecuaciones analíticas del modelo se introducen en el esquema de control y se diseñan observadores neuro-adaptativos

para estimar la masa y las perturbaciones. En este caso, las redes neuronales artificiales se centran en los términos que varían: la masa, la perturbación producida por el viento y, en general, las dinámicas no modeladas. Esta estrategia de control ha sido probada intensivamente mediante simulación con diferentes trayectorias: lineal, helicoidal, helicoidal circular, incluso con una lemniscata. En efecto, se observa como los estimadores neuro-adaptativos incrementan la robustez del controlador reduciendo los efectos producidos por estas variaciones.

5.2 TRABAJOS FUTUROS

Existen varias posibles líneas de trabajo futuras. Primeramente, las técnicas de modelado y control expuestas pueden aplicarse a otros tipos de UAVs, como el helicóptero de rotor único, el UAV de alas fijas, el VTOL, etc. Todos ellos podrían beneficiarse de los resultados de esta investigación. Mas aún, el campo de aplicación podría ser incluso extendido a otras áreas ya que las técnicas de modelado y control bajo estudio son de aplicabilidad general.

La técnica híbrida de modelado propuesta proporciona una excelente relación precisión-complejidad. Sin embargo, esta relación no es constante; depende de la configuración de la red neuronal (número de capas ocultas, número de neuronas, etc). En este sentido se podría configurar un algoritmo de optimización que considere el MSE y las ecuaciones analizadas durante la Tesis sobre la complejidad de la red. Por otro lado, la técnica de modelado propuesta emplea la conexión en cascada, por lo que también podrían explorarse otras topologías diferentes para diseñar nuevos modelos híbridos.

Respecto al control del cuatrirrotor, una posible línea futura de trabajo es analizar la complejidad computacional del esquema de control con los estimadores neuronales adaptativos para establecer los requerimientos mínimos de hardware. El número total de operaciones elementales se incrementará conforme lo haga la frecuencia de control y la frecuencia de aprendizaje de las redes. Un tema interesante es examinar la relación entre estas frecuencias y el rendimiento del controlador, lo que ayudará a establecer un compromiso entre el coste del hardware y el rendimiento.

Otra posible línea de trabajo futura es extender el rango de las perturbaciones que han sido consideradas. En este sentido, el control de cuatrirrotores con cargas suspendidas por un cable es actualmente un reto. En este problema aparecen nuevas componentes no consideradas: el centro de gravedad del sistema completo cambiará cada vez que la posición relativa de la

carga respecto del cuatrirrotor se desplace; el UAV está sujeto a fuerzas externas cambiantes producidas por la tensión del cable, etc.

Finalmente, otra línea extensa de investigación podría ser estudiar la mejor manera de introducir el concepto de los estimadores neuronales adaptativos en otras estrategias de control tales como LQR, control deslizante, etc, para mejorar sus resultados.

II Summary of the Research

1 Introduction

This chapter outlines: some general ideas about UAVs (Unmanned Aerial Vehicles) (Section 1.1) and soft computing (Section 1.2), the motivation of the research (Section 1.3), its aims and scope (Section 1.4), and the main contributions (Section 1.5).

1.1 UNMANNED AERIAL VEHICLES (UAVS)

The concept of UAV is applicable to any vehicle which is flying in the air without any driver on-board and with certain autonomy of movements [1]. But it is usually found with different names like Remotely Piloted Vehicle (RPV), Remotely Piloted Aircraft (RPA), Remotely Operated Aircraft (ROA), Remote Controlled Helicopter (RC-Helicopter), Unmanned Aircraft System (UAS) and Unmanned Vehicle System (UVS). They are defined by the Department of Defence (DOD) of USA as powered, aerial vehicles that do not carry a human operator, use aerodynamic forces to provide vehicle lift, can fly autonomously or be piloted remotely, can be expendable or recoverable, and can carry a lethal or nonlethal payload [2].

UAVs provide many civil and military applications. For example, they are used in agriculture for crop monitoring; in the environment sector for fire detection; in security and civil protection for surveillance; in architecture and construction for photography and structure inspections; in inspection operations of power lines and windmills, etc. One of the most recent applications is the traffic surveillance in roads and highways.

These applications demand the design of efficient and robust controllers. However, the modelling and the control of UAVs is not an easy task. Their complexity comes from the randomness of the airstreams and the external forces, the non-linearity of the dynamics, the coupling between the internal variables, the uncertainty in the measurements, etc. These factors make the techniques based in artificial intelligence a promising approach for the identification and the control of these systems. That is why the modelling and control of these complex and unstable systems still motivate the research and the interest of the scientific community [3-10].

There exists a wide range of types of UAVs and it will be shown in this section [11].

Multi-rotor

Normally they are the cheapest way to get smart and quality aerial videos and landscape photos. The downside of multi-rotors is their limited endurance and speed, making them unsuitable for large scale aerial mapping, long endurance monitoring and long-distance inspection such as pipelines, roads and power lines. The most used model is the quadrotor, but there are other models with 6, 8, 10, 12 and even with more rotors. The Figure 1.1 shows an example of 8 blades multi-rotor [12].

Fig. 1.1. A multirotor UAV [12]

Fixed-Wing

Fixed-wing UAVs use standard wings like we are used to see in airplanes to provide the lift. For this reason, they can cover longer distances, map much larger areas, and loiter for long times monitoring their point of interest. In addition to the greater efficiency, it is also possible to use gas engines as their power source, and with the greater energy density of fuel many fixed-wing UAVs can stay aloft for 16 hours or more. The main disadvantage of a fixed-wing aircraft is obviously their inability to hover in one spot, which rules them out for any general aerial photography work. This also makes launching and landing them a lot trickier, as depending on their size you can need a runway or catapult launcher to get them into the air, and either a runway, parachute or net to recover them safely again at the end. One example of this type is shown in the Figure 1.2 [13].

Fig. 1.2. A fixed-wing UAV [13]

Single-Rotor Helicopter:

A single rotor helicopter UAV has only one rotor on the top plus a tail rotor used to control the heading. One example is shown in the Figure 1.3 [14]

Fig. 1.3. A single-rotor helicopter [14]

This type is more efficient than the multirotor, besides they normally can be powered by a gas motor for longer endurance. They are very useful to transport heavy payloads such as a LIDAR. The main defects are: complexity, cost, vibration and higher hazard level due to the blades.

Fixed-Wing Hybrid VTOL (Vertical Take-off and Landing)

New models of these types of UAVs are being designed nowadays. For example, the Figure 1.4 shows a VTOL recently designed by the NASA [15]. The main feature is the vertical take-off and landing (VTOL) capability. This fact allows to reduce the take-off/landing site requirements at minimum.

Fig. 1.4. A fixed-wing hybrid VTOL UAV [15]

The most know UAV is the quadrotor; for this reason the Thesis is focused on this model. Furthermore the system is enough complex to test the new proposed modelling and control techniques.

1.2 SOFT COMPUTING

The term Soft Computing was proposed by the inventor of fuzzy logic, Lotfi A. Zadeh:

"Soft computing is a collection of methodologies that aim to exploit the tolerance for imprecision and uncertainty to achieve tractability, robustness, and low solution cost. Its principal constituents are fuzzy logic, neurocomputing, and probabilistic reasoning….. The role model for soft computing is the human mind" [16].

Soft computing is not precisely defined. It consists of distinct concepts and techniques which aim to overcome the difficulties encountered in real world problems. These problems result from the fact that our world seems to be imprecise, uncertain and difficult to categorize. For example, the uncertainty in a measured quantity is due to inherent variations in the measurement process itself [17].

The Soft computing also may be the base of the Computational Intelligence field (CI). The difference between the traditional Artificial Intelligence (AI) and the Intelligence Computational is that the AI is based on hard computing and CI is based on soft computing [17].

During the development of this Thesis, artificial neural networks have been used in the process of modelling and in the design of control strategies, mainly exploring their on-line learning ability. The fuzzy logic has also been used during the modelling by the ANFIS technique (for more details see Section 2.2.3).

1.3 MOTIVATION AND GENERAL PROBLEM FORMULATION

In recent years, new and valuable applications of unmanned aerial vehicles (UAV) have emerged in different sectors such as defence, security, construction, agriculture, entertainment, shipping, etc. [18-21]. These and other applications demand the design of efficient and robust controllers for those autonomous vehicles.

By the other hand, one important step in order to know and effectively control a physical system is to find out a reliable and useful mathematical model that describes its behaviour. The importance of having a good model grows with the complexity of the system. There exist many techniques and methods to approach this target; they may be classified into two main groups. The first one is based on applying the well-known physical equations that govern the dynamic behaviour and using this knowledge to generate the differential equations that represent it. This methodology is straightforward when all variables are known, and the relations between them are simple. But in most cases, it is necessary to deal with equations too complex or even impossible to obtain. In this case, it is necessary to apply some more advanced methods such as the systems identification. This other main group of techniques is based on designing experiments for efficiently generating informative data; and then applying different mapping mechanisms to describe the hidden internal relationships between the inputs and the outputs. These mapping processes iteratively update their parameters for fitting the model as well as reducing its order [22]. In this last category it may be found, among others, parametric methods and intelligent techniques provided by soft computing discipline, such as neuronal and neuro-fuzzy networks and genetic algorithms [23]. These methods are proper when dealing with strongly non-linear and complex systems such as we face in this work.

Due to the learning ability, these intelligent strategies are especially interesting when the model's parameters vary while the system is working. For example, the total mass will undergo variations when the vehicle is performing logistic tasks, since the total mass depends on the loads that are shipped. Also, they are particularly useful when the system is subject to high external disturbances such as the wind, as in our case.

For these reasons, this Thesis propose the use of artificial neural networks and fuzzy logic to improve the current modelling and control techniques for quadrotor UAVs. In this line other previous studies have been carried out, but there are still points to be deeper studied, as for example: the reduction of the computational complexity of the UAVs models thanks to the hybridization of the modelling techniques; or the benefits of the on-line learning to reject disturbances. These topics and other have been considered in this work, which could contribute to meaningful improvements in the modelling and control of quadrotors.

1.4 THESIS OBJECTIVES

The general objective of this thesis is to develop advanced modelling methods and apply innovative control strategies to perform efficient adaptive intelligent control of UAVs able to adapt to changes in the system and the environment, and to reject disturbances. Given the wide range of UAVs to be considered, this thesis focuses on the quadrotor type. Nevertheless, many of the results in this Thesis could be generalized to other UAV models.

The following specific targets are addressed in this work:

- Study of the state of the art about the modelling and control of quadrotors, paying special attention to the previous works based on intelligent techniques.

Modelling:

- Analysis of the viability of the hybridization of techniques to improve the features of the models.
- Comparison of the performance of the different mathematical models, considering the error in the time and the frequency domain, as well as their computational complexity.
- Study of the influence of the data partition in the performance of the models obtained by neural networks.

Control:

- Proposition and validation by simulation of a strategy of UAVs control based on artificial neural networks (neuro-control)

- Providing learning capabilities to the designed control strategy (adaptive neuro-control), allowing it to react to changes in the environment and model parameters.
- Study the capability of the neural networks to be integrated in an anti-disturbance strategy to improve the capacity to reject disturbances, such as the wind.

1.5 MAIN CONTRIBUTIONS

The contributions of this Thesis are summarized in the main conclusions of this work and can be organized in the following developments and publications.

1.5.1 Contributions

The results of this Thesis include the development of modelling and control algorithms, implemented in MATLAB [24]. The most important contributions are summarized below:

- Quadrotor models based on neural networks
- Hybridization of parametric techniques and soft-computing techniques
- Adaptive neuro-control strategy based in inverse plant model
- Adaptive neuro estimators for the mass and the wind disturbance
- Adaptive tracking control based on the adaptive neuro estimators of the previous point.

1.5.2 Publications

During the PhD Thesis several articles have been published in specialized journals, and contributions in national and international conferences. Most of the papers have been obtained as direct result of this Thesis, being the doctoral candidate the first author in all of them.

Indexed Journal Papers Published:

- Sierra. J.E., Santos, M. (2018). Modelling engineering systems using analytical and neural techniques: hybridization. Neurocomputing, 271, 70-83 **(Q1)**
- Sierra. J.E., Santos, M. (2019). Wind and payload disturbance rejection control based on adaptive neural estimators: application on quadrotors. Complexity, Article ID 6460156, 20 pages **(Q2)**

Papers in Conferences:

- Sierra, J.E., Santos, M. (2013) Estudio comparativo de modelos de un vehículo aéreo obtenidos mediante técnicas analíticas y basadas en redes neuronales. Actas Multiconferencia CAEPIA'13 (XV Conferencia de la Asociación Española para la Inteligencia Artificial), 1270-1279. ISBN: 978-84-695-8348-7

- Sierra, J.E., Santos, M. (2013) Modelado de un vehículo aéreo no tripulado mediante aplicación conjunta de técnicas paramétricas y neuronales. Actas de las XXXIV Jornadas de Automática, 189-194, 2013. ISBN: 978-84-616-5063-7

- Sierra, J.E., Santos, M. (2014) Modelado de un vehículo aéreo no tripulado mediante combinación de técnicas paramétricas y neurodifusas. In Actas del XVII Español sobre Tecnologías y Lógica Fuzzy, 339-344. ISBN: 978-84-15688-76-1.

- Sierra, J. E., Santos, M. (2015). Adaptive neural control-oriented models of unmanned aerial vehicles. Advances in Intelligent Systems and Computing, (pp. 329-337). Springer International Publishing. (10th International Conference on Soft Computing Models in Industrial and Environmental Applications). ISBN 2194-5357.

- Sierra, J. E., Santos, M. (2017). Control de un vehículo cuatrirrotor basado en redes neuronales. Actas de las XXXVIII Jornadas de Automática, pp 431-436. ISBN: 978-84-16664-74-0.

- Sierra, J. E., Santos, M. (2019). Disturbances based adaptive neuro-control for UAVs: A first approach. Advances in Intelligent Systems and Computing (pp. 293-302). Springer, Cham (In The 13th International Conference on Soft Computing Models in Industrial and Environmental Applications). ISBN: 978-3-319-94119-6

2 Techniques and Materials

This chapter describes the materials used during the modelling phase (Section 2.1) and the different techniques analysed for the application to UAVs modelling and control (Section 2.2).

2.1 MATERIALS

Data provided by the Control Engineering Group of the Spanish Committee of Automation (CEA) [25] were used for the UAV modelling. The data were recorded from an unmanned aerial vehicle ARDRONE 2.0 by Parrot (Figure 2.1). This system is made of carbon fibre and plastic PA66. It is equipped with four high efficiency brushless motors, powered by one 11V 1000mAh lithium battery, which gives up to 12 minutes flight autonomy. It weighs 380 g without casing and 420 g with casing. It has a three-axis accelerometer, a two-axis gyroscope (pitch and roll), and a high precision gyroscope for the yaw angle [26].

Fig. 2.1. ARDRONE by Parrot [27]

The input/output signals are represented in the Figure 2.2. The pitch signal and velocity in the x-axis are shown in blue meanwhile the roll signal and the velocity in the y-axis are represented in green. As it may be seen in the figure, the input signals consist of a train of steps with different amplitude. This kind of signals has been traditionally used in identification to try to extract the maximum possible information from the system.

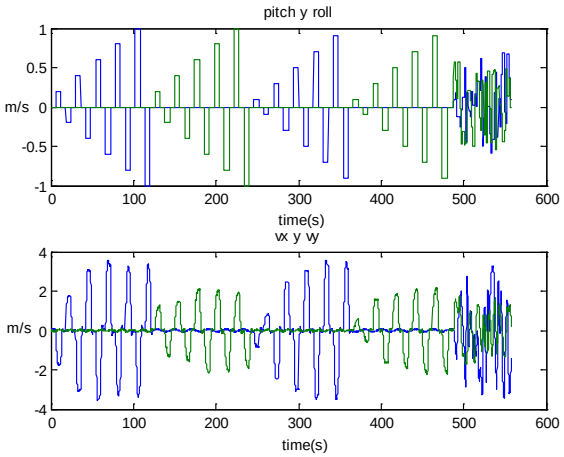

Fig. 2.2. Pitch and roll (inputs) and velocity (output)

In order to extract as much information as possible from the plant, the input signals used during the identification must have as much power as possible, in other words, a high order EP to excite as much system's frequencies as possible; its duration must be high, the higher the number of input data is the lower the variance of the estimated parameters will be; and they must not be correlated with the disturbances. By the other hand, it is also wanted the experiments to be plant-friendly, to do it, they must be short and produce a minimal disturbance in the controlled variables, this is, to introduce a low variance and low deviations from the target point.

Accordingly, to select an input signal a balance between friendliness and information extraction capability. The signals in Figure 2.2 present an adequate balance between these indicators to identify the system.

2.2 TECHNIQUES

2.2.1 PEM Models (Prediction Error Methods)

Let us consider an invariant linear system described by the following differential equation:

$$y(t) = G(\theta, q)u(t) + H(\theta, q)e(t) \tag{2.1}$$

Where $y(t)$ is the output, $u(t)$ is the input, $G(\theta, q)$ is the plant's transfer function and $H(\theta, q)$ is the perturbation transfer function, q is the delay operator and $e(t)$ is the one step prediction error. $G(\theta, q)$, and $H(\theta, q)$ is given by the expressions [28]:

$$G(\theta,q)=\sum_{k=1}^{\infty}g(\theta,k)q^{-k} \qquad H(\theta,q)=1-\sum_{k=1}^{\infty}h(\theta,k)q^{-k} \tag{2.2}$$

In order to obtain the step predictor, let us assume that $e(t)$ is white noise, therefore its variance is given by $Var(e)=\lambda^2(\theta)I$. Then the output $\hat{y}(\cdot,\theta)$ that minimizes$E(y-\hat{y})^T(y-\hat{y})$ has the form:

$$\hat{y}(\cdot,\theta)=L_1(q,\theta)y+L_2(q,\theta)u \tag{2.3}$$

Past inputs and outputs are mapped to give the new predicted output.

$$y-\hat{y}=Gu+He-L_2u-L_1y \tag{2.4}$$

Notice that $y=Gu+He$, manipulating the terms the equation becomes:

$$y=(I-H^{-1})y+H^{-1}Gu+e \tag{2.5}$$

Therefore,

$$y-\hat{y}=(I-H^{-1}-L_1)y+(H^{-1}G-L_2)u+e=z+e \tag{2.6}$$

Where $E(y-\hat{y})^T(y-\hat{y})$ must be minimized,

$$E(y-\hat{y})^T(y-\hat{y})=E(z^Tz)+E(e^Te)\geq\lambda^2(\theta)I \tag{2.7}$$

The lower bound is achieved when $z=0$, then from (2.3) L_1 and L_2 must be:

$$L_1=I-H^{-1} \qquad L_2=H^{-1}G \tag{2.8}$$

Finally, the one step predictor results,

$$\hat{y}(t,\theta)=\big(1-H^{-1}(q,\theta)\big)y(t)+H^{-1}(q,\theta)G(q,\theta)u(t) \tag{2.9}$$

And the one step prediction error is,

$$\varepsilon(t,\theta)=y-\hat{y}=H^{-1}(q,\theta)[y-G(q,\theta)]u \tag{2.10}$$

The basis of the parametric techniques is the minimum prediction error paradigm. Given $[y(t),u(t),t\leq N]$, find $\widehat{\theta}$ (the estimation of the parameters θ) which minimizes

$$Var(\varepsilon)=\frac{1}{N}\sum_{t=1}^{N}\|\varepsilon(t,\theta)\|^2 \tag{2.11}$$

The model defined in (2.1) is based on PEM (Prediction Error Method), $G(q)$ and $H(q)$ are rational functions that are given by the coefficients in the numerator and the denominator, which are the model's parameters.

In general, a PEM model may be expressed by the following equation:

$$y(t) = \frac{B(q)}{A(q)F(q)}u(t - nk) + \frac{C(q)}{A(q)D(q)}e(t) \tag{2.12}$$

Where $A(q), B(q), C(q), D(q)$ y $F(q)$ are polynomials:

$$\begin{aligned} A(q) &= 1 + a_1 q^{-1} + \cdots + a_{na} q^{-na} \\ B(q) &= b_1 q^{-1} + \cdots + b_{nb} q^{-nb} \\ C(q) &= 1 + c_1 q^{-1} + \cdots + c_{nc} q^{-nc} \\ D(q) &= 1 + d_1 q^{-1} + \cdots + d_{nd} q^{-nd} \\ F(q) &= 1 + f_1 q^{-1} + \cdots + f_{nf} q^{-nf} \end{aligned} \tag{2.13}$$

The structure of a PEM model is defined by the order of the polynomials (na, nb, nc, nd, nf) and the delay nk. Since there are five different polynomial functions, it is possible to find up to 32 different PEM structures. To carry out this study, two of them have been selected according their importance and simplicity: ARX and OE.

ARX (Autoregressive with Exogenous Input)

An ARX model is defined by the following differential equation [28]:

$$A(q, \theta)y(t) = B(q, \theta)u(t - nk) + e(t) \tag{2.14}$$

It can also be expressed as,

$$\begin{aligned} &y(t) + a_1 y(t-1) + \cdots + a_{n_a} y(t - n_a) \\ &= b_1 u(t - 1 - nk) + \cdots + b_{n_b} u(t - n_b - nk) + e(t) \end{aligned} \tag{2.15}$$

Considering the last equations, it is easy to realize that the vector of parameters in this case is $\theta = [a_1 \ \ldots \ a_{n_a} \ b_1 \ldots \ b_{n_b}]$

The acronym ARX comes from Autoregressive with Exogenous Input, because $A(q, \theta)y(t)$ is an autoregression and $B(q, \theta)u(t - nk)$ is the contribution of the exogenous input u. In this case, the problem of searching θ can be solved like a linear regression problem. The correspondence between the polynomials and the model defined in the equation (2.1) is:

$$G(q, \theta) = \frac{B(q, \theta)}{A(q, \theta)} \qquad H(q, \theta) = \frac{1}{A(q, \theta)} \tag{2.16}$$

ARMAX (Autoregressive Moving Average with Exogenous Input)

An ARMAX model is defined by the following differential equation [28]:

$$A(q,\theta)y(t) = B(q,\theta)u(t-nk) + C(q,\theta)e(t) \tag{2.17}$$

It can also be expressed as,

$$\begin{aligned} &y(t) + a_1 y(t-1) + \cdots + a_{n_a} y(t-n_a) \\ &= b_1 u(t-1-nk) + \cdots + b_{n_b} u(t-n_b-nk) + \\ &\quad c_1 e(t-1) + \cdots + c_{n_c} e(t-n_c) + e(t) \end{aligned} \tag{2.18}$$

This expression is equivalent to,

$$\begin{aligned} &y(t) + a_1 y(t-1) + \cdots + a_{n_a} y(t-n_a) \\ &= b_1 u(t-1-nk) + \cdots + b_{n_b} u(t-n_b-nk) + v(t) \end{aligned} \tag{2.19}$$

Where $v(t)$ is a mobile average of the error signal $e(t)$ and it is given by:

$$v(t) = c_1 e(t-1) + \cdots + c_{n_c} e(t-n_c) + e(t) \tag{2.20}$$

Taking into account the last equations, the vector of parameters in this case is $\theta = [a_1 \ \dots \ a_{n_a} \ b_1 \dots \ b_{n_b} \ c_1 \dots \ c_{n_c}]$.

The acronym ARMAX comes from Autoregressive Moving Average with Exogenous Input, because $A(q,\theta)y(t)$ is an autoregression and $B(q,\theta)u(t-nk)$ is the contribution of the exogenous input u, and $v(t)$ is the mobile average of the error signal. The correspondence between the polynomials and the model defined in (2.1) is:

$$G(q,\theta) = \frac{B(q,\theta)}{A(q,\theta)} \qquad H(q,\theta) = \frac{C(q,\theta)}{A(q,\theta)} \tag{2.21}$$

OE (Output Error)

The model is defined by the following differential equation [28]:

$$y(t) = \frac{B(q,\theta)}{F(q,\theta)} u(t-nk) + e(t) \tag{2.22}$$

Where

$$\begin{aligned} B(q) &= b_1 q^{-1} + \cdots + b_{nb} q^{-nb} \\ F(q) &= 1 + f_1 q^{-1} + \cdots + f_{nf} q^{-nf} \end{aligned} \tag{2.23}$$

The meaning of the acronym OE is Output Error. This name is explained by the fact that the noise source $e(t)$ is exactly the perturbation source, since $H(q,\theta) = 1$. In contrast to the last case, the problem of searching θ must be solved like a non-linear regression problem. The correspondence between the polynomials and the model defined in (2.1) is:

$$G(q,\theta) = \frac{B(q,\theta)}{F(q,\theta)} \qquad H(q,\theta) = 1 \tag{2.24}$$

2.2.2 Neural Networks Applied to System Identification

Neural networks are massively parallel distributed information processing systems, implemented in hardware or software form. Basic neural networks are composed by three elements: a set of highly interconnected identical o similar simple processing units, commonly called neurons, which perform local processing, and are arranged in an ordered topology; a learning algorithm to acquire knowledge from their environment; and finally, a recall algorithm to use the learned knowledge [29]. The topology of the net and the type of neuron determine the name of the neural network. In general, the networks may be classified into two different groups: supervised learning neural networks, and non-supervised leaning networks. For identification purposes, the most interesting ones are the supervised neutral networks. Focusing on this category, the following topologies may be highlighted, among others: MLP (Multilayer Perceptron), RBF (Radial Basis Function) and CMAC (Cerebellar Model Articulation Controller) [30].

Some of the main features of the neural networks are: complex nonlinear input-output mapping, adaptability, learning capacity, distributed architecture, fault tolerance, and neurobiological analogy. The first two properties are exploited in the identification process. As the real systems are causal, the value of the output in the actual instant of time depends on the values of the inputs in the same instant and on the values of the inputs and output in the previous instants to the actual time. This fact can be formalized by the following expression:

$$y(k) = f(u(k), u(k-1), \dots, u(k-N), y(k-1), \dots, y(k-M)\,) \tag{2.25}$$

Where k represents the actual instant of time, y is the output of the system, u is the input of the system and f is a nonlinear mapping function. The structure of the function f comes given by the topology of the net and the type of neurons. The behaviour of this function f is modulated by a set of weights, usually described by w. The learning process of the network modifies w, in such a way that the output of the model is more and more similar to the real output due to the application of an optimization rule. This way, when the learning finishes, the function f approximates the dynamic of the system.

The supervised learning algorithm is fed with a training dataset, where the mapping relation between the inputs and the output is established. For each target data (the value of the output signal), an input vector (regression vector) is assigned. The correct choice of learning algorithm is very important to reach an appropriate speed and even to ensure the convergence. Several classes of algorithms may be found: gradient methods, exhaustive search, random search and genetic search. Focusing on gradient methods, the most known methods are: steepest descent, Newton, Levenberg-Marquardt [31] and conjugate gradient.

2.2.2.1 Multi-layer Perceptron (MLP)

They are composed by several layers of neurons feedforward connected, each neuron of each layer is connected with every neuron of the following layer. This way, the graph of the network structure has not any cycle. In each layer one different activation function may be used. The connection between the neuron i and the neuron j (of two consecutive layers) is associated with a weighting coefficient or weight w_{ij}, and the neuron i has a threshold coefficient or bias b_i. The output of the neuron i comes given by the perceptron expression:

$$\mathrm{x_i} = \mathrm{f}(b_i + \sum_{j=1}^{N} w_{ij}\,\mathrm{x}_j) \tag{2.26}$$

Where f is the activation function and N is the number of neurons in the previous layer. Usually, a linear transfer function is used in the last layer. In the internal layers, also called hidden layers, the sigmoid function, the hyperbolic tangent, the RELU (Rectified Linear Unit) function, or the step are often used, depending on the problem to be solved. However, there are another possible activation functions.

2.2.2.2 Radial Basis Function Networks (RBF)

Radial Basis Function Networks have been used due to its well-known aptitude to approximate functions; this fact was proved by Park in 1991 [32]. The topology of the network is shown in the Figure 2.3.

This type of neural network has three layers: the input layer which receives the signals from the exterior, the hidden layer that is connected with the input layer and where each element is associated to a radial basis function, and the output layer where the results of the activation functions of the hidden layer are linearly weighted [33].

The output of each element of the hidden layer is calculated as the distance between the input pattern x(n) and the centre c_i of the cluster weighted by d_i. This value is applied to a radial basis function according to the following expression:

$$z_i(n) = \Phi\left(\frac{\sqrt{\sum_{j=1}^{N}(x_j(n) - c_{ij})^2}}{d_i}\right), i = 1, \dots M \tag{2.27}$$

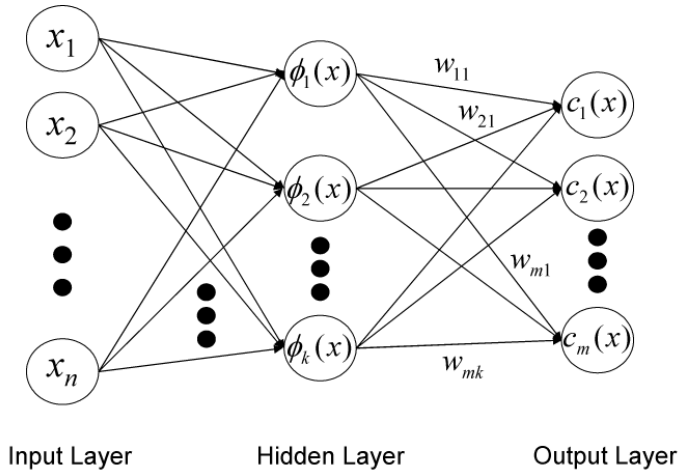

Fig. 2.3. Radial Basis Function Neural Network Topology

Where N is the number of elements of the input pattern x(n) and the basis radial function is defined by:

$$\Phi(n) = e^{-n^2} \tag{2.28}$$

Output values of the hidden layer are linearly weighted to give the final output of the network:

$$y_k(n) = \sum_{i=1}^{M} w_{i,k} z_i(n) + \mu_k, k = 1, \dots K \tag{2.29}$$

Where the number of neurons in the hidden layer is M, K is the number of nodes in the output layer, $w_{i,k}$ are the weights of the output layer, and μ_k are the activation thresholds.

2.2.2.3 Generalized Regression Networks

It is very similar to the Radial Basis Function Network, the difference is in the output layer, where the linear weighting of the expression (2.29) is substituted by a normalized product function [34].

$$y_k(n) = \frac{\sum_{i=1}^{M} w_{i,k} z_i(n)}{\sum_{i=1}^{M} z_i(n)}, k = 1, \dots K \qquad (2.30)$$

Where the number of neurons in the hidden layer is M, K is the number of nodes in the output layer and $w_{i,k}$ are the weights of the output layer.

2.2.3 ANFIS

ANFIS (Adaptive Neural Fuzzy Inference System) technique has been widely applied to identification [35]. The Figure 2.4 shows the structure of layers which composes the ANFIS topology.

It is based on the generation of a decision tree to classify data between p^n regression models, where the number of inputs is n and the number of partitions of each variable is p.

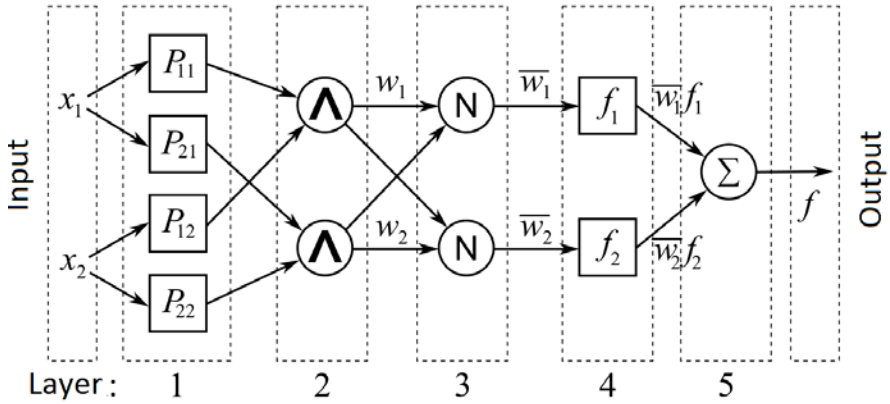

Fig. 2.4. Layer structure of ANFIS topology [25]

These networks are composed by 5 layers. In the input layer, also named value layer, the inputs are fuzzified. This layer is composed by p^n nodes, and the output of each node is given by:

$$O_{1,i} = \mu_{A_{i,j}}(x_k), i = 1, \dots p^n, j = 1, \dots p, k = 1, \dots n \qquad (2.31)$$

Where A is a linguistic label (small, big, …), and the function μ is given by:

$$\mu_A(x_k) = \frac{1}{1 + \left|\frac{x_k - c_i}{a_i}\right|^{2b}} \tag{2.32}$$

With the outputs of the nodes in the first layer, the fuzzy rules are fired in the second layer (rules layer), using some norm (minimal, product, …):

$$O_{2,i} = w_i = T(\mu_{A,j}, \mu_{B,j}, \ldots), i = 1, \ldots p^n, j = 1, \ldots p \tag{2.33}$$

In the layer 3 (normalization layer), the outputs from layer 2 are normalized:

$$O_{3,i} = \overline{w}_i = \frac{w_i}{\sum_{i=1}^{p^n} w_i}, i = 1, \ldots p^n \tag{2.34}$$

In layer 4 (function layer), the outputs from the layer 3 (w_i), weight the lineal regression function results from its node, making the output rules:

$$O_{4,i} = \overline{w}_i f_i = \overline{w}_i (p_{1,i}\, x_i + \cdots\ p_{n,i}\, x_n), i = 1, \ldots p^n \tag{2.35}$$

Where $p_k, k = 1, .. n$ are the consequence parameters. At the end, in the output layer (layer 5) a weighted sum with the outputs from layer 4:

$$O_{5,i} = \sum_{i=1}^{p^n} \overline{w}_i f_i = \frac{\sum_{i=1}^{p^n} w_i f_i}{\sum_{i=1}^{p^n} w_i} \tag{2.36}$$

3 Modelling of Quadrotors

This chapter starts with the description of the state of the art, citing the main contributions of different authors to the quadrotor modelling by analytic, neural and neuro-fuzzy techniques are cited (Section 3.1). Then an analytic model of the quadrotor is briefly explained (Section 3.2). In the sections 3.3 and 3.4 the procedure used to obtain neural and neurofuzzy models during the Thesis is shown. Finally, the new hybridization technique developed during the Thesis is presented (Section 3.5).

3.1 STATE OF THE ART

In the recent years the research and the development of unmanned aerial vehicles has received the attention of the scientists [3-10]. Focusing on the modelling, it is possible to find different strategies in the related papers, from analytic models based on the physical equations which represent the dynamical behaviour of the system, up to intelligent models based on soft computing techniques [36].

One of the first analytic models of the dynamic behaviour of one helicopter was proposed by Gessow and Myers in 1952 [37]. Since then other contributions have been carried out by other authors, starting from helicopters, such as the work of Bramwell in 1976 [38] and then covering whatever type of UAV. Analytic models of helicopter also appear in Cai [39], Budiyono [40] and El-Saadany [41]. Del Cerro [42] presents a hybrid model, analytic and empiric, where the parameters are calculated by genetic algorithms. One deep survey of mathematical models and control strategies of different types aerial vehicles appears in Castillo et al. [43].

Among these mathematical approaches, several authors obtain models of different UAVs by the application of PEM techniques. For example, Wu et al. [44] applies ARX to identify the model of one micro aerial vehicle in flight by using the measurements of the UAV attitude. Morris et al. [45] and Cai et al. [39] use PEM based on state space to identify either the model of a full vehicle or focusing on one subsystem, as for example the variation of the *yaw* angle. The model proposed by Mettler [46] is widely cited by other authors as a complex but effective model. Other parametric model of a helicopter is obtained by the application of a frequency domain, method developed by the USA Navy and the NASA, known as CIFER

(Comprehensive Identification from Frequency Responses) [47]. While CIFER was specifically developed for rotorcrafts, it has been successfully used in a wide range of fixed wing and rotary wing vehicles, and in unconventional aircraft applications [48].

Other authors also use classical parametric structures to identify the model but then apply advance strategies in order to tune the values of the model parameters. Some of them apply soft computing techniques for the tuning. For example, Ahmad et al. use the parametric model NARX (Nonlinear ARX) and they apply neural networks to adjust the parameters [49]. Another example is offered by del Cerro in [42] where the tuning is done by genetic algorithms.

Hashimoto et al. [50] compares the performance of the identification models of an unmanned helicopter obtained by ARX, ARMAX, OE and BJ parametric structures. In our study we have also applied ARX, ARMAX and OE but, unluckily, Hashimoto does not provide quantitative results; therefore, it has not been possible to compare our results with their models. Yuan [51] applies the OE method to estimate the model of an unmanned helicopter. Manai [52] uses OE structure for the identification of an UAV; in this case the purpose of the study is the design of controllers. Our paper is also aimed to design control-oriented UAV models, but in our work, we present a hybridization technique which can improve the accuracy while reducing the computational cost. Besides we have also defined some figures to quantitatively evaluate the performance of the models.

Some theoretical works have rigorously proved that, even with only one hidden layer, neural networks are universal function approximators [53]. Thus, they are often used to model non-linear systems like the UAVs.

Among the application of neural networks to UAVs modelling, MLP and RBF networks are the most used to simulate the pitch, roll, yaw, and the position of the unmanned vehicles [54-61]. A RBF network is also used in [62]. Neural networks based on Nonlinear Auto-Regressive models with eXogenous inputs (NARX models) have been also adopted due to its proven good performance especially for nonlinear complex systems [63]

In [64] three different methods are used to obtain the parameters of a quadrotor model: the Iterative Bi-Section Shooting method, IBSS; an artificial neural network, ANN; and a hybrid ANN-IBSS. In this paper, as in our case, the hybrid method provides higher accuracy than the rest of methods individually applied. Another hybridization technique is proposed by Jin in [65], in this case combining RBF and ARX. Puttige [66] approaches the modelling with a multi-network architecture by the combination of off-line and on-line methods.

Fuzzy logic has been also applied to the identification of UAVs. There are some works that could be highlighted, such as the one by Salman, who obtains the UAV model applying states space modelling and fuzzy logic identification and compares them [67]. Other examples of application of soft computing techniques (neuro-fuzzy) to model complex non-linear systems can be found [68-70].

3.2 ANALYTIC MODELS

A quadrotor vehicle is composed by four perpendicular arms, each one with a motor and a propeller (Figure 3.1). The four motors drive the lift and direction control.

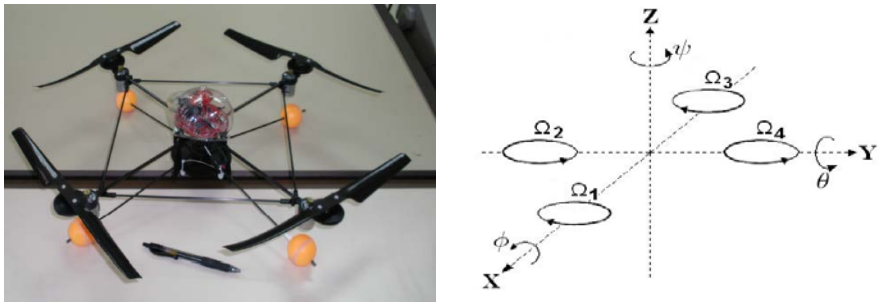

Fig. 3.1. A quadrotor vehicle (left) and UAV's coordinate system (right)

The UAV absolute position is described by three coordinates, (x, y, z), and the attitude is given by the three Euler's angles (ϕ, θ, ψ). Some conditions in the Euler's angles must be ensured to avoid an undesired effect, known as "gimbal lock", this is the loss of one degree of freedom when two rotation axes are aligned. These conditions are: $(-\pi \leq \psi < \pi)$ for the yaw, $(-\pi/2 \leq \phi < \pi/2)$ for the roll, and $(-\pi/2 \leq \theta < \pi/2)$ for the pitch, in radians. These limitations would not be needed in a model based on quaternions.

The system is based on two couples of propellers opposed each other, (1, 3) and (2, 4) (Figure 3.1, right). To keep the balance of the system, one pair of motors turns clockwise while the other one spins counterclockwise. The increment of the speed of rotor 3 respect to rotor 1 produces a positive pitch (θ>0), while increasing the speed of rotor 4 regarding rotor 2 produces a positive roll (ϕ>0). The increment of the speeds of rotors 1 + 3 respect to rotors 2 + 4 produces a positive yaw (ψ >0).

Using the Newton-Euler's method, the angular dynamic of the system is represented as follows [71]:

$$\tau = J\dot{\omega} + \omega \times J\omega \tag{3.1}$$

$$J = \begin{pmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{pmatrix} \tag{3.2}$$

Where τ is the vector of torques in the three axes in N·m, J is the inertia tensor in Kg·m^2, ω is the angular velocities vector in rad/s, and × represents the vector product.

The translational dynamic is given by:

$$m\dot{v} = RT - mge_3 \tag{3.3}$$

Where m is the mass of the quadrotor in Kg, R is the rotation matrix which is dimensionless, g is the gravitational acceleration in m/s^2, T is a vector of forces in N, and $e_3 = [0,0,1]^T$ is a unit vector which describes the rotor orientation.

The vectors τ and T are a function of the velocities of the propellers:

$$\tau = \begin{pmatrix} bl(\Omega_4^2 - \Omega_2^2) \\ bl(\Omega_3^2 - \Omega_1^2) \\ d(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 - \Omega_3^2) \end{pmatrix} \tag{3.4}$$

$$T = \begin{pmatrix} 0 \\ 0 \\ b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \end{pmatrix} \tag{3.5}$$

In equations (3.4-3.5), b is the thrust coefficient in N·s^2, d is the drag coefficient, l is the longitude of each arm in m, and $\Omega_1, \dots, \Omega_4$ are the velocities in rad/s of the rotors 1 to 4, respectively.

To simplify the calculations, instead of using the speed of the rotors, it is possible to define a set of control signals u_1, u_2, u_3 and u_4 as follows (2.6):

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -1 & 0 & 1 \\ -1 & 0 & 1 & 0 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} \tag{3.6}$$

This matrix is invertible, so it is possible to generate speed references for the rotors from a set of control signals.

Finally, from equations 3.1 to 3.6, the following system of equations is derived:

$$\ddot{\phi} = \dot{\theta}\dot{\psi}\,(I_y - I_z)/I_x + (lb/I_x)u_2 \tag{3.7}$$

$$\ddot{\theta} = \dot{\phi}\dot{\psi}\,(I_z - I_x)/I_y + (lb/I_y)u_3 \tag{3.8}$$

$$\ddot{\psi} = \dot{\phi}\dot{\theta}\,(I_x - I_y)/I_z + (d/I_z)u_4 \quad (3.9)$$

$$\ddot{X} = -(sin\theta cos\phi)\,(b/m)u_1 \quad (3.10)$$

$$\ddot{Y} = (sin\phi)\,(b/m)u_1 \quad (3.11)$$

$$\ddot{Z} = -g + (cos\theta cos\phi)\,(b/m)u_1 \quad (3.12)$$

3.3 MODELS BASED ON ARTIFICIAL NEURAL NETWORKS

In order to test the effectiveness of the neural techniques, one comparative study of models obtained by different PEM techniques and neural networks with different structures and configurations was carried out during the development of the Thesis. The preliminary results were initially published in [72]. In particular, the PEM models: ARX, OE and based on state space were compared with models obtained by artificial neural networks with the structures: multilayer perceptron, radial basis function networks, and generalized regression networks (for more details about these techniques see the Chapter 2). The MSE and MSE of the estimated step response were used to compare the results.

To perform the study, the dataset of input data of the network was made with the input signal in the current sampling time and in some previous sampling times, $pitch(t - Ki)$ where $i = 1 \ldots 100/K$ for different values of K. The output dataset was made with the values of the output signal (velocity in the x-axis, $vx(t)$.) in the current sampling time. 5000 samples were used, the 60% was used for the training and the remaining ones for the simulation (for more details of the training data see Section 2.1. It is graphically and quantitatively checked that the neural networks provide better results than the PEM models.

In other later work [73], the utility of the on-line learning ability of the artificial neural networks to model UAVs is analysed. In this case the input dataset was made with the signals pitch, roll and the velocity in the x-axis, in other words $[roll(t-i), pitch(t-i), v_x(t-i)] \longrightarrow v_x(t)$, and for the y-axis $[roll(t-i), pitch(t-i), v_y(t-i)] \longrightarrow v_y(t)$, where $i = 1 \ldots N$ with different values of N for each signal, N_θ for the roll, N_Φ for the pitch, N_{vx} for the velocity in the x-axis and N_{vy} for the velocity in the y-axis. Different combinations of $\{N_\theta, N_\Phi, N_{vx}\}$ and $\{N_\theta, N_\Phi, N_{vy}\}$ were tested to find out what provides less MSE. The aim of this study was to contrast the advantages of the on-line learning with the off-line learning. It is indeed observed how the neural networks with off-line learning are more sensitive to the training dataset, in

contrast, by the on-line learning, this bias is reduced due to the network learns and improves along the time.

3.4 NEURO-FUZZY MODELS

After checking the effectiveness of the neural networks for the UAVs modelling, it was studied whether the neuro-fuzzy networks were also able to provide good results. To do it, different models were obtained by ARX (PEM), radial basis function networks and ANFIS (Adaptive Neuro-fuzzy Inference System). The results were published in [74]. The structure of ANFIS is explained in the Section 2.2.3 of the Thesis.

In this case, the input dataset was made with the input signal in the current sampling time, $pitch(t)$, and the values in the 5 previous sampling times, $pitch(t - iTs)$ where $i = 1 \dots 5$, and Ts is the sampling period. The output dataset was made with the values of the output signal in the current sampling time $vx(t)$. In this article, the study also collects the relation $roll \longrightarrow vy$.

The configuration of the ARX was $na = 10$, $nb = 10$ and $tk = 1$, where na is the order of the numerator, nb the order of the denominator and tk is the number of the delays in the input signal. The number of neurons in the hidden layer was limited to 30. In the ANFIS technique, the number of partitions was set to $p = 2$, obtaining a set of 2^6=64 rules. In the same way that in the previous study, 5000 samples were used, the 60% were used in the training and the rest in the validation. Other new contribution in this study was the use of the MSE of the estimated frequency response to include the frequency domain in the comparative analysis.

3.5 NEW APPROACH: HYBRIDIZATION OF PARAMETRIC AND SOFT COMPUTING TECHNIQUES

Modelling is one of the approaches more used to work with complex systems. One representation of the system let us to analyse its behaviour and get a deeper knowledge about it. In control engineering, the availability of simulation models is essential to design control laws and test them without damaging the real systems. Therefore, one of the most useful steps to effectively control physic systems is to obtain useful and realistic models which describe them.

The importance of having a good model grows with the complexity of the system. On one side, the model must come closer to reality as much as possible but, by the other hand, the model should keep as simple as possible to reduce the computational cost. A low computation cost is an enabling key factor to run the model in low cost general purpose hardware platforms. The inner idea of the proposal of the hybrid model is to obtain models of complex systems keeping a good balance between accuracy and hardware requirements needed to run it.

When a modelling technique is selected, the final goal is to get realistic and achievable models. Realism is directly related to model's accuracy. It means that one of the premises must be to reduce the error to the possible extent. Meanwhile, the model's achievability is linked to the simplicity and the efficiency. In other words, more complex models are often less achievable and efficient. Normally, these ones are antagonistic concepts; improvements in accuracy often cause growth in complexity and reduction in achievability. And in the same line, more realistic models are often less achievable in terms of computational cost.

In this sense, PEM techniques provide simple models but whose accuracy may not be highlighted. On the other hand, methods based on intelligent techniques like neural networks, provide models which better fit the real behaviour of the system, but they usually demand much more memory and processing capability. These ideas suggest that the best method will be one which had the ability of joining accuracy and simplicity, combining them

This idea was the source of inspiration to the development of the method what its described below. The accuracy provided by the application of one modelling techniques may be enough for some uses, but it is possible that we need to increase its performance. To do it, one first option is to modify the configuration parameters of the model. For example: the polynomials in the PEM techniques, the size of the input layer of the neural networks, or the number of partitions in the ANFIS method. When this either does not work, or work but the increment of the computational cost is not acceptable, the hybrid techniques may play a major role.

The proposed hybrid method is composed of one topology more one training procedure. The topology is shown in the Figure 3.2. As it can be observed in the figure, it is based in a cascade connection of a parametric technique and a soft computing technique. A set of input signals, in the example of the figure just the $pitch$ in one case, and the $roll$ in the other one, feeds to a PEM model. The PEM model generate an only output what is the input of the soft computing technique. The PEM model is used to approximate the linear behaviour of the system

and the soft computing technique to carry out a fine tuning of the non-linearities. Thus, one qualitative or heuristic tuning of a parametric technique is performed.

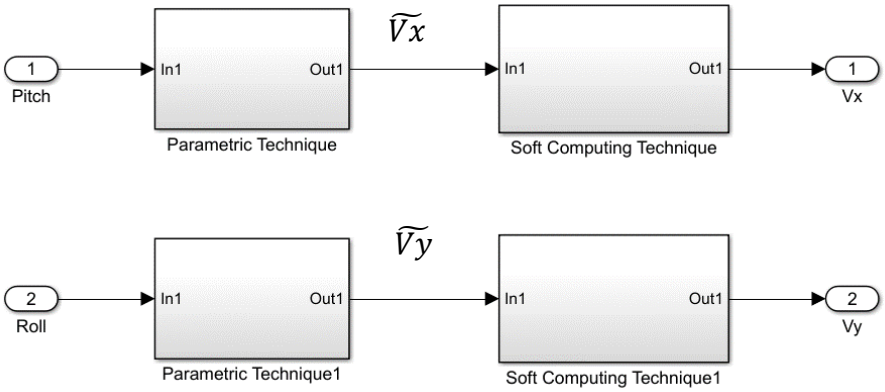

Fig. 3.2. Cascade connection of techniques

The procedure to obtain the complete model is composed of different steps:

1. **Dominant input identification:**

   This step is only needed when we work with an either MIMO or MISO system. One step signal is introduced in each input to identify what is the dominant input in each output. For example, in the case of the UAV (MIMO system), the Figure 3.3 shows the step response for the velocities vx (upper row) and vy (bottom row), respect the signals $pitch$ (left column) and $roll$ (right column).

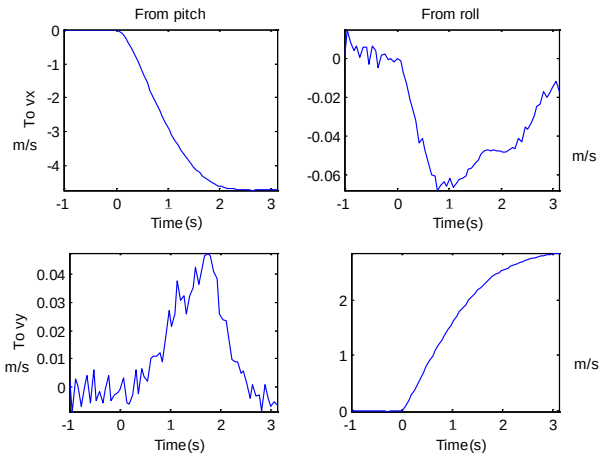

Fig. 3.3. Step response of the UAV

In the previous figure it is possible to observe that the stationary gain of the elements in the main diagonal is much higher than in the rest of the elements. This

fact indicates that the system may be partially decoupled, and therefore the MIMO system may be converted to two SISO systems. Considering this simplification, in this example we will focus in the influence of the $pitch$ in vx and the $roll$ in the vy. Because of this, the parallel modelling of two independent systems $pitch \rightarrow vx$ y $roll \rightarrow vy$ appears in the Figure 3.2 .

2. **PEM model identification**:

   Using input/output data of the real system, for example in the Figure 3.2 the input is the $pitch$ and the output is vx, the PEM model what fits better the system's response is selected. To do it, for each possible PEM structure (ARM, ARMAX, ARIMAX, BJ, OE, …) so many models as possible combinations of configuration parameters are generated. For example, the ARX structure has 2 configuration parameters: NA and NB. The first 100 configurations are generated obtaining the different combinations of NA and NB, where NA= {1,…10} and NB={1….10}. For each PEM structure the configuration what fits better the system's behaviour is selected. Finally, the best PEM structure is selected between the previous selected configurations.

3. **Simulation of the output of the PEM model:**

   The expected output of the system is simulated by the PEM model using a previously known set of input signals. In the example of the Figure 3.2 in the output of the PEM models the signals $\widetilde{vx}$ y $\widetilde{vy}$ are obtained. These outputs are what we would be expected to obtain if the system was purely linear.

4. **Soft computing technique training**:

   The soft computing technique is trained using the result of the simulation obtained in the previous point as input dataset, and the real known data as output dataset. In the example of the Figure 3.2, the input would be the estimated signal $\widetilde{vx}$ and the output would be the real signal vx for one of the models, and $\widetilde{vy}$ and vy for the other one.

To reach this procedure the possibility of combining PEM and neural techniques in cascade and parallel connection was initially studied. The initial results were published in [75]. As a better performance using the cascade combination was observed, since then this architecture has been used. Later it is checked that the proposal method is also valid to combine PEM and

neuro-fuzzy techniques. The results were published in [74]. The method is consolidated in [76], where it is proved that is able to simultaneously improve the accuracy and the computational complexity.

3.5.1 Results

In order to test the validity of the proposal, during the development of the Thesis, different models were obtained by PEM, neural and neuro-fuzzy techniques individually applied, as well as combining them by the proposed hybridization technique. The Figure 3.4 shows the best combination of techniques. For each axis, the dark blue line is the output of the real system, the green, red, light blue, magenta, and yellow lines represent the values of the output signal obtained by the models ARX-(10,10,1), RBF-(30,200), ANFIS-(8,2), ARX-(10,10,1)-RBF(5,50), and ARX-(10,10,1)-ANFIS(5,2), respectively. As it may be expected, the signals obtained only with the ARX models are the worst ones to both axis. By the other hand, the hybrid methods ARX-RB and ARX-ANFIS, provide better performance than the techniques individually applied, even although the technique is configured with the best parameters (highest size of the input layer and highest number of neurons in the hidden layer).

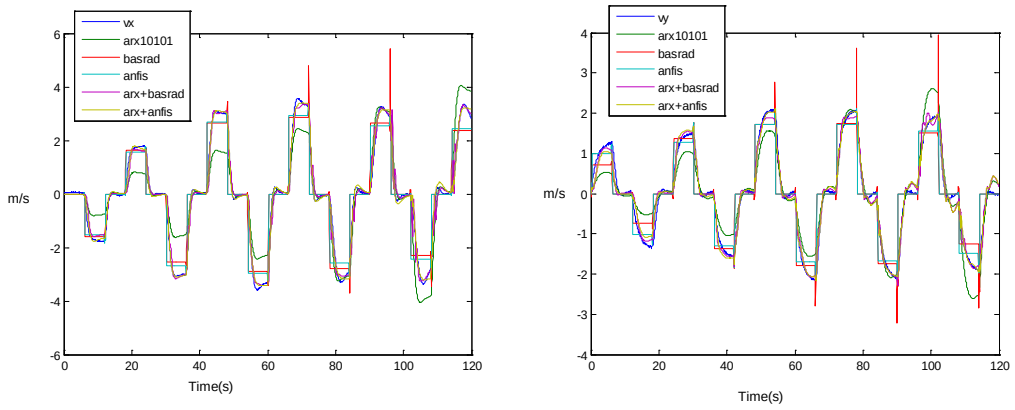

Fig. 3.4. Comparison of velocities in the x-axis (left) and in the y-axis (right) obtained by different methods

In the Table 3.1 there are three main groups of lines, each one of them is separated from the rest ones by a horizontal line. The first group of rows shows the results obtained by the individual application of the selected techniques. The second group of rows presents the results of the models generated by the hybridization of intelligent and parametric techniques. The third one also shows the result of the hybrid techniques, but in this case with a slightly increment of the complexity of the configuration; the parameter of the neural network change from (5,30)

either to (5,50) or to (10,50). For each column and for each main row, the best results have been boldfaced, and for each column the best results have been highlighted.

The last column of Table 3.1 represents the time complexity (TC). Analysing this indicator jointly with the MSE_C, it is interesting to point out the differences between the values for soft computing techniques, such as RB-(30,200), and the corresponding hybrid method (ARX-RB). In this case, TC= 38136 against TC=2356.6. On the one hand, RB is more accurate than the parametric technique ARX (0.0995 vs. 0.1746 MSEC error), but it is much more complex in terms of computation time (38136 vs. 91.6). Nevertheless, with the hybridization ARX-RB the precision is now 0.0170 while the TC is decreased to 2356.6.

In general, (Table 3.1) the proposed hybrid technique provides better results than the techniques individually applied, obtaining important reductions in the time complexity (more simplicity) without losing accuracy.

Table 3.1. Comparison in terms of MSE and TC for different techniques and combinations

Technique	v_x				v_y				TC
	MSE_T	MSE_S	MSE_F	MSE_C	MSE_T	MSE_S	MSE_F	MSE_C	
ARX10,10,1	0.5056	**0.0122**	**0.0060**	0.1746	0.1326	0.0024	**0.0077**	0.0476	**91.6**
OE6,6,1	0.2915	0.0418	0.0352	0.1228	0.0820	**0.0010**	0.0085	**0.0305**	54.56
RB-(5,30)	0.5097	1.1919	3.4022	1.7013	0.2345	0.3796	1.5013	0.7051	1497.9
RB-(30,200)	**0.0240**	0.0171	0.2574	**0.0995**	**0.0159**	0.0145	0.2729	0.1011	38136
ANFIS-(5,2)	0.4816	1.1807	6.1789	2.6137	0.2106	0.3765	2.1761	0.9211	3743.8
ANFIS-(8,2)	0.3363	0.8236	4.9864	2.0488	0.1531	0.2731	1.7381	0.7214	36276
ARX-ANFIS(5,2)	**0.0145**	**0.0087**	**0.0252**	***0.0161***	**0.0100**	0.0057	0.0198	0.0118	3835.4
OE-ANFIS(5,2)	0.0295	0.1356	0.1433	0.1028	0.0103	0.0030	0.0281	0.0138	3798.4
ARX-RB(5,30)	0.0572	0.0172	0.0774	0.0506	0.0144	**0.0015**	**0.0059**	**0.0073**	1589.5
OE-RB(5,30)	0.0429	0.1163	0.0848	0.0813	0.0127	0.0039	0.0162	0.0109	**1552.5**
ARX-RB(5,50)	0.0125	**0.0031**	0.0355	**0.0170**	0.0068	0.0036	0.0108	0.0071	**2356.6**
OE-RB(5,50)	0.0286	0.1114	0.0754	0.0718	0.0071	0.0007	0.0110	0.0063	**2356.6**
ARX-RB(10,50)	**0.0103**	0.0040	**0.0340**	0.0287	0.0059	0.0022	0.0108	0.0063	3764.1
OE-RB(10,50)	0.0270	0.0863	0.0445	0.0526	**0.0057**	**0.0004**	**0.0078**	***0.0046***	3727.1

Almost all previous works presented in the state-of-the-art section don't provide cuantitative results to compare with the data contained in the Table 3.1. Only in [56], MSE data for a velocity variable are provided. In this case, Puttige presents three neural models to identify a fixed-wing UAV, with the offline trained model obtains a MSE of 1.64 for the velocity, with on-line model 0.0534, and 0.0088 with the combined model. In one of the works developed during the Thesis [73], it is also observed how the models with on-line learning provide better results than the models with off-line learning.

4 Control of Quadrotors

This chapter describes the concepts and methods used to control the quadrotor, as well as the main outcomes. The Section 4.1 cites some of the main contributions of different authors in the control of quadrotors by neural techniques. The Section 4.2 is focused on the proposed control method based on artificial neuronal networks. Then the disturbance models are shown (Section 4.3). Finally the new anti-disturbance control scheme developed during the Thesis is studied (Section 4.4).

4.1 STATE OF THE ART

The neural networks have been widely used to control UAVs. One of the first significant papers comes from Kim and Calisse [77-78], who present the theoretical development of one architecture of adaptive control based on neuronal networks. Since then, it is possible to identify many other authors using these techniques to control UAVs. Mo and Farid collect a deep survey on the state of the art regarding adaptive intelligent control applied to UAVs [79]. Hereinafter some significant works in this field are cited.

Efe trains a neural network to provide the coefficients of a FIR filter, which approximates the response of a PI $^{\lambda}$ D $^{\mu}$ controller with variable parameters and derivation an integration orders [80]. The filter FIR provides a similar response to the PI $^{\lambda}$ D $^{\mu}$ but it reduces the computational cost. In [81], one hierarchical neural controller is used to control micro-quadrotors. In [82] an adaptive neural controller is proposed to stabilize the quadrotor against errors in the model and external disturbances. The same target is considered in [83], but with a different solution, in this case a neuro-fuzzy system made by a fuzzy inference Takagi-Sugeno (TS), where the consequent section is made by a set on differential neural networks is used. In [84] a network with one hidden layer based on ELM (Extreme Learning Machine) to approximate the non-modelled dynamic is used. In [85] neural networks are also used to approximate the uncertainties in the UAV's dynamics.

In several papers, the neural networks are used to model observers. In [86] a neural observer is used to estimate the translational and angular velocities of the UAV. In [87] the control errors are estimated by a neural state observer. In [88-89] a neural sliding mode observer is proposed to reconstruction of states. This observer has the same structure than a simple sliding mode observer, but to reduce the sensitivity to the noise, a neural network is added as a

correction term. All these observers may be especially useful to develop anti-disturbance controller, as we will see below.

Roughly speaking there are two controller design strategies regarding its behaviour against the disturbances: firstly, some controllers are designed without considering disturbances, then these ones are introduced in different points of the control scheme examining what is its influence and testing its robustness, by contrast, there is other category of controllers that are specially designed to react against the disturbances attenuating or rejecting them.

In this sense, [90] a survey of anti-disturbance control strategies is presented. Guo identifies two anti-disturbance groups: disturbance attenuation methods (such as stochastic control theory and robust control theory) and disturbance rejection schemes. The typical disturbance rejection approaches in turn include internal model control, output regulation theory, active disturbance rejection control (ADRC), embedded model control and disturbance observer-based control (DOBC). The Figure 4.1 shows how the disturbance observer may be introduced in the control loop. For linear systems, internal model control can be used to compensate the disturbance input with neutral stability conditions. The classical output regulation theory is applicable to the disturbance described with exogenous model [91-93]. The feasible disturbance compensation methods or output regulation results are difficult to achieve when the model or parameter of exogenous model is unknown. For a nonlinear system with uncertainties an adaptive internal model control method was proposed, where adaptive control technology was applied to the output regulation problem [94-96]. Furthermore, disturbance rejection methods based on adaptive estimation were presented for disturbance with uncertain model or unknown parameter [97-98]. For nonlinear output regulation theory, the solution is based on a Francis–Isidori–Byrnes nonlinear partial differential equation (PDE) [99]. ADRC theory has better prospects for engineering applications, but still lacks a rigorous theoretical proof of the stability [100].

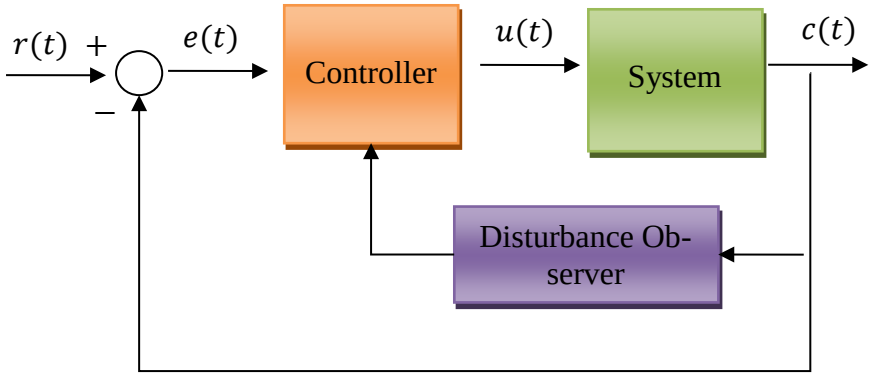

Fig. 4.1 Control loop with disturbance observer

Focusing on the control of quadrotors, there are few works that study the effect of the payload variation on the quadrotor dynamics and that take it into account. In [101], an adaptive control is designed to mitigate the impact of the parameter variation by estimating them under guaranteed performance. In [102], least square and gradient methods are implemented for adaptive parameter estimation, which are used to update the control output to the current UAV mass and inertia moment. Recently, Wang [103] applies again the same strategy to estimate the variations in the payload and the effect of the wind gusts.

Other studies are focused on the rejection of wind disturbances. In [104], Lyapunov-based observers are used to estimate the external force disturbances. In [105], a control strategy based on sliding-mode and adaptive control techniques is proposed to deal with slow and fast time-varying wind conditions. In [106], a switching model predictive attitude controller for an unmanned aerial vehicle subject to atmospheric disturbances is presented. In [107], a nonlinear adaptive state feedback controller for thrust and torque actuation is designed, so it guarantees global convergence of the closed-loop path following in the presence of constant wind disturbances. In [108], a sliding mode control driven by sliding mode disturbance observer (SMC-SMDO) approach is used to design a robust flight controller for a small quadrotor.

The cited studies are mainly focused either on payload variations or on wind disturbances rejection. Few recent papers have been found that address both problems at the same time, such in [103]. This approach was addressed during the development of the Thesis, as we will see later, and the results were published in [109].

4.2 PROPOSED METHOD: ADAPTIVE NEURO-CONTROLLER

The ability of the artificial neural networks to approximate functions allows them to be used to design controllers. Regarding this point, there are different control strategies based on neural networks: direct control, inverse direct control, indirect learning, generalized learning, specialised learning… One systematic classification is presented in [110]. During this work, in a first approach [111], a variant of the generalized learning algorithm (GLA) was used to control the quadrotor. The Figure 4.2 shows the structure of the GLA.

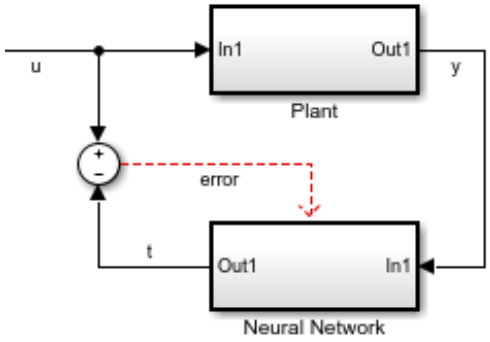

Fig. 4.2.Generalized learning algorithm

In the generalized learning [112], a set of desired motor commands, denoted $\{u\}$ is used to drive the system and the set of resulting trajectories is denoted $\{y\}$. Then the artificial neural network (in the literature sometimes it is called Inverse Dynamics System or IDS) receives $\{y\}$ as input and yield a set of reference motor commands, denoted as $\{t\}$. The goal of the generalized learning is to minimize the errors between $\{t\}$ and $\{u\}$ in the least square sense. After the neural network is well trained, if a real input $\{y'\}$ is sufficiently close to one trajectory in the set $\{y\}$, the controller should be able to retrieve a proper motor command $\hat{t}$, making the actual movement $\hat{y}$ closely follows y'.

The variation proposed respect the GLA is the on-line refinement and the combination with a PID. The first step is the application of the GLA algorithm to off-line training the neural network in order to identify the inverse dynamic of the plant (Figure 4.2). Once the network has been off-line trained, it is placed in cascade connection with the plant and a PID controller. Then, the configuration of the network is on-line refined. To do this, during each control interval two processes are sequentially applied to the net-work (first the simulation, later the on-line learning):

1. **Simulation:** The output of the PID, $PID_{OUT}(t_i)$, feeds one of the inputs of the artificial neural network; the rest of the inputs are past values of the plant output, $PLANT_{OUT}(t_i - j * Ts)$. The network generates the control input, u_1, which is the input of the plant (Figure 4.3, switch in the upper position).

$$u_1(t) = f_{NET}\left(PID_{OUT}(t), PLANT_{OUT}(t - i * Ts), par_{NET}(t - Ts)\right) \quad i = 1 \dots (N_{inputs} - 1) \quad (4.1)$$

2. **On-line learning:** The neural network is trained again with the current and previous outputs of the plant, to generate the control output, u_1, obtaining the new configuration parameters par_{NET}. The network input dataset is made up of the past values of the plant output, $PLANT_{OUT}(t_i - j * Ts)$. The output dataset is the current value of the plant input $u_1(t_i)$ (Figure 4.3, switch in the lower position)

$$par_{NET}(t) = f(PLANT_{OUT}(t - i * Ts), u_1(t), par_{NET}(t - Ts)) \quad i = 0 \dots (N_{inputs} - 1) \quad (4.2)$$

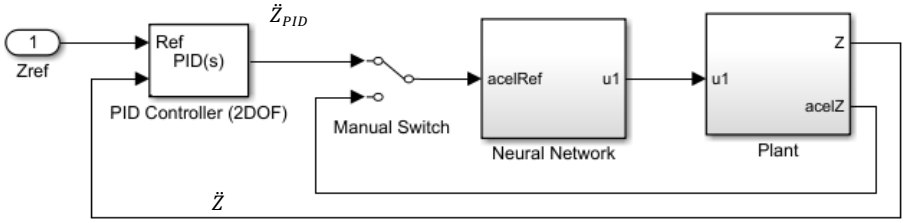

Fig. 4.3. Adaptive altitude neuro-control strategy

In the Figure 4.3 a specific configuration of the method to be applied in altitude control is shown. UAVs are normally provided with accelerometers, so it is assumed that the acceleration in the z-axis ($\ddot{Z}$) is available. The network must be able to simulate the control signal u_1 by using acceleration measurements. Thanks to the artificial neural network, the PID does not need to include the plant gain. The network is able to learn the plant gain and work with it. In other words, with this approach it is not necessary to know the system parameters to control it [111].

Other advantage to highlight of this proposal is the on-line learning ability. This feature let the network to learn continuously while the system is working and adapt to variations along the time. This is extremely useful when we must model systems subject to changes, as for example, outdoor systems subject to changing weather conditions or external actions able to modify the system's parameters. Both conditions may be applied to the quadrotor control. The artificial neural networks are usually off-line trained. During this process their inner parameters are adjusted to reduce the MSE between the real output and the output data of the training

dataset. Since then, new input-output data are presented to the network and its parameters are slightly updated to keep a low MSE along the time. This last process is normally called either adaption or on-line learning.

In [111] the robustness of the method against changes in the mass and external step disturbance in the altitude control is checked. Later the validity of the method also to control the Euler's angles is tested. Also the step disturbance is substituted by a more realistic model of the wind disturbance. The results are published in [113]. These papers don't propose a specific anti-disturbance control strategy, but an adaptive neuro-control strategy and test its robustness against disturbances and variations. In the Section 4.4 the specific disturbance reject controller developed during the Thesis will be presented.

4.2.1 Results

In the Figure 4.4 some results of the application of this method in the control of altitude and the control of the Euler's angles. In the left, the result of the altitude control is shown (blue line), and the green line is the reference. The right side collects an example of control of Euler's angles, where the red line is the roll, the yellow one is the pitch, the purple one is the yaw, and the blue one is the reference.

In both cases the controllers are off-line trained from t=0 s up to t=2 s. For this reason, the reference is zero in this interval. The control signals used to train the network produce changes in the altitude and the attitude during this period, these changes can be observed in the figure. The control phase starts at t=2 s; in this point the reference of altitude is set to 5 m and the Euler's angles to $\pi/4$ radians. In the figures it may be seen how the control is able to stabilize the signals around the reference signal.

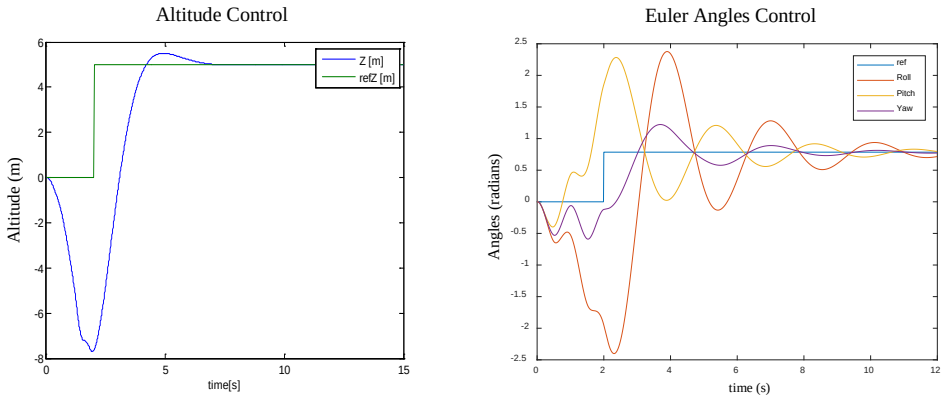

Fig. 4.4. Control of altitude (left) and control of Euler angles (right)

In other experiment the validation the controllers against variations in the mass was tested. The performance of the controllers with and without on-line learning is compared in the Table 4.1. It is noticeable how the overshoot and the steady state error are clearly reduced by the on-line learning. The settling time is also slightly reduced. As expected, the learning makes the response a bit slower. It is significant how this control approach improves the response respect to the PID, specially the overshoot

Table 4.1 Performance of the controllers with variations in the mass (altitude control)

Parameter	Without on-line learning	With on-line learning	Only with PID	Wang(2014)[102]
Rise time (s)	**2.05**	2.24	2.27	1
Settling time (s)	9.42	**7.44**	8.56	8
Overshoot (%)	9.66	**2.26**	34.04	10
Steady state error	5.31	**0.015**	**0.015**	0.02

In Table 4.1, data of the experiment of Wang [102] also have been included. These data must be cautiously compared due to in his article the UAV is different and the variation in the mass is also different.

4.3 MODELS OF CHANGE OF MASS AND INFLUENCE OF THE WIND

In the Thesis the variation in the payload and the influence of the wind were considered as disturbances during the design of the controllers. The variation of the payload is simulated by the addition of the term $dist_m$ in the equations 3.10 to 3.12, obtaining the equations 4.3 to 4.5.

$$\ddot{X} = -(sin\theta cos\phi)\,(b/(m * dist_m))u_1 \qquad (4.3)$$

$$\ddot{Y} = (sin\phi)\,(b/(m * dist_m))u_1 \qquad (4.4)$$

$$\ddot{Z} = -g + (cos\theta cos\phi)\,(b/(m * dist_m))u_1 \qquad (4.5)$$

Two different payload variations have been considered: a step profile is applied (4.6), and a sinusoidal variation profile (4.7).

$$dist_m(t) = 2 + step(t-4) \qquad (4.6)$$

$$m(t) = m_{model}\,\big(step(t) - step(t-4)\big) + step(t-4)\left(m_{model} + \frac{m_{model}}{2}\sin\left(\frac{2\pi}{8}t\right)\right) \qquad (4.7)$$

To model the influence of the wind new terms are added to the acceleration of the system, $dist_wX$, $dist_wY$, $dist_wZ$, given by the equations 4.9 to 4.11. The wind is assumed in the direction of the movement [114]:

$$v_w(Z) = v_{w(Z=20)} \cdot \frac{\log(Z/C)}{\log(20/C)} \tag{4.8}$$

$$dist_wX = sgn(v_w(Z)) \cdot \rho_{air} \cdot A_x \cdot Cd \cdot \left(\dot{X} - 2 * v_w(Z)\right)^2/(2m) \tag{4.9}$$

$$dist_wY = sgn(v_w(Z)) \cdot \rho_{air} \cdot A_y \cdot Cd \cdot \left(\dot{Y} - 2 * v_w(Z)\right)^2/(2m) \tag{4.10}$$

$$dist_wZ = sgn(v_w(Z)) \cdot \rho_{air} \cdot A_z \cdot Cd \cdot \left(\dot{Z} - v_w(Z)\right)^2/(2m) \tag{4.11}$$

Where $v_{w(Z=20)}$ is the wind speed at 20 m of altitude in m/s, $v_w(Z)$ is the wind speed at Z altitude in m/s, C is a constant related to the flight (in this experiment the value is set to 1.5) which is dimensionless, ρ_{air} is the air density in Kg·m3, A_x, A_y and A_z are the effective area of the quadrotor exposed to each component of the wind in m2, Cd is the drag coefficient respect to the wind which is dimensionless, $\dot{X}$, $\dot{Y}$, $\dot{Z}$ are the velocities in the x-axis, y-axis and z-axis in m/s, and sgn denotes the sign function.

The wind speed is simulated by a step with Gaussian noise at t = 4 s. The SNR between the average wind and the noise is 10 dB. The average wind speed is 12 m/s in the z-axis and 24 m/s in the x-axis and y-axis. These values match numbers 6 and 9 in the Beaufort's scale (strong breeze and strong gale) [115]

The Figure 4.5 shows an example of the wind disturbance in the x-axis (left) and y-axis (right). The simulated disturbance is represented in blue and the estimated disturbance by the controller in red

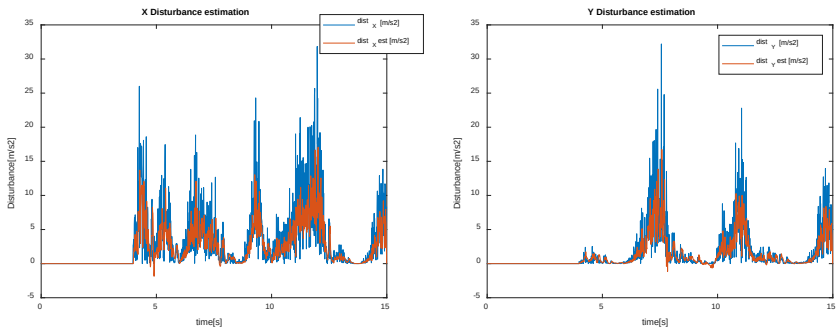

Fig. 4.5 Wind disturbance estimation in the x and y axis

4.4 PROPOSED METHOD: CONTROLLER BASED ON ADAPTIVE NEURO-ESTIMATORS

During the development of this Thesis, as an evolution of the neural controller shown in the Section 4.2, the design of an intelligent control strategy based on adaptive neural networks to cope with external disturbances, such as payload changes and wind was proposed. The results were published in [109]. The final goal is to stabilize the UAV and to improve the system performance. The control strategy that has been used falls within the category disturbance observer-based control (DOBC), seen in Section 4.1. In our case adaptive neuro observers are used to estimate the payload and wind disturbance.

The control scheme for quadrotor tracking control proposed to tackle payload and wind disturbances is shown in the Figure 4.6. The UAV model control inputs are the four control signals, $u_i, i = 1,..,4$, that represent the power of the rotors. Four main controllers are defined to obtain these model inputs, the controllers of the X, Y, and Z coordinates, and another for the yaw angle. This is because the pitch and roll angles are used to track the $Xref$ and $Yref$ reference coordinates. The control of Z is carried out by the control signal u_1. The Y controller generates the roll reference, and the tracking of the roll angle is performed by the control signal u_2. In the same way, the controller of X generates the pitch angle reference value, and control signal u_3 will be in charge of getting this value. The control signal u_4 is used to stabilize the yaw angle around zero. The aim of the PIDs controllers that appear in Figure 4.6 is to generate the acceleration references to make the attitude error ($\phi_r - \phi, \theta_r - \theta, \psi_r - \psi$) and the tracking error ($Xref - X, Yref - Y, Zref - Z$) converge to zero. The rest of the controllers are used to compensate the non-linearities of the system. Adaptive neuro estimators feed the inputs of the X, Y and Z controllers to compensate the payload and wind influence.

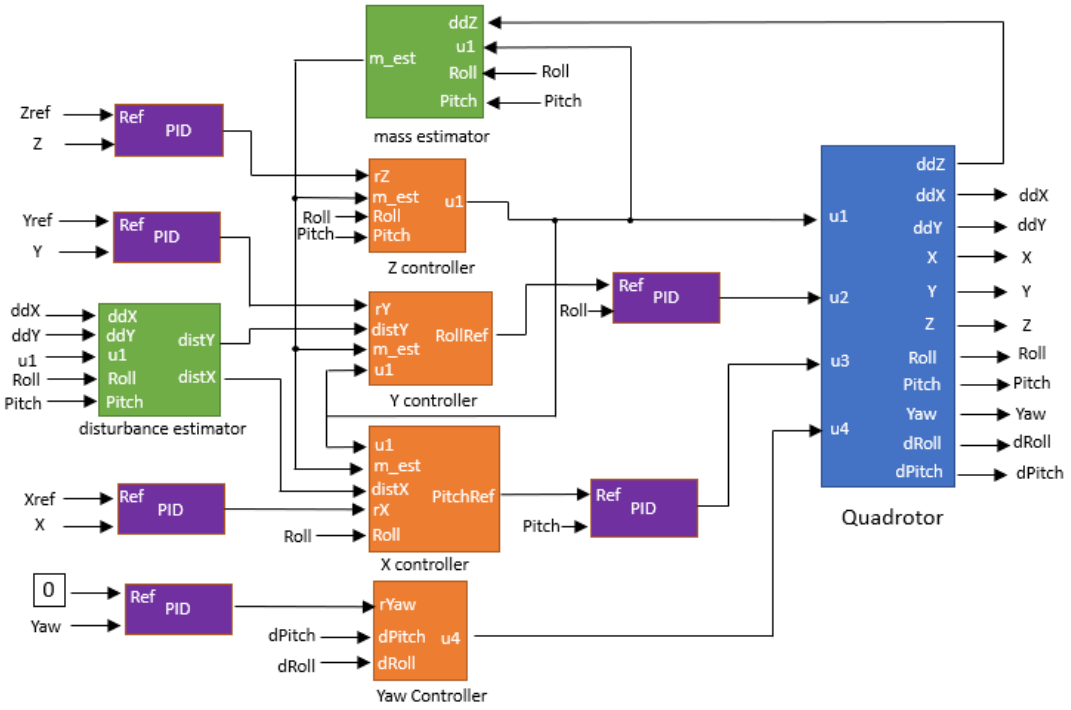

Fig. 4.6.. Control system with mass and disturbance estimators

Although the mass estimation (m_{est}), the estimation of the disturbance in the x-axis ($distX_{est}$) and the estimation of the disturbance in the y-axis ($distY_{est}$) have been implemented in different blocks, their inner structure is the same. The differences come from the configuration parameters and the input/output signal. Therefore, they can be jointly explained.

The estimator is based on an artificial neural network with on-line learning. Thus, there is one neural network to, let say, model each of the three parameters or disturbances considered. The Figure 4.7 represents its generic structure.

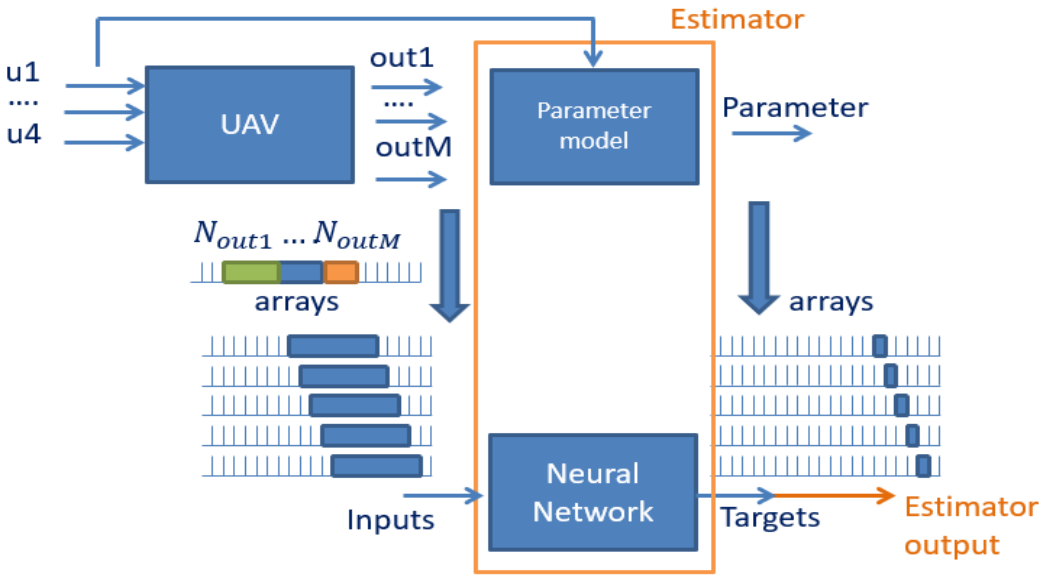

Fig. 4.7. Neural estimator structure and configuration

The parameter model receives the inputs and outputs of the UAV. The output of the parameter model is used as target output of the neural network during the training. The output of the proposed estimator is always the output of the neural network. The parameter model is needed because we are using supervised artificial networks and, the inputs and their corresponding outputs must be known.

We will use one parameter model for the mass, one parameter model for the wind disturbance in the x-axis and other for the y-axis.

Each element of the dataset used to off-line train the network is composed by:

- Target: Parameter (disturbance) value calculated with the model at instant t_i, param(t_i)
- Network inputs: for each output signal i, the previous N_i values to t_i are collected and structured as an array.

A theoretical example would be the following. At t_i=10 and for M = 3 outputs with a configuration Nout1=3, Nout2=2, Nout3=1.

- Target: param(10)
- Inputs: [out1(7), out1(8), out1(9), out2(8), out2(9), out3(9)]

If the parameter cannot be calculated (division by zero, squared root of negative numbers, or any other singularity) that element is not included into the training dataset. In case that the parameter cannot be calculated, the network is especially useful to estimate the disturbances. In these occasions, it may be said that the network generates new knowledge.

The specific configuration for each estimator of the Figure 4.7. is the following:

Mass estimation:

- Target output: The mass is approximated with the model:

$$\widetilde{m}(t_i) = \frac{\ddot{Z}(t_{i-1})+g}{u_1(t_{i-1})cos\theta_{i-1}cos\phi_{i-1}} \qquad (4.12)$$

- Network inputs:
  - Acceleration in the z-axis, $\ddot{Z}(t_{i-1})$
  - Roll and pitch cosine angles multiplied ($cos\theta_{i-1}cos\phi_{i-1}$)
  -

Disturbance in X axis estimation:

- Target output: The disturbance is approximated with the model:

$$\widetilde{distX}(t_i) = -(sin\theta_{i-1}cos\phi_{i-1})\frac{bu_1(t_{i-1})}{m_{est}(t_{i-1})} - \ddot{X}(t_{i-1}) \quad (4.13)$$

- Network inputs:
  - Acceleration in the x-axis: $\ddot{X}(t_{i-1})$
  - Roll angle: θ_{i-1}
  - Pitch angle: ϕ_{i-1}

Disturbance in Y axis estimation:

- Target output: The disturbance is approximated with the model:

$$\widetilde{distY}(t_i) = sin\phi_{i-1}\frac{bu_1(t_{i-1})}{m_{est}(t_{i-1})} - \ddot{Y}(t_{i-1}) \quad (4.14)$$

- Network inputs:
  - Acceleration in the y-axis: $\ddot{Y}(t_{i-1})$
  - Roll angle: θ_{i-1}
  - Pitch angle: ϕ_{i-1}

4.4.1 Results

Figure 4.8, left shows the tracking of the trajectory, the reference in blue, the trajectory with adaptive neuro estimators in red, and without estimators in yellow. At the right side of Figure 4.8, the tracking error is shown with estimators (red line) and without estimators (blue line). In both figures it is possible to see how the performance of the control strategy with the neuro estimators is much better. Also, it is possible to see how the tracking error is the same until t = 4 s; because before there is not wind disturbance, but from that moment on the tracking error of the controller without the estimators increases significantly.

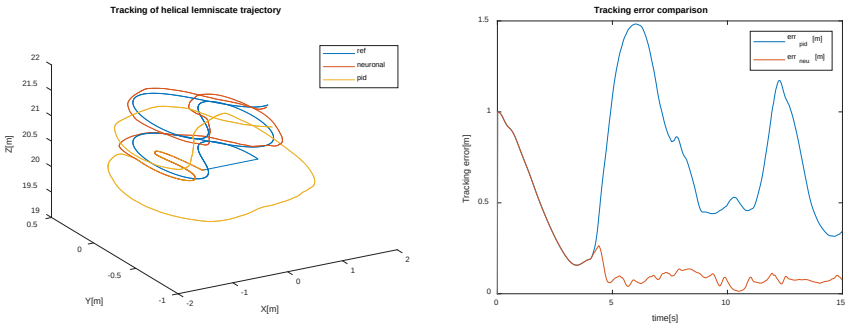

Fig. 4.8. UAV tracking of a helical lemniscate trajectory with wind disturbance (left) and its tracking error (right)

The controller has been extensively tested for different trajectories and the MSE numerical results are summarized in the Table 4.2 for the different trajectories and for each coordinate, with the neural estimator (Neuro) and without it (PID). The last column shows the absolute error. The best result for each component and trajectory is boldfaced.

Table 4.2: Comparison of the MSE of the tracking error for different trajectories with wind disturbances

Trayectoria	MSE_X		MSE_Y		MSE_Z		MSE_T	
	Neuro	PID	Neuro	PID	Neuro	PID	Neuro	PID
Lineal	0.0707	0.0707	0.0708	0.0708	**1.6163**	2.1214	**0.6730**	1.0130
Circular	**0.2805**	0.8612	**0.0739**	0.2596	**1.6186**	2.2711	**0.7591**	1.6014
Helicoidal	**0.0951**	0.4850	**0.0042**	0.2009	**0.0048**	0.2847	**0.2181**	0.8666
Helicoidal Cíclica	**0.1053**	0.3435	**0.0132**	0.2000	**0.0069**	0.2596	**0.2746**	0.8084
Lemniscata	**0.0911**	0.1928	**0.0011**	0.0335	**0.0005**	0.3036	**0.1748**	0.6008
Lemniscata Helicoidal	**0.0940**	0.2614	**0.0032**	0.0695	**0.0010**	0.2460	**0.2015**	0.6903

According to Table 4.2, the controller with adaptive neuro estimators provides less or equal tracking error for every applied trajectory. The maximum tracking error is obtained with the circular trajectory due to the high error of the initial conditions.

5 Conclusions and Future Work

5.1 CONCLUSIONS

- In this Thesis different models of unmanned aerial vehicles (UAVs) have been generated by the application of parametric methods, soft computing techniques and hybrid techniques. The comparison of the results has released that the hybrid strategies which combine parametric and soft computing techniques provide models whose relation accuracy-complexity is better than for the individually applied techniques. To carry out the comparison, specific error indicators in time and frequency domains were generated. The computational complexity was analysed by the breakdown of the individual techniques into their elemental operations.

- Also, it has been analysed how the training dataset influences the model when supervised learning neural networks are applied. The neural networks with off-line learning indeed have the disadvantage of being very sensitive to the training dataset partition. On the other hand, it has been observed how the adaptive neural networks with on-line learning are more robust because they are continuously learning from the real data of the system. Therefore, it is not needed to select a partition of the dataset to train the network and this fact does not produce bias.

- In a first approach to the design of anti-disturbance controllers, an adaptive neural controller was proposed. To do it, a MLP with a generalized learning algorithm modified to include on-line learning was applied. The simulation results validated the effectivity of this controller, even with disturbances such as the mass variations and the wind. Furthermore, the on-line learning improved the robustness of the controller by the reduction of the effects of the disturbances in the system.

- Finally, an anti-disturbance control strategy based on adaptive neural estimators was designed. In the proposed architecture the analytic equations of the model are introduced in the control scheme and adaptive neural observers are designed to estimate the mass and the disturbances. In this case, the neural networks focus on the varying terms: the mass, and the disturbance produced by the wind, and in general the non-modelled dynamics. This control strategy was intensively tested by simulation with different trajectories: linear, helicoidal, circular helicoidal, even a lemniscate. In fact, it is observed

how the adaptive neural estimators improve the robustness of the controller by the reduction of the effects produced by these variations.

5.2 FUTURE WORK

There exist several future lines of work. Firstly, the modelling and control techniques exposed may be applied to different types of UAVs. As it has been shown in the first chapter, nowadays the quadrotor is not the only possible UAV, but there exist others such as the single-rotor helicopter, fixed-wings, VTOL, etc. All of them could be seen benefited from the outcomes of this research. Moreover, the field of the application even could be extended to other areas since the modelling techniques under study provide general applicability.

The proposed hybrid modelling technique previously seen, provides an excellent accuracy-complexity relation. However, this relation is not constant, it depends on the configuration of the neural network (number of hidden layers, number of neurons, etc). This relation could be maximized by a optimization algorithm that consider the MSE and the analysed equations of the complexity of the net. This research would help to accelerate the deployment of the neural networks in applications with low cost microprocessors. Also, the modelling technique proposed uses the cascade connection, thus other different topologies could be explored to design new hybrid models.

Regarding the quadrotor control topic, a typical question when a new controller is proposed is: what are the minimal hardware requirements to run it in a real hardware platform? Therefore, a possible future line is analysing the computational complexity of the control scheme with the adaptive neuro estimators in order to establish the minimal hardware requirements. The total number of elemental operations will increase in accord to the control frequency and the learning frequency. One interesting point is to examine the relation between these frequencies and the performance of the controller, this will help to establish a compromise between the cost of the hardware and the control performance.

Other future work is to extend the range of disturbances considered. In this sense, a current challenge is the control of quadrotors with suspended loads by a wire. In this problem appear new components not considered previously: first, the centre of gravity of the complete system changes every time the relative position of the load respect the quadrotor is displaced; second, the quadrotor is subject to a changing external force produced by the strain in the wire.

Finally, another broad research line could be studying the best way to introduce the concept of the adaptive neuro estimators in other control strategies such as LQR, sliding, etc to improve their performance.

Bibliography

1. Chen, H., Wang, X. M., & Li, Y. (2009, November). A survey of autonomous control for UAV. In 2009 International Conference on Artificial Intelligence and Computational Intelligence (Vol. 2, pp. 267-271). IEEE.
2. Leonard, Barry, ed. Department of Defense dictionary of military and associated terms. DIANE Publishing, 2011.
3. Ruiz, A. G. E., Alazki, H., Rubio, J. E. V., & Salazar, O. G. (2016). Embedded super twisting control for the attitude of a quadrotor. IEEE Latin America Transactions, 14(9), 3974-3979.
4. Santos, M., Lopez, V., & Morata, F. (2010, November). Intelligent fuzzy controller of a quadrotor. In Intelligent Systems and Knowledge Engineering (ISKE), 2010 International Conference on (pp. 141-146). IEEE.
5. Wu, Z., Guan, Z., Yang, C., & Li, J. (2017). Terminal guidance law for UAV based on receding horizon control strategy. Complexity, 2017, Article ID 2750172, Volume 2017 (2017)
6. Ortega, J.J., Sigut, M. (2016). A low-cost mobile prototype for high-realism flight simulation. RIAI - Revista Iberoamericana de Automática e Informática Industrial, 13(3):293-303.
7. Ríos, H., Falcón, R., González, O. A., & Dzul, A. (2019). Continuous Sliding-Mode Control Strategies for Quadrotor Robust Tracking: Real-Time Application. IEEE Transactions on Industrial Electronics, 66(2), 1264-1272.
8. Garcia-Auñón, P., Santos, M., Cruz, J.M. (2017). A new UAV ship-tracking algorithm. Preprints of the 20th IFAC World Congress, pp. 13632-13637.
9. Zhou, Z., Wang, H., & Hu, Z. (2018). Event-based time varying formation control for multiple quadrotor UAVs with markovian switching topologies. Complexity, 2018.
10. Zhang, X., Li, X., Wang, K., & Lu, Y. (2014). A survey of modelling and identification of quadrotor robot. In Abstract and Applied Analysis (Vol. 2014). Hindawi
11. Chapman, A. (2018). Types of drones: multi-rotor vs fixed-wing vs single rotor vs hybrid VTOL. https://www.auav.com.au/articles/drone-types/
12. Pxhere (2019). https://pxhere.com/en/photo/762909
13. Naval Sea Systems Command (2019). http://www.navsea.navy.mil/Home/Team-Ships/Media-Gallery/PhotoGallery/igphoto/2001263757/
14. Pexels (2019). https://www.pexels.com/photo/drone-field-grass-helicopter-167777/
15. Nasa Technology Transfer Program (2019). https://technology.nasa.gov/patent/LAR-TOPS-241
16. Zadeh, Lotfi. A. (1994). Soft Computing and Fuzzy Logic. IEEE Software, vol. 11, no. 6, 48-56.
17. Pratihar, D. K. (2007). Soft computing. Alpha Science International, Ltd.
18. Ping, J. T. K., Ling, A. E., Quan, T. J., & Dat, C. Y. (2012, October). Generic unmanned aerial vehicle (UAV) for civilian application-A feasibility assessment and market survey on civilian application for aerial imaging. In Sustainable Utilization and Development in Engineering and Technology, 2012 IEEE Conference on (pp. 289-294). IEEE.

19. Adams, S. M., & Friedland, C. J. (2011, September). A survey of unmanned aerial vehicle (UAV) usage for imagery collection in disaster research and management. In 9th International Workshop on Remote Sensing for Disaster Response (Vol. 8).
20. Lin, J., Tao, H., Wang, Y., & Huang, Z. (2010, June). Practical application of unmanned aerial vehicles for mountain hazards survey. In Geoinformatics, 2010 18th International Conference on (pp. 1-5). IEEE.
21. San Juan, V., Santos, M., & Andújar, J. M. (2018). Intelligent UAV map generation and discrete path planning for search and rescue operations. Complexity, Article ID 6879419, 17 pages.
22. Goodwin, G. C., & Payne, R. L. (1977). Dynamic system identification: experiment design and data analysis (Vol. 136, pp. x+-291). New York: Academic press.
23. Narendra KS, Parthasarathy K (1990): Identification and control of dynamical systems using neural networks. IEEE Trans Neural Networks 1(1):4–27.
24. Mathworks (2019). https://es.mathworks.com/products/matlab.html
25. CEA (2016). Control Engineering Group of the Spanish Committee of Automatic: http://intranet.ceautomatica.es/og/ingenieria-de-control
26. Parrot (2019). https://www.parrot.com
27. Wikimedia (2019). https://upload.wikimedia.org/wikipedia/commons/thumb/d/d6/81RNYV29HCL._SL1500_%281%29.jpg/800px-81RNYV29HCL._SL1500_%281%29.jpg
28. Diaz J.M, Muñoz R (2012): Apuntes de identificación de sistemas. UNED
29. Horváth,G. (2005): Neural networks for system modeling. Technical Report, Budapest University of Technology and Economics.
30. Albus, J.S., (1975): A new approach to manipulator control: the cerebellar model articulation controller (CMAC). In: Trans. ASME, Series G. Journal of Dynamic Systems, Measurement and Control, 97, 220-233, 1975.
31. Levenberg, K. (1944): A method for the solution of certain non-linear problems in least squares. Quarterly of Applied Mathematics 2: 164–168.
32. Park J, Sandberg IW (1991): Universal approximation using radial-basis-function networks. Neural computation 3(2): 246-257.
33. Norgaard M (Ed.) (2000) Neural networks for modelling and control of dynamic systems - A practitioner's handbook. Springer, London, UK.
34. Mathworks (2019). https://es.mathworks.com/help/deeplearning/ug/generalized-regression-neural-networks.html
35. Jang, J.S.R (1993): ANFIS: adaptive-network-based fuzzy inference system. Systems, Man and Cybernetics, IEEE Transactions on 23 (3), 665-685.
36. Hoffer, N. V., Coopmans, C., Jensen, A. M., & Chen, Y. (2014). A survey and categorization of small low-cost unmanned aerial vehicle system identification. Journal of Intelligent & Robotic Systems, 74(1-2), 129-145.
37. Gessow, A., & Myers, G. C. (1952). Aerodynamics of the helicopter. Frederick Ungar.
38. Bramwell, A. R. S. (1976). Helicopter dynamics. Edward Arnold Ltd. Publisher, London.
39. Cai, G., Chen, B. M., Peng, K., Dong, M., & Lee, T. H. (2008). Modeling and control of the yaw channel of a UAV helicopter. Industrial Electronics, IEEE Transactions on, 55(9), 3426-3434.
40. Budiyono, A., Yoon, K. J., & Daniel, F. D. (2009, June). Integrated identification modeling of rotorcraft-based unmanned aerial vehicle. In Control and Automation, 2009. MED'09. 17th Mediterranean Conference on (pp. 898-903). IEEE
41. El-Saadany, A., Medhat, A., & Elhalwagy, Y. Z. (2009, May). Flight simulation model for small scaled rotor craft-based UAV. In 13th International Conference on Aerospace Sciences & Aviation Technology, Paper: ASAT-13-CT-31

42. del-Cerro, J., Martinez, A., & Barrientos, A. (2009). Modeling and control prototyping of unmanned helicopters. In: Aerial Vehicles, Thanh Mung Lam(Ed.), INTECH Open Access Publisher, pp. 87-106.
43. Castillo, P., Lozano, R., & Dzul, A. E. (2005). Modeling and control of mini-helicopters. Modeling and control of mini-flying machines. Advances in Industrial Control pp 81-119, 81-119, Springer.
44. Wu, H., Sun, D., & Zhou, Z. (2004). Model identification of a micro air vehicle in loitering flight based on attitude performance evaluation. Robotics, IEEE Transactions on, 20(4), 702-712.
45. Morris, J. C., Van Nieuwstadt, M., & Bendotti, P. (1994, June). Identification and control of a model helicopter in hover. In Proceedings of 1994 American Control Conference-ACC'94 (Vol. 2, pp. 1238-1242). IEEE.
46. Mettler, B., Tischler, M. B., & Kanade, T. (1999, May). System identification of small-size unmanned helicopter dynamics. In Annual Forum Proceedings- American Helicopter Society (Vol. 2, pp. 1706-1717).
47. Tischler, M. B., & Cauffman, M. G. (1992). Frequency-response method for rotorcraft system identification: flight applications to BO 105 coupled rotor/fuselage dynamics. Journal of the American Helicopter Society, 37(3), 3-17.
48. Tischler, M. B. (2018). System identification methods for aircraft flight control development and validation. In Advances in aircraft flight control (pp. 35-69). Routledge.
49. Ahmad, S. M., Shaheed, M. H., Chipperfield, A. J., & Tokhi, M. O. (2000). Nonlinear modelling of a twin rotor MIMO system using radial basis function networks. In Proceedings of the IEEE 2000 National Aerospace and Electronics Conference. NAECON 2000. Engineering Tomorrow (Cat. No. 00CH37093) (pp. 313-320). IEEE.
50. Hashimoto, S., Ogawa, T., Adachi, S., Tan, A., & Miyamori, G. (2000). System identification experiments on a large-scale unmanned helicopter for autonomous flight. In Proceedings of the 2000. IEEE International Conference on Control Applications. Conference Proceedings (Cat. No. 00CH37162) (pp. 850-855). IEEE.
51. Yuan, W., & Katupitiya, J. (2011, December). A time-domain grey-box system identification procedure for scale model helicopters. In Proceedings of the 2011 Australasian Conference on Robotics and Automation.
52. Manai, M., Desbiens, A., & Gagnon, E. (2005, August). Identification of a UAV and design of a hardware-in-the-loop system for nonlinear control purposes. In AIAA Guidance, Navigation, and Control Conference and Exhibit (p. 6483).
53. Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. Neural networks, 2(5), 359-366.
54. San Martin, R., Barrientos, A., Gutierrez, P., & del Cerro, J. (2006, May). Unmanned aerial vehicle (UAV) modelling based on supervised neural networks. In Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006. (pp. 2497-2502). IEEE.
55. San Martin, R. S., Barrientos, A., Gutierrez, P., & Cerro, J. D. (2006, July). Neural networks training architecture for UAV modeling. In Automation Congress, 2006. WAC'06. World (pp. 1-6). IEEE.
56. Puttige, V., & Anavatti, S. (2008). Real-time system identification of unmanned aerial vehicles: a multi-network approach. Journal of computers, 3(7), 31-38.
57. Putro, I. E., Budiyono, A., Yoon, K. J., & Kim, D. H. (2009, February). Modeling of unmanned small scale rotorcraft based on neural network identification. In Robotics and Biomimetics, 2008. ROBIO 2008. IEEE International Conference on (pp. 1938-1943). IEEE.

58. Taha, Z., Deboucha, A., & Dahari, M. B. (2010, November). Small-scale helicopter system identification model using recurrent neural networks. In TENCON 2010-2010 IEEE Region 10 Conference (pp. 1393-1397). IEEE.
59. Lin, Q., Cai, Z., Wang, Y., Yang, J., & Chen, L. (2013, September). Adaptive flight control design for quadrotor UAV based on dynamic inversion and neural networks. In Instrumentation, Measurement, Computer, Communication and Control (IMCCC), 2013 Third International Conference on (pp. 1461-1466). IEEE.
60. Narendra, K. S., & Parthasarathy, K. (1990). Identification and control of dynamical systems using neural networks. Neural Networks, IEEE Transactions on, 1(1), 4-27.
61. Shamsudin, S. S., & Chen, X. (2012). Identification of an unmanned helicopter system using optimised neural network structure. International Journal of Modeling, Identification and Control, 17(3), 223-241.
62. Li, Y., Sundararajan, N., & Saratchandran, P. (2001). Neuro-controller design for nonlinear fighter aircraft maneuver using fully tuned RBF networks. Automatica, 37(8), 1293-1301.
63. Tijani, I. B., Akmeliawati, R., Legowo, A., & Budiyono, A. (2014). Nonlinear identification of a small scale unmanned helicopter using optimized NARX network with multi-objective differential evolution. Engineering Applications of Artificial Intelligence, 33, 99-115.
64. Hatamleh, K. S., Al-Shabi, M., Al-Ghasem, A., & Asad, A. A. (2015). Unmanned aerial vehicles parameter estimation using artificial neural networks and iterative bi-section shooting method. Applied Soft Computing, 36, 457-467.
65. Jin, H., Li, X., Zhang, X., & Li, Y. (2015, May). Study on the modeling methods of the quad-rotor unmanned helicopter. In The 27th Chinese Control and Decision Conference (2015 CCDC) (pp. 4835-4839). IEEE.
66. Puttige, V. R., & Anavatti, S. G. (2006, November). Real-time neural network based online identification technique for a UAV platform. In Computational Intelligence for Modeling, Control and Automation, 2006 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on (pp. 92-92). IEEE.
67. Salman, S. A., Puttige, V. R., & Anavatti, S. G. (2006, October). Real-time validation and comparison of fuzzy identification and state-space identification for a UAV platform. In Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control, 2006 IEEE (pp. 2138-2143). IEEE.
68. Santos, M., López, R., & de la Cruz, J. M. (2006). A neuro-fuzzy approach to fast ferry vertical motion modelling. Engineering Applications of Artificial Intelligence, 19(3), 313-321.
69. Badrzadeh, H., Sarukkalige, R., & Jayawardena, A. W. (2018). Intermittent stream flow forecasting and modelling with hybrid wavelet neuro-fuzzy model. Hydrology Research, 49(1), 27-40.
70. Nayak, P. C., Sudheer, K. P., Rangan, D. M., & Ramasastri, K. S. (2004). A neuro-fuzzy computing technique for modeling hydrological time series. Journal of Hydrology, 291(1-2), 52-66.
71. Nicol, C., Macnab, C. J. B., & Ramirez-Serrano, A. (2008, May). Robust neural network control of a quadrotor helicopter. In Electrical and Computer Engineering, 2008. CCECE 2008. Canadian Conference on (pp. 001233-001238). IEEE.
72. Sierra, J.E., Santos, M. (2013) Estudio comparativo de modelos de un vehículo aéreo obtenidos mediante técnicas analíticas y basadas en redes neuronales. In Actas

Multiconferencia CAEPIA'13 (XV Conferencia de la Asociación Española para la Inteligencia Artificial), pages 1270-1279, 2013. ISBN: 978-84-695-8348-7.
73. Sierra, J. E., Santos, M. (2015). Adaptive neural control-oriented models of unmanned aerial vehicles. Advances in Intelligent Systems and Computing, (pp. 329-337). Springer International Publishing. (10th International Conference on Soft Computing Models in Industrial and Environmental Applications). ISBN 2194-5357.
74. Sierra, J.E., Santos, M. (2014) Modelado de un vehículo aéreo no tripulado mediante combinación de técnicas paramétricas y neurodifusas. In Actas del XVII Congreso Español sobre Tecnologías y Lógica Fuzzy, pages 339-344, 2014. ISBN: 978-84-15688-76-1).
75. Sierra, J.E., Santos, M. (2013) Modelado de un vehículo aéreo no tripulado mediante aplicación conjunta de técnicas paramétricas y neuronales. In Actas del Congreso JJAA 2013, XXXIV Jornadas de Automática, pages 189-194, 2013. ISBN: 978-84-616-5063-7.
76. Sierra. J.E., Santos, M. (2018). Modelling engineering systems using analytical and neural techniques: hybridization. Neurocomputing, 271, 70-83.
77. Calise, A. J., and Rysdyk, R. T. (1998): Nonlinear adaptive flight control using neural networks. IEEE Control Systems Magazine,18(6), 14-25.
78. Kim, B. S., & Calise, A. J. (1997): Nonlinear flight control using neural networks. AIAA Journal of Guidance, Control, and Dynamics, 20(1),26-33.
79. Mo, H., & Farid, G. (2019). Nonlinear and adaptive intelligent control techniques for quadrotor UAV–A survey. Asian Journal of Control.
80. Efe, M. Ö. (2011). Neural network assisted computationally Simple $PI^{\lambda}D^{\mu}$ Control of a Quadrotor UAV. IEEE Transactions on Industrial Informatics, 7(2), 354-361.
81. Shepherd III, J. F., & Tumer, K. (2010, July). Robust neuro-control for a micro quadrotor. In Proceedings of the 12th annual conference on Genetic and evolutionary computation (pp. 1131-1138). ACM.
82. Nicol, C., Macnab, C. J. B., & Ramirez-Serrano, A. (2008, May). Robust neural network control of a quadrotor helicopter. In Electrical and Computer Engineering, 2008. CCECE 2008. Canadian Conference on (pp. 001233-001238). IEEE.
83. Cervantes, J., Muñoz, F., González-Hernández, I., Salazar, S., Chairez, I., & Lozano, R. (2017, June). Neuro-fuzzy controller for attitude-tracking stabilization of a multi-rotor unmanned aerial system. In Unmanned Aircraft Systems (ICUAS), 2017 International Conference on (pp. 1816-1823). IEEE.
84. Zhang, Y., Xu, B., & Li, H. (2015). Adaptive neural control of a quadrotor helicopter with extreme learning machine. In Proceedings of ELM-2014 Volume 2 (pp. 125-134). Springer, Cham.
85. Szanto, N., Narayanan, V., & Jagannathan, S. (2017, May). Event-sampled control of quadrotor unmanned aerial vehicle using neural networks. In 2017 American Control Conference (ACC) (pp. 2956-2961). IEEE.
86. Dierks, T., & Jagannathan, S. (2010). Output feedback control of a quadrotor UAV using neural networks. IEEE transactions on neural networks, 21(1), 50-66.
87. Rosales, C., Rossomando, F., Soria, C., & Carelli, R. (2018, June). Neural control of a quadrotor: A state-observer based approach. In 2018 International Conference on Unmanned Aircraft Systems (ICUAS) (pp. 647-653). IEEE.
88. Bouhali, O., & Boudjedir, H. (2011, June). Neural network control with neuro-sliding mode observer applied to quadrotor helicopter. In Innovations in Intelligent Systems and Applications (INISTA), 2011 International Symposium on (pp. 24-28). IEEE.
89. Boudjedir, H., Bouhali, O., & Rizoug, N. (2012). Neural network control based on adaptive observer for quadrotor helicopter. International Journal of Information Technology, Control and Automation, 2(3), 39-54.

90. Guo, L., & Cao, S. (2014). Anti-disturbance control theory for systems with multiple disturbances: A survey. ISA transactions, 53(4), 846-849.
91. Isidori A, Byrnes CI. Output regulation of nonlinear systems. IEEE Trans Autom Control 1990;35(2):131–40.
92. Marino, R., & Tomei, P. (1995). Nonlinear control design: geometric, adaptive and robust (Vol. 136). London: Prentice Hall.
93. Byrnes, C. I., Priscoli, F. D., & Isidori, A. (2012). Output regulation of uncertain nonlinear systems. Springer Science & Business Media.
94. Huang J, Chen Z. A general framework for tackling the output regulation problem. IEEE Trans Autom Control 2004;49(12):2203–18.
95. Xian, B., Jalili, N., Dawson, D. M., & Fang, Y. (2003). Adaptive tracking control of linear uncertain mechanical systems subjected to unknown sinusoidal disturbances. Transactions American Society of Mechanical Engineers Journal of Dynamics Systems Measurement and Control, 125(1), 129-133.
96. Xi ZR, Ding Z. Global adaptive output regulation of a class of nonlinear systems with nonlinear exosystems. Automatica 2007;43(1):143–9
97. Nikiforov V. Nonlinear servocompensation of unknown external disturbances. Automatica 2001;37(10):1647–53.
98. Bodson M, Jensen JS, Douglas SC. Active noise control for periodic disturbances. IEEE Trans Control Syst Technol 2001;9(1):200–5
99. Aguilar, C. O., & Krener, A. J. (2013). Patchy solution of a Francis–Byrnes–Isidori partial differential equation. International Journal of Robust and Nonlinear Control, 23(9), 1046-1061
100. Han JQ. From PID to active disturbance rejection control. IEEE Trans Ind Electron 2009;56(3):900–6.
101. Min, B.-C., Hong, J.-H., Matson, E.T.: Adaptive robust control (ARC) for an altitude control of a quadrotor type UAV carrying an unknown payloads. In: 2011 11th International conference on control, automation and systems Korea, pp. 26–29 (2011)
102. Wang, C., Nahon, M., Trentini, M., Nahon, M., Trentini, M.: Controller development and validation for a small quadrotor with compensation for model variation. In: 2014 International conference on unmanned aircraft systems (ICUAS). IEEE (2014)
103. Wang, C., Song, B., Huang, P., & Tang, C. (2016). Trajectory tracking control for quadrotor robot subject to payload variation and wind gust disturbance. Journal of Intelligent & Robotic Systems, 83(2), 315-333.
104. Yüksel, B., Secchi, C., Bülthoff, H. H., & Franchi, A. (2014, July). A nonlinear force observer for quadrotors and application to physical interactive tasks. In Advanced Intelligent Mechatronics (AIM), 2014 IEEE/ASME International Conference on (pp. 433-440). IEEE.
105. Escareño, J., Salazar, S., Romero, H., & Lozano, R. (2013). Trajectory control of a quadrotor subject to 2D wind disturbances. Journal of Intelligent & Robotic Systems, 70(1-4), 51-63.
106. Alexis, K., Nikolakopoulos, G., & Tzes, A. (2011). Switching model predictive attitude control for a quadrotor helicopter subject to atmospheric disturbances. Control Engineering Practice, 19(10), 1195-1207.
107. Cabecinhas, D., Cunha, R., & Silvestre, C. (2015). A globally stabilizing path following controller for rotorcraft with wind disturbance rejection. IEEE Transactions on Control Systems Technology, 23(2), 708-714.
108. Besnard, L., Shtessel, Y. B., & Landrum, B. (2012). Quadrotor vehicle control via sliding mode controller driven by sliding mode disturbance observer. Journal of the Franklin Institute, 349(2), 658-684.

109. Sierra. J.E., Santos, M. (2019). Wind and payload disturbance rejection control based on adaptive neural estimators: application on quadrotors. Complexity, Article ID 6460156, 20 pages.
110. Agarwal, M. (1997). A systematic classification of neural-network-based control. IEEE Control Systems, 17(2), 75-93.
111. Sierra, J. E., Santos, M. (2017). Control de un vehículo cuatrirrotor basado en redes neuronales. Actas de las XXXVIII Jornadas de Automática, pp 431-436. ISBN: 978-84-16664-74-0.
112. King, S. Y., & Hwang, J. N. (1989). Neural network architectures for robotic applications. IEEE Transactions on Robotics and Automation, 5(5), 641-657.
113. Sierra, J. E., & Santos, M. (2018, June). Disturbances based adaptive neuro-control for UAVs: A first approach. In The 13th International Conference on Soft Computing Models in Industrial and Environmental Applications (pp. 293-302). Springer, Cham.
114. NASA Glenn Research Center.(2019). https://www.grc.nasa.gov/WWW/K-12/airplane/ldrat.html
115. Royal Meteorological Society, (2019): https://www.rmets.org/weather-and-climate/observing/beaufort-scale

III Publicaciones

1 Modelling engineering systems using analytical and neural techniques: hybridization

BIBLIOGRAPHIC DESCRIPTION

Title

Modelling engineering systems using analytical and neural techniques: hybridization.

Citation

Sierra. J.E., Santos, M. (2018). Modelling engineering systems using analytical and neural techniques: hybridization Neurocomputing, 271, 70-83.

Abstract

From real input/output data, different control-oriented models of a quadrotor unmanned aerial vehicle (UAV) are obtained by applying different identification methods. Parametric techniques, neural networks, neuro-fuzzy inference systems, and the hybridization of some of them are applied. The identified models are analysed and compared in the time and frequency domains. We conclude that the hybridization of analytical and intelligent techniques is a good choice to model of complex systems while keeping a good balance between accuracy and computational cost. In addition, off-line trained neural networks and adaptive networks with on-line learning are analysed, and their advantages and disadvantages regarding modelling are presented. The influence of the partition of the training and validation dataset on the model error is also discussed.

References

Zhu, Quanmin, et al (2015); Hybertson, Duane W. (2016); Puccia, Charles J. et al (2013); Goodwin, G. C. et al (1977);. Ugalde, H. M. R. et al (2013); Santos, M. et al (2006); Weng, L. et al (2014); Santos, M. et al (2010); Marichal, N. et al (2012); García-Auñón, P. et al (2016); Menoyo-Larrazabal, J. et al (2016); Estellés-Martínez, S et al (2014); Hoffer, N. V. et al (2014);. Gessow, A. et al. (1952); Bramwell, A. R. S. (1976).; Cai, G. et al (2008). Budiyono, A. et al (2009); El-Saadany, A. et al (2009); del-Cerro, J. et al (2009); Castillo, P. et al (2005); Wu, H. et al. (2004); Morris, J. C. et al (1994). Mettler, B. et al (1999); Tischler, M. B. et al (1992); Tischler, M. B. (1996); Ahmad, S. M. et al. (2000); Hashimoto, S. et al (2000);

Yuan, W. et al (2011); Manaı, M. et al (2005); Hornik, K. et al (1989); San Martin, R. et al. (2006); San Martin, R. et al (2006); Puttige, V. et al (2008); Putro, I. E. et al (2009); Taha, Z. et al (2010); Lin, Q. et al (2013); Narendra, K. S. et al (1990); Shamsudin, S. S. et al (2012); Li, Y. et al (2001); Tijani, I. B. et al (2014); Hatamleh, K. S. et al (2015); Jin, H. et al (2015); Puttige, V. R. et al (2006); Salman, S. A. et al. (2006); Mahfouz, M. et al (2013); Struik, D. J. (2014); Walter, E. et al (1997); Park, J. et al (1991); Jang, J. S. R. (1993); Sierra, J. E. et al (2015)

Impact Factor

Neurocomputing has an impact factor of 3.241 according to Thomson Reuters Journal Citation Reports (2017).

Neurocomputing 271 (2018) 70–83

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Modelling engineering systems using analytical and neural techniques: Hybridization

J. Enrique Sierra[a,b,*], Matilde Santos[a]

[a] Computer Science Faculty, Complutense University of Madrid, C/Profesor García Santesmases 9, 28040 Madrid, Spain
[b] Department of Civil Engineering, Area of Languages and Computer Systems, University of Burgos, Av. Cantabria, 09006 Burgos, Spain

ARTICLE INFO

Article history:
Received 6 May 2016
Revised 15 September 2016
Accepted 3 November 2016
Available online 6 July 2017

Keywords:
Identification
Adaptive neural networks
Neuro-fuzzy
Parametric techniques
Hybridization
Unmanned aerial vehicles (UAV)

ABSTRACT

From real input/output data, different control-oriented models of a quadrotor unmanned aerial vehicle (UAV) are obtained by applying different identification methods. Parametric techniques, neural networks, neuro-fuzzy inference systems, and the hybridization of some of them are applied. The identified models are analyzed and compared in the time and frequency domains. We conclude that the hybridization of analytical and intelligent techniques is a good choice to model of complex systems while keeping a good balance between accuracy and computational cost. In addition, off-line trained neural networks and adaptive networks with on-line learning are analyzed, and their advantages and disadvantages regarding modelling are presented. The influence of the partition of the training and validation dataset on the model error is also discussed.

1. Introduction

Modelling is one of the most important approaches to deal with complex systems. Any representation of a real system allows us to analyze its behaviour and to get a deeper knowledge of it. In the area of engineering control, the availability of simulation models is essential in order to design control laws and to test them without endangering the real system. Therefore, one of the most important steps to effectively control a physical system is to obtain reliable and useful models that describe its behaviour [1].

The importance of having a good model grows with the complexity of the system. On the one hand, the model must be as close as possible to reality but, on the other hand, the model should be kept as simple as possible in order to reduce the computational cost. Low computational cost is a key enabling factor to run the model in low-cost general-purpose hardware platforms [2]. The underlying idea of our proposal is to obtain models of complex systems by keeping an adequate balance between accuracy and the hardware resources needed to execute them.

Basically, there are two ways of approaching the modelling: the first one is to apply the well-known physical equations that govern the dynamic behaviour of the real system, and use this knowledge to generate the usually non-linear high-order differential equations that represent it. This methodology is straightforward when all variables are known, and the relations between them are relatively simple. However, these mathematical models are usually difficult to handle because of the nonlinear nature of the equations. Even more, there are real systems which are too complex to be modelled this way or it is even impossible to obtain an analytic expression of its dynamics. In this case, it is necessary to apply other approaches [3].

The second way identifies the model using experiments to obtain representative input and output data of the real system. Analyzing this information by the application of identification strategies, it is possible to estimate the internal relationship between inputs and outputs [4]. Then mapping mechanisms are applied to represent the hidden internal relations between inputs and outputs. These mapping procedures iteratively update their parameters to tune the model as well as to reduce its order [5]. The system is then considered as a black-box because there is not physical knowledge about the system represented in the model. Within this approach, called identification, parametric and intelligent methodologies are found. Among the latter, soft computing techniques, such as neural networks and neuro-fuzzy systems are included. These methods, closer to human learning, have been proved better when dealing with strongly non-linear and complex systems, such as marine and aerial autonomous vehicles [6–10].

Parametric techniques provide simple models but whose accuracy may not be highlighted. On the other hand, methods based

* Corresponding author at: Department of Civil Engineering, Area of Languages and Computer Systems, University of Burgos, Av. Cantabria, 09006 Burgos, Spain.
E-mail addresses: jesierra@ubu.es, jesier01@ucm.es (J.E. Sierra), msantos@ucm.es (M. Santos).

http://dx.doi.org/10.1016/j.neucom.2016.11.099
0925-2312/

on artificial intelligence generate models which better fit the real behaviour of the system but can be very demanding in terms of computational requirements.

These ideas suggest that a good choice would be to unite precision and simplicity in a single strategy. Following this line of thought, this paper aims to combine both types of techniques in order to get the best of both worlds, the straightforwardness of the parametric models and the accuracy of soft computing techniques. First, we have applied parametric techniques (ARX, ARMAX, and OE) independently, with different configuration. Then various types of neural networks (radial basis, feedforward), and neuro-fuzzy systems have been used to identify the UAV model. Finally, the hybridization of both approaches, the parametric and the intelligent ones, is proposed and the advantages of this combination are discussed.

Besides, we have also analyzed how the training dataset influences the obtained model when supervised neural networks are applied. In fact, off-line learning neural networks have the disadvantage of being very sensitive to the partition of the training dataset. On the contrary, adaptive neural networks with online learning are much more robust because they are continuously learning from the real data of the system. It is not necessary to select any dataset to train the network and therefore it is not biased. In addition, they can be used to identify models of systems whose dynamic changes over time, as for example, systems that work in outdoor environments subject to changing weather conditions [11].

In addition, this paper provides another original contribution to the system identification field: the definition of a new criterion to measure and analyze the performance of the models. This criterion is a hybrid Mean Square Error (MSE) computed by the combination of the estimated step response error in the time domain and the estimated frequency response error.

We are working with a real unmanned aerial vehicle (UAV), the Ardrone 2.0 (Parrot). This system is complex and strongly non-linear. Using real input/output data, different models of the quadrotor have been obtained by applying system identification based on neural, adaptive neural and neuro-fuzzy networks, traditional parametric methods, and the combination of these strategies. The resulting models have been compared in the time and frequency domains in terms of MSE. In today's world, UAVs' modelling is gaining importance because of their potential applications in different sectors such as defence, security, construction, agriculture, fire fighting, environment, entertainment, etc. These applications demand to get the most from these devices, and a necessary step to reach this target is to obtain models which help us to design smart controllers to stabilize the vehicle motion.

The paper is organized as follows: Section 2 presents a brief review of the literature related to the modelling of unmanned aerial vehicles, both the parametric approach and the intelligent approach. Section 3 describes the materials and methods, that is: the description of the UAV; the parametric and intelligent techniques applied; and the evaluation criteria used to rate the methods. The results of every identification technique and the influence of the on-line/off-line learning algorithm on the modelling are discussed in Section 4. The paper ends with the conclusions and future works.

2. Background

2.1. UAV identification by analytical techniques

In recent years, the research and development of unmanned aerial vehicles have gained much attention in academic communities around the world [12]. Focusing on modelling, different strategies can be found in the scientific literature, from analytical models based on physical equations representing the dynamic behaviour of the system up to intelligent models based on soft computing techniques [13].

One of the first analytical models of the dynamic behaviour of a helicopter was proposed by Gessow and Myers in 1952 [14]. Since them, many contribution can be found, first on helicopters as the work by Bramwell in 1976 [15] and then for any type of UAV. Analytic models of helicopters also appear in Cai et al. [16], Budiyono et al. [17] and El-Saadany et al. [18]. del Cerro et al. [19] present a hybrid model, analytical and empirical, where the parameters have been calculated by genetic algorithms. A very complete survey of mathematical models and control strategies for different types of aerial vehicles is presented in Castillo et al. [20].

Among these mathematical approaches, several authors obtain models of different UAVs by the application of parametric error techniques. Under some soft assumptions, a discrete-time multivariable non-linear stochastic system can be represented by the multi variable NARMAX model. For example, Wu et al. [21] applies ARX to identify the model of a micro air vehicle in loitering flight based on attitude performance evaluation. Morris et al. [22] and Cai et al. [16] use parametric error models based on state space to identify either a whole vehicle or a subsystem model, as for example the yaw motion. The model proposed by Mettler [23] is widely cited by other authors as a complex but effective model. Another parametric model of a helicopter is obtained applying a frequency domain method developed by the U.S. Army and NASA, known as Comprehensive Identification from Frequency Responses (CIFER) [24]. While CIFER was specifically developed for rotorcrafts, it has been successfully used in a wide range of fixed wing and rotary wing vehicles, and also in unconventional aircraft applications [25].

Other authors also use classical parametric structures to identify the model but then apply advanced strategies in order to tune the values of the model parameters. Some of them apply soft computing techniques for the tuning. For example, Ahmad uses the parametric model NARX (Nonlinear ARX) and he applies neural networks to adjust the parameters [26]. Another example is offered by del Cerro [19] where the tuning is done by genetic algorithms.

Hashimoto et al. [27] compares the performance of the identification models of an unmanned helicopter obtained by ARX, ARMAX, OE and BJ parametric structures. In our study we have also applied ARX, ARMAX and OE but, unluckily, Hashimoto does not provide quantitative results; therefore it has not been possible to compare our results with his models. Yuan and Katupitiya [28] apply the OE method to estimate the model of an unmanned helicopter. Manai [29] also uses OE structure for the identification of an UAV; in this case the purpose of the study is the design of controllers. Our paper is also aimed to design control oriented UAV models, but in our work we present a hybridization technique which is able to improve the accuracy while reducing the computational cost. Besides, we have also defined some figures to quantitatively evaluate the performance of the models.

2.2. UAV identification by neural networks and neuro-fuzzy inference systems

Some theoretical works have rigorously proved that, even with only one hidden layer, neural networks are universal function approximators [30]. Thus modelling non-linear systems by neural networks is a good alternative.

Among the application of neural networks to UAVs modelling, MLP and RBF networks are the most used to simulate the pitch, roll, yaw, and the position of the unmanned vehicles [31–38]. A RBF network is also used in [39]. Neural networks based on Nonlinear Auto-Regressive models with exogenous inputs (NARX models) have been also adopted due to its proven good performance especially for nonlinear complex systems [40]. Nevertheless, in this

Fig. 1. UAV (unmanned aerial vehicle), Parrot [46].

paper there is not a proper hybridization as those techniques are not used together.

In [41], three different methods are used to obtain the parameters of a quadrotor model: the Iterative Bi-Section Shooting method, IBSS; an artificial neural network, ANN; and a hybrid ANN–IBSS. In this paper, as in our case, the hybrid method provides higher accuracy than the rest of methods individually applied. Another hybridization technique is proposed by Jin [42], in this case combining RBF and ARX. Puttige and Anavatti [43] approach the modelling with a multi-network architecture by the combination of offline and online methods.

Fuzzy logic has been also applied to the identification of UAVs. There are some works that could be highlighted, such as the one by Salman, who obtains the UAV model applying states space modelling and fuzzy logic identification, and compares them [44]. Other examples of application of soft computing techniques (neuro-fuzzy) to model complex non-linear systems can be found, for example, in [6,45].

3. Materials and methods

3.1. Description of the system

The identification of an UAV is a complex issue. Its difficulty comes from several sources, such as the uncertainty of the airstreams and the exogenous forces, its tendency to be unstable, the inhomogeneity of the internal mechanical structure, and the strongly non-linear system dynamic. All of these characteristics led us to consider this physical system as our benchmark.

We work with the unmanned aerial vehicle ARDRONE 2.0, by Parrot (Fig. 1). This system is carbon fibber and plastic PA66. It is equipped with four high efficiency brushless motors, powered by one 11 V 1000 mAh lithium battery, which gives up to 12 min flight autonomy. It weighs 380 g without casing and 420 g with casing. It has a three-axis accelerometer, a two-axis gyroscope (pitch and roll), and a high precision gyroscope for the yaw angle [46].

Fig. 2 shows the coordinate system of the UAV. The pitch is controlled by setting the speed of the longitudinal motors. If the speed of the rotors 1 and 2 increases regarding rotors 3 and 4, the vehicle pitches down and the lift forces cause the vehicle to move along the x-axis direction (forward movement). If the speed of the rotors 1 and 2 decrease regarding rotors 3 and 4, the vehicle pitches up and the lift forces cause the vehicle to move backwards. On the other hand, the roll control is carried out by changing the speed of the lateral motors. If the speed of rotors 1 and 4 increases the vehicle moves along the direction of the y-axis (left-side movement). If the speed of rotors 1 and 4 decreases regarding rotors 2 and 3, the vehicle rolls and the lift forces cause the vehicle to move toward right.

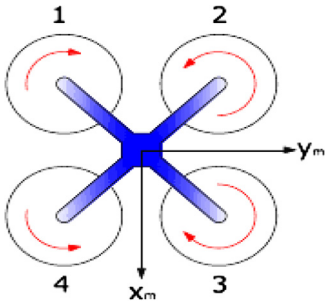

Fig. 2. Coordinate system of the UAV.

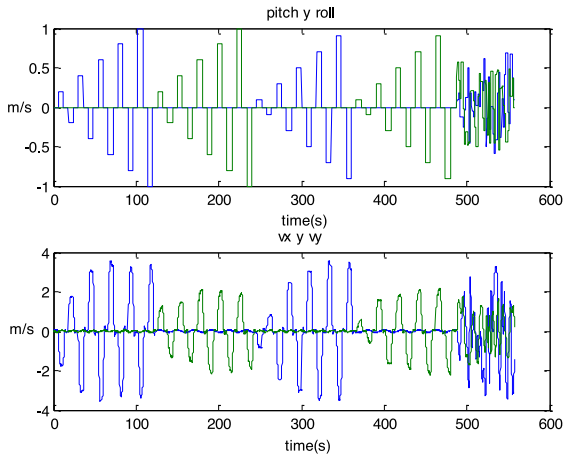

Fig. 3. Pitch and roll (inputs) and velocity (output). (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)

The input and output signals of the UAV are shown in Fig. 3. They have been provided by the Control Engineering Group of the Spanish Committee of Automatic Control (CEA) [47]. The pitch signal and the velocity in the x-axis, v_x, are shown in blue meanwhile the roll and the velocity in the y-axis, v_y, are represented in green. As it may be seen in the figure, the input signals consist of a train of steps with different amplitude. This kind of signals is traditionally used in identification to extract the maximum possible information from the system.

These signals will be used as targets by the identification techniques. They have been measured in the real system, during 9 min of flight, sampled with a time period of 60 ms [47].

As it was said, the necessity of applying advanced techniques in order to obtain useful models of the UAV, or to determine its parameters, is justified by the complexity of the system. Its dynamics can be represented by the following equations [42]:

$$\dot{v}_b = \begin{bmatrix} -\sin\theta g \\ \sin\phi\cos\theta g \\ -\frac{T}{m} + \cos\phi\cos\theta g \end{bmatrix} \tag{1}$$

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{L}{I_x} \\ \frac{M}{I_y} \\ \frac{N}{I_z} \end{bmatrix} + \begin{bmatrix} \frac{rqI_y - qrI_z}{I_x} \\ \frac{-rpI_z + prI_z}{I_y} \\ \frac{qpI_x - pqI_y}{I_z} \end{bmatrix} \tag{2}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3}$$

Where ϕ is the roll angle, θ is the pitch angle, ψ is the yaw angle, all of them in rad; p, q and r, are the roll, pitch and yaw velocity, respectively (rad/s). The roll, pitch and yaw torques are L, M,

and N, respectively (N m); I_x to I_z are the inertias in x-axis to z-axis, respectively, in kg m^2. The gravity acceleration is ρ and T is the lift force, in N. These expressions are strongly non-linear and the variables of the system are coupled, but the real model would even be more complex because in these equations the aerodynamic forces, random external forces, inhomogeneity parts, etc., have not been taken into account.

3.2. Evaluation criteria

There are two kinds of indicators that we are going to use to evaluate the models. On the one hand, the system response in the time and frequency domain will be used, that is, the similarity of the model with the available real data. On the other hand, we will estimate the computational time required by the identification algorithms.

3.2.1. Time and spectral analysis

A complex system like the one mentioned above demands a clear procedure to analyze its performance. In order to compare the models we think it is not enough to only measure the error between the models' output and the real system output. The behaviour of a system is also determined by the step response in the time domain and the frequency response. Even more, the step response allows us to estimate the degree of coupling between the system variables.

Therefore, we first need to define the correlation and the spectral analysis. Once the estimation of the covariance between the inputs and outputs and the variance of the input are obtained, the impulse response can be estimated by the following expressions:

$$\hat{R}^N_{yu}(\tau) = \frac{1}{N}\sum_{t=1}^{N} y(t)u(t-\tau) \tag{4}$$

$$\hat{\lambda}^N = \frac{1}{N}\sum_{t=1}^{N} u(t)^2 \tag{5}$$

$$\hat{g}^N(\tau) = \frac{\hat{R}^N_{yu}(\tau)}{\hat{\lambda}^N} \tag{6}$$

$$\hat{S}^N(\tau) = \sum_{t=1}^{\tau} \hat{g}^N(t) \tag{7}$$

Where y is the system output, u is the input, $\hat{R}^N_{yu}$ is the estimated covariance, $\hat{\lambda}^N$ is the estimated variance, $\hat{g}^N$ is the estimated impulse response, and $\hat{S}^N$ is the estimated step response. Fig. 4 shows the step response of the UAV, for v_x (top row) and v_y (bottom row) velocities, pitch (left) and roll (right) angles.

In Fig. 4, it is possible to see that the stationary gain of the elements in the diagonal is bigger than in the rest of elements. This fact indicates that the system dynamic is partially decoupled and thus the MIMO (multi-input multi-output) system may be converted into two SISO (single-input single-output) systems. Considering this simplification, we will focus on the how the pitch influences the velocity in the x-axis and the roll affects velocity in the y-axis.

Using the estimated covariance (4), the frequency response is calculated using the Fourier transform weighed by a windowing function, as described by the following expressions:

$$\hat{\Phi}^N_{yu}(\omega) = \sum_{\tau=-M}^{M} W_M \hat{R}^N_{yu}(\tau)e^{-i\omega\tau} \tag{8}$$

$$\hat{\Phi}^N_{u}(\omega) = \sum_{\tau=-M}^{M} W_M \hat{R}^N_{u}(\tau)e^{-i\omega\tau} \tag{9}$$

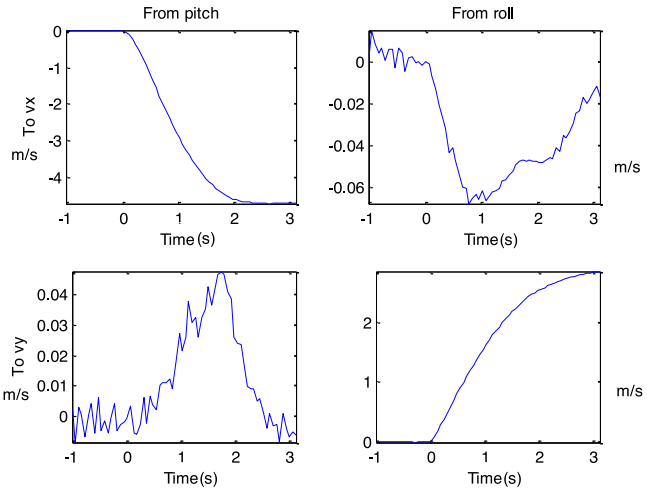

Fig. 4. Step response of the MIMO system.

$$\hat{G}^N(e^{i\omega T}) = \frac{\hat{\Phi}^N_{yu}(\omega)}{\hat{\Phi}^N_{u}(\omega)}. \tag{10}$$

Where $\hat{\Phi}^N_{yu}(\omega)$ is the windowed Fourier transform of the covariance, $\hat{\Phi}^N_{u}(\omega)$ is the windowed Fourier transform of the discretized input, $\hat{G}^N(e^{i\omega T})$ is the estimated frequency response and T is he sampling time.

Fig. 5 shows the frequency response obtained by the procedure described in Eqs. (8)–(10). It is possible to see how the cutoff frequency of the system is low, around 2 rad/s. The exact values are 1.74 rad/s for the x-axis and 1.15 rad/s for the y-axis.

3.2.2. Error indicators

With the correlation and spectral analysis previously estimated, we are ready to define the indicators which will be used to analyze the performance of the different identification techniques. These indicators are the mean square errors given by the following expressions:

$$\mathrm{MSE}_T = \frac{1}{N_T}\sum_{t=1}^{N_T} (y_m(t) - y_r(t))^2 \tag{11}$$

$$\mathrm{MSE}_S = \frac{1}{N_S}\sum_{t=1}^{N_S} \left(\hat{S}^N_m(t) - \hat{S}^N_r(t)\right)^2 \tag{12}$$

$$\mathrm{MSE}_F = \frac{1}{2N_F}\sum_{f=1}^{N_F} \left(\left|\hat{G}^N_m(f)\right|^2 - \left|\hat{G}^N_r(f)\right|^2\right) \tag{13}$$

Where the subscript m means model and it indicates that the value has been obtained using the data from the modelled output signal; r means real, indicating that the value was taken from the real output signal. The subscript T refers to the values of the temporal sequence, S to values of the step response, and F represents the values of the frequency response.

In order to combine these indicators, first we have to check if these figures share similar properties and the combination has dimensional consistency. It is trivial to see that MSE_T represents the power of the error signal:

$$y_e = y_m - y_r. \tag{14}$$

Besides, $\hat{S}^N_m(t)$ as well as $\hat{S}^N_r(t)$ are the step response signals. Therefore MSE_S will be the power of the error signal:

$$\hat{S}^N_e(t) = \hat{S}^N_m(t) - \hat{S}^N_r(t). \tag{15}$$

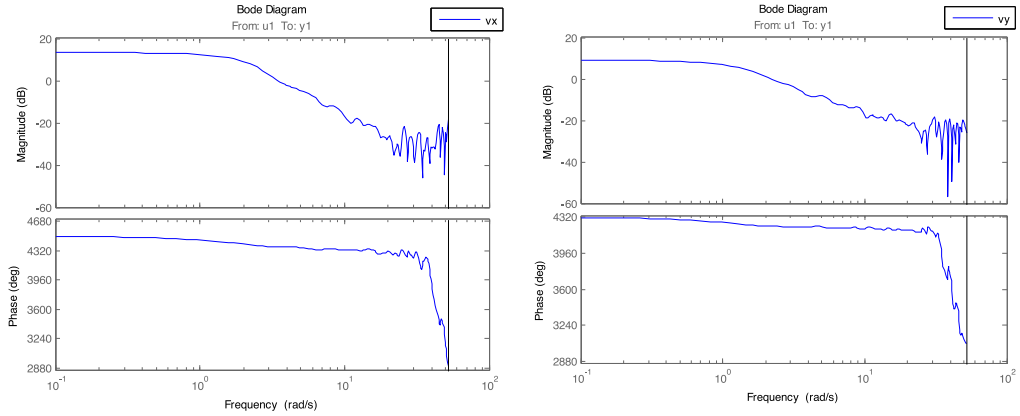

Fig. 5. Frequency response of the system (v_x on the left and v_y on the right).

In the case of MSE_F, $|\hat{G}_m^N(f)|$ is the gain of the output sinusoidal signal of the system, when a sinusoidal signal of unitary amplitude and frequency f is introduced as input. The power of the sinusoidal signal is given by

$$P = \frac{A^2}{2} \tag{16}$$

being A is the amplitude. Taking into account the last equation the gain or loss of power due to the application of the model is:

$$\frac{1}{2}\left(\left|\hat{G}_m^N(f)\right|^2 - \left|\hat{G}_r^N(f)\right|^2\right) \tag{17}$$

Therefore, MSE_F is the average of this difference when a sweep frequency is carried out. At this point, it has been proved that the types of MSE proposed are compatible and can be combined:

$$MSE_C = \tfrac{1}{3}(MSE_T + MSE_S + MSE_F). \tag{18}$$

3.2.3. Computational complexity

We may decompose any algorithm into elemental operations. In our case these operations are: addition, subtraction, multiplication, division, exponentiation and the exponential function. In order to estimate the complexity of each identification technique, an indicator of the computational cost that combines the contribution of each type of operation has been generated.

The computational cost of the addition and subtraction is set to one unit. That is, the cost unit is defined as the time required to computing an addition. The cost of the multiplication can be considered the same than the cost of the division. The ratio between the cost of an addition and the cost of a multiplication depends on the electronics and hardware of the device (microprocessor, microcontroller, DSP…) used to execute the code and on the algorithm itself. For example, if the device has a dedicated multiplication unit in its ALU, as in DSPs, the cost will be much lower than if the multiplication is carried out using additions. The relation between these costs will be represented by the constant K_M (always equal or greater than 1).

In the same way, some proportional relations have been established for the rest of operations: K_Φ for the evaluation of the exponential function, and K_E for the exponentiation operation. In our case, the cost will be estimated taken into account only the algorithm used to compute the operation, and not the hardware of the device. The following expressions summarize the cost of the each operation in units of cost:

$$C_A = 1 \tag{19}$$

$$C_i = K_i \cdot N_i,\ i = M, E, \Phi. \tag{20}$$

Where C indicates time cost. The subscript indicates the operation: A for addition and subtraction, M for multiplication and division, Φ for the evaluation of the exponential function, and E for the exponentiation.

In order to determine the value of the K_i constants, we apply the concept of time complexity order when performing computations on a multitape Turing machine. Since an algorithm performance time may vary for different inputs even of the same size, the worst-case time complexity of an algorithm, denoted as $T(n)$, is used, which is defined as the maximum time for any input of size n. Time complexity is given by the function $T(n)$.

The complexity of a linear time algorithm is $T(n) = O(n)$, therefore an addition/subtraction will give $T(n) = O(n)$. For a multiplication/division, it depends on the algorithm applied to calculate it [48]; for the basic case is $T(n) = O(n^2)$, but for the Karatsuba's algorithm would be $T(n) = O(n^{1.585})$], for the 3-way Toom–Cook algorithm is $T(n) = O(n^{1.465})$, and so on for the Schönhage–Strassenh's, Fürer's, etc. The complexity of the exponential function developed as a Taylor series is $T(n) = O(n^{0.5} \cdot M(n))$, where $M(n)$ is the time complexity for the multiplication. The exponentiation complexity is $T(n) = O(k \cdot M(n))$, where k is the number of digits of the exponent. Assuming a Toom–Cook algorithm for the multiplication, the K constants may be calculated by the following expressions:

$$K_M = \frac{n^{1.465}}{n} = n^{0.465} \tag{21}$$

$$K_\Phi = n^{\frac{1}{2}} \cdot n^{0.465} = n^{0.965} \tag{22}$$

$$K_E = k \cdot n^{0.465}. \tag{23}$$

Most of the industrial embedded controllers are based on 16-bit microcontrollers, therefore we can assume $n = 16$, and $k = 16$. Hence, the following values are obtained: $K_M = 3.63$, $K_\Phi = 14.52$, $K_E = 58.08$. These K values will be used to compare the complexity between the different identification techniques.

3.3. Description of the techniques

3.3.1. Parametric techniques

Analytic models can been obtained using parametric techniques for the identification [49]. In this work, ARX, ARMAX and OE structures have been used under different configuration to identify the UAV model. Autoregressive with Exogenous Input (ARX) model is represented by the following expression:

$$A(q,\theta)y(t) = B(q,\theta)u(t - nk) + e(t), \tag{24}$$

where $y(t)$ is the output, $u(t)$ the input, $e(t)$ the error, A and B are polynomials with parameters defined by θ and nk is the delay. The first term of the equation, $A(q,\ \theta)y(t)$, is an auto-regression and $B(q,\theta)u(t-nk)$ is the contribution of the exogenous input u. Autoregressive Moving Average with Exogenous Input (ARMAX) models are described by a similar equation, where $C(q,\ \theta)$ is also a polynomial,

$$A(q,\theta)y(t) = B(q,\theta)u(t-nk) + C(q,\theta)e(t) \tag{25}$$

The Output Error (OE) models are given by the following expression:

$$y(t) = \frac{B(q,\theta)}{F(q,\theta)}u(t-nk) + e(t), \tag{26}$$

where $F(q,\ \theta)$ is also a polynomial.

In order to compute the computational complexity of each technique, the number of basic operations for each algorithm is obtained (Eqs. (27)–(29)). The number of adds/subtracts is N_A, and N_M represents the number of multiplications/divisions. The order of each polynomial is given by n_a, n_b, n_c, n_f, the order of the polynomial $B(q,\ \theta)$ incremented by 1 is represented by n_b,

$$\text{ARX}: \quad \boldsymbol{N_A = n_a + n_b - 1} \quad \boldsymbol{N_M = n_a + n_b} \tag{27}$$

$$\text{ARMAX}: \ \boldsymbol{N_A = n_a + n_b + n_c} - 1 \quad \boldsymbol{N_M = n_a + n_b + n_c} \tag{28}$$

$$\text{OE}: \quad \boldsymbol{N_A = n_b + n_f} - 1 \qquad \boldsymbol{N_M = n_b + n_f}. \tag{29}$$

3.3.2. Neural networks

Supervised neural networks have been successfully applied to model complex systems [37]. In this study, several topologies of neural networks have been tested. On the one hand, backpropagation multilayer feedforward networks have been selected due to their well-known properties as universal approximators. The influence of the on-line/off-line learning will be shown on these kinds of networks. They consist of several layers of neurons connected feedforward; a different activation function can be assigned to each layer of neurons. Each neuron in a particular layer is connected to all nodes in the next layer (full connection). The connection between the ith and jth neurons (two consecutive layers) is characterized by the weight coefficient, w_{ij}, and the ith neuron by the threshold coefficient b_i and so on. The output of the ith neuron is given by

$$x_i = f\left(b_i + \sum_{j=1}^{N} w_{ij}x_j\right) \tag{30}$$

Where f is the activation function and N is the number of neurons in the previous layer. In order to reduce the computational cost, the number of hidden layers of the network has been fixed to one. In this hidden layer, the activation function was set to the hyperbolic tangent sigmoid; the linear transfer function was assigned to the input and output layers. The training algorithm used was Levenberg–Marquardt; gradient descent was used as learning algorithm, and the performance function was set to the mean square error.

In addition, Radial Basis Function Networks have been also applied due to its well-known ability to approximate functions, as it was proved by Park and Sandberg [50]. This type of neural network has three layers; the input layer which receives the signals from the exterior, the hidden layer connected with the input layer and where each element is associated to a radial basis function and the output layer where the results of the activation functions of the hidden layer are linearly weighted. The output of each element of the hidden layer is calculated as the distance between the input pattern $x(n)$ and the centre c_i of the cluster weighted by

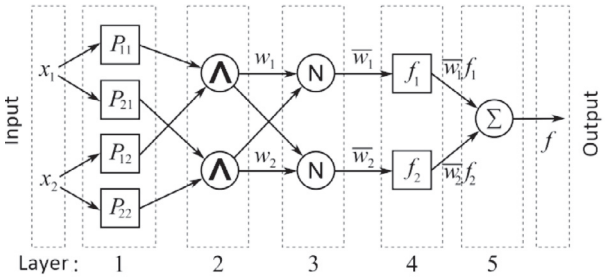

Fig. 6. Layer structure of ANFIS topology.

d_i. This value is applied to a radial basis function according to the following expression:

$$z_i(n) = \Phi\left(\frac{\sqrt{\sum_{j=1}^{N}\left(x_j(n)-c_{ij}\right)^2}}{d_i}\right), \ i = 1,\ldots\ldots L \tag{31}$$

Where N is the number of elements of the input pattern $x(n)$ and the basis radial function is defined by

$$\Phi(n) = e^{-n^2}. \tag{32}$$

Output values of the hidden layer are linearly weighted to give the final output of the network:

$$y_k(n) = \sum_{i=1}^{L} w_{i,k}z_i(n) + \mu_k, \ k = 1,\ldots,K. \tag{33}$$

Where the number of neurons in the hidden layer is L, K is the number of nodes in the output layer, $w_{i,\ k}$ are the weights of the output layer and μ_k are the activation thresholds.

The complexity of the Radial neural network is calculated as follows. If the number of exponential functions is N_Φ, the number of operations associated to the computation of one neuron in the hidden layer is:

$$N_{A_{hn}} = 2N - 1 \ \ N_{M_{hn}} = N + 1 \ \ N_{\Phi_{hn}} = 1. \tag{34}$$

The number of operations associated to an output node is equal to the number of neurons in the hidden layer, so:

$$N_{A_{on}} = L \qquad N_{M_{on}} = L \tag{35}$$

Taking into account that there are L neurons in the hidden layer and K outputs nodes, the number of operations of the whole network is:

$$N_A = N_{A_{hn}} \cdot L + N_{A_{on}} \cdot K \quad N_M = N_{M_{hn}} \cdot L + N_{M_{on}} \cdot K \ \ N_\Phi = N_{\Phi_{hn}} \cdot L \tag{36}$$

$$N_A = (2N-1)\cdot L + L\cdot K \quad N_M = (N+1)\cdot L + L\cdot K \ \ N_\Phi = L. \tag{37}$$

3.3.3. Neuro-fuzzy inference system

Finally, (Adaptive Neural Fuzzy Inference System (ANFIS) [51] has been also applied. Fig. 6 shows the structure of layers of an ANFIS system. It is based on the generation of a decision tree to classify data between p^n regression models, where the number of inputs is n and the number of partitions of each variable is p. These networks are composed by 5 layers. In the input layer, also named value layer, the inputs are fuzzified. This layer is composed by p^n nodes, where the output of each node is given by

$$O_{1,i} = \mu_{A,j}(x_k), \ \ i = 1,\ldots,\ldots p^n, \ \ j = 1,\ldots p, \ \ k = 1,\ldots\ldots,n \tag{38}$$

Where A is a linguistic label (small, big...), and the membership function μ is given by

$$\mu_A(x_k) = \frac{1}{1+\left|\frac{x_k - c_i}{a_i}\right|^{2b}} \tag{39}$$

With the outputs of the nodes in the first layer, the fuzzy rules are fired in the second layer (rules layer) using some T-norm (minimum, product...):

$$O_{2,i} = w_i = T\left(\mu_{A,j},\ \mu_{B,j}, \ldots\ldots\right),\ i = 1, \ldots\ldots, p^n,\ j = 1, \ldots, \ldots p. \tag{40}$$

In layer 3 (normalization layer), the outputs from layer 2 are normalized:

$$O_{3,i} = \overline{w_i} = \frac{w_i}{\sum_{i=1}^{p^n} w_i},\ i = 1, \ldots\ldots p^n \quad . \tag{41}$$

In layer 4 (function layer), the outputs from the layer 3 (w_i), weight the lineal regression function results from its node, making the output rules:

$$O_{4,i} = \overline{w_i} f_i = \overline{w_i}(p_{1,i}\, x_i + \ldots\ p_{n,i}\, x_n),\ i = 1, \ldots\ldots p^n \tag{42}$$

Where $p_{k,i},\ k = 1, \ldots, ..n,\ i = 1, \ldots, \ldots p^n$ are the consequence parameters. At the end, in the output layer (layer 5) a weighted sum is carried out with the outputs from layer 4:

$$O_{5,i} = \sum_{i=1}^{p^n} \overline{w_i} f_i = \frac{\sum_{i=1}^{p^n} w_i f_i}{\sum_{i=1}^{p^n} w_i} \tag{43}$$

To obtain the complexity, being N_E the number of the exponentiation operations, the number of operations of one node in layer 1 is given by: $N_{AL1} = 2$, $N_{ML1} = 2$; $N_{EL1} = 1$.

The number of operations of a node in layer 2, considering that the T-norm is the product and n is the size of the input layer, is: $N_{ML2} = n$.

The number of operations of a node in layer 3 (taken into account that accumulation may be calculated once for the complete layer, they will sum $n-1$ additions in the layer 3, N_{AL3}) is: $N_{ML3} = 1$.

The number of operations of each node of layer 4 is given by: $N_{AL4} = n$; $N_{ML4} = n + 1$.

Finally, the number of operations in the output layer is given by: $N_{AL5} = p^n - 1$;$N_{ML5} = n + 1$.

If there are p^n nodes in layer 1, and layers 2, 3 and 4, the number of operations of the complete network is:

$$N_A = p^n(N_{AL1} + N_{AL4}) + N_{AL3} + N_{AL5} \tag{44}$$

$$N_A = p^n(2+n) + n - 1 + p^n - 1 = p^n \cdot (3+n) + n - 2 \tag{45}$$

$$N_M = p^n(N_{ML1} + N_{ML2} + N_{ML3} + N_{ML4}) + N_{ML5} \tag{46}$$

$$N_M = p^n(2+n+1+n+1) + n + 1 = p^n \cdot (4+2n) + n + 1 \tag{47}$$

$$N_E = N_{EL1} \cdot p^n = p^n. \tag{48}$$

3.4. A new proposal of hybridization of intelligent and analytical identification techniques

So far, parametrical, neural and neuro-fuzzy techniques have been presented. All of these techniques have been individually applied to the modelling of the UAV. The model accuracy obtained with these methods may be enough for some applications, but in other cases we need to improve their performance in terms of precision or computational time. A first option is to modify the configuration. For example: the polynomials in the parametric techniques, the number of neurons of the hidden layer in the radial basis network, or the number of partitions per variable in the ANFIS system. When either it does not work well enough, or it works but the increment of computational cost is unacceptable, the hybridization of techniques may play an important role.

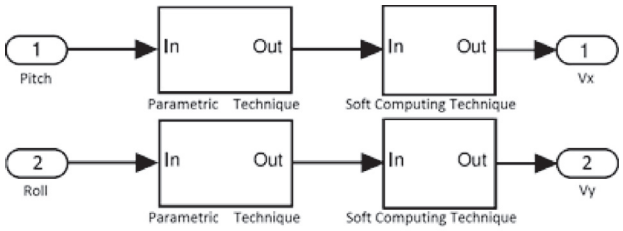

Fig. 7. Cascade connection of techniques.

The hybrid method is shown in Fig. 7. The process to obtain the complete model is composed by the following steps:

- The best parametric model is selected.
- The expected system output is simulated by applying a previously known set of signals.
- Finally, the soft computing technique is trained using the result of the simulation obtained in the previous step as input dataset, and the real output data as output dataset.

4. Results of the UAV model identification

4.1. Application of each technique individually

This subsection shows the graphical results of the application of each technique individually. In all the figures, velocity in the x-axis v_x is on the left and velocity in the y-axis v_y on the right, and the real output data is represented in blue. The response of the system is obtained for different steps as reference input (see Fig. 3).

Regarding the analytical techniques, Fig. 8 shows the output of the model generated by the application of the ARX approach. The numbers of the legends in the figures are associated to the configuration of the model. For instance, 221 means that the order of the polynomial $A(q, \theta)$ is 2, the order of the polynomial $B(q, \theta)$ is 2, and nk is 1 (see Eq. (24)).

As it is possible to see in Fig. 8, the higher the order of the polynomial is, the better the response gets. But in all the cases the models fail to represent the amplitude, which is underestimated, although they follow the signal changes.

The results of the ARMAX models are presented in Fig. 9. Again the configuration of each model is described by a sequence of numbers where the first digit is the order of the polynomial $A(q, \theta)$, the second one is the order of the polynomial $B(q, \theta)$, the third number is the order of the polynomial $C(q, \theta)$, and the last one is the number of delays nk (see Eq. (25)).

In this case, the model seems to represent better the behaviour of the real system than the ARX model. But again there is a difference between the model amplitude and the real output of the system.

The result of the application of OE parametric structure is shown in Fig. 10. In this case, the first number of the legend of each curve is the order of the polynomial $B(q, \theta)$, the second digit is the order of the polynomial $F(q, \theta)$, and the last one is the number of delays nk (see Eq. (26)).

In Fig. 10, it is possible to see how the model still presents this error in the amplitude, but in this case is lower than with the other parametric models, at least in the velocity in the y-axis.

Among the ARX models, the configuration 10–10–1 is the one that gives the best performance. Polynomial order higher than 10 does not improve the results. The same happens to the ARMAX models, 10 is the best polynomial order and higher order gives

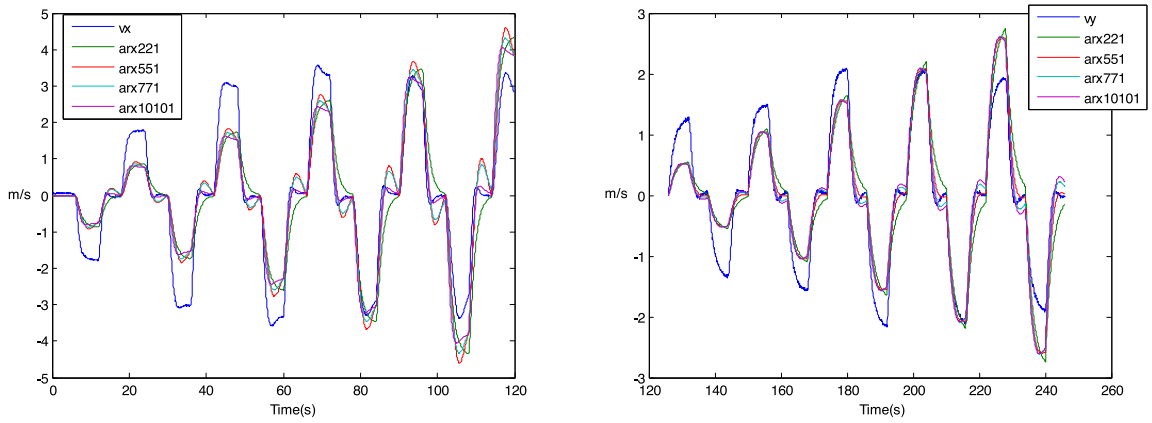

Fig. 8. Velocity in the x-axis (left) and in the y-axis (right) obtained by ARX models.

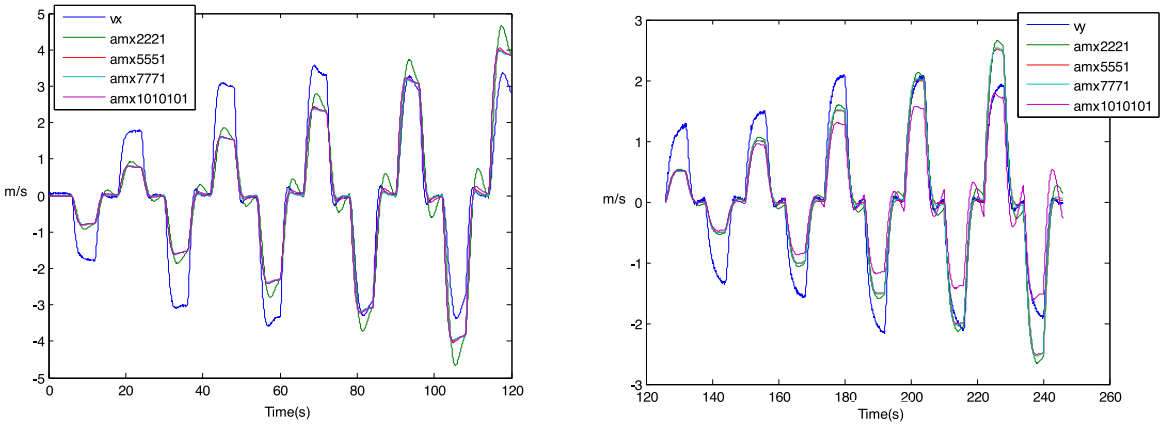

Fig. 9. Velocity in the x-axis (left) and in the y-axis (right) obtained by ARMAX models.

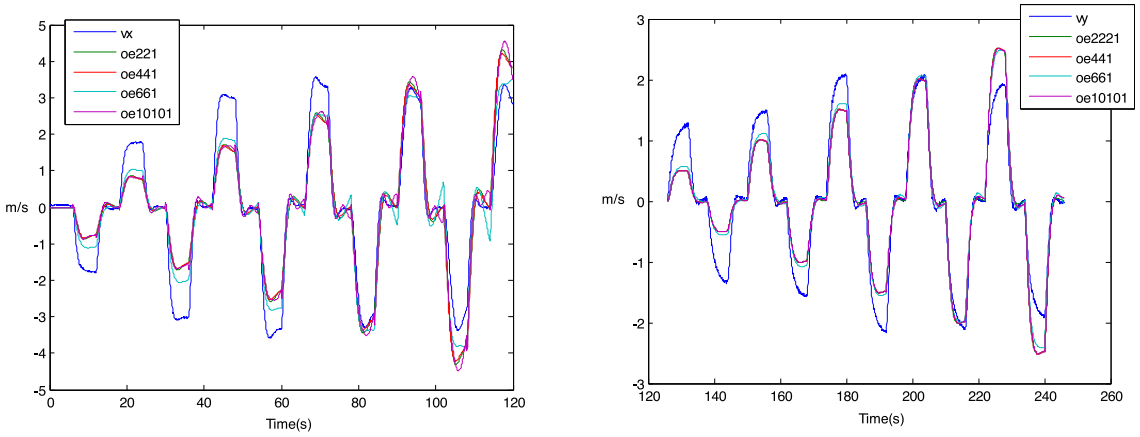

Fig. 10. Velocity in the x-axis (left) and in the y-axis (right) obtained by OE models.

even poorer performance while making the model more complex. Regarding the OE, the best value for the order of the polynomial has been found to be 6. A comparison between Figs. 8–10 reveals that OE is the parametrical technique which provides the most accurate model. In these figures it is also possible to see how, in general, v_y is better modelled than v_x.

Fig. 11 presents the simulation of the Radial Basis Networks (RBN) models. In this case, the configuration is given by the network input size (first number) and the number of neurons in the hidden layer (second digit in the legend). Again these models do not estimate well the peaks of the real signal; nevertheless, the response to changes is much better than with the polynomial models. Another clear outcome is that better models are obtained when the network input size increases. A similar behaviour is obtained when the number of hidden neurons increases.

Fig. 12 shows the output of the models obtained by the application of the ANFIS technique. In this case, the configuration is given by the network input size (first digit) and the number of partitions in the first layer (second number). In this figure it is possible to see how these models do not approximate well the complete shape of the signals; nevertheless, the response to the changes in amplitude is better than for the polynomial models. The comparison between Figs. 11 and 12 shows that, for the configurations tested, the models based on radial basis networks simulate better the response

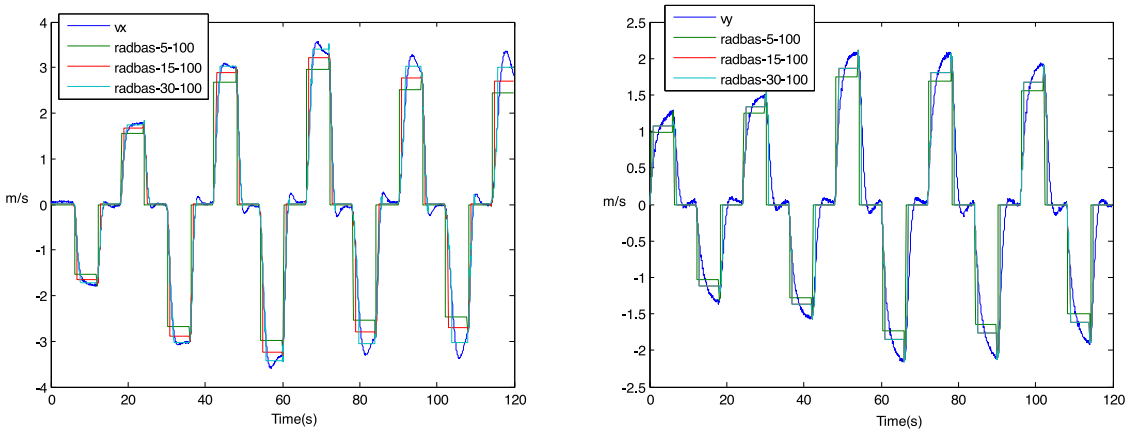

Fig. 11. Velocity in the *x*-axis (left) and in the *y*-axis (right) obtained by Radial Basis Networks.

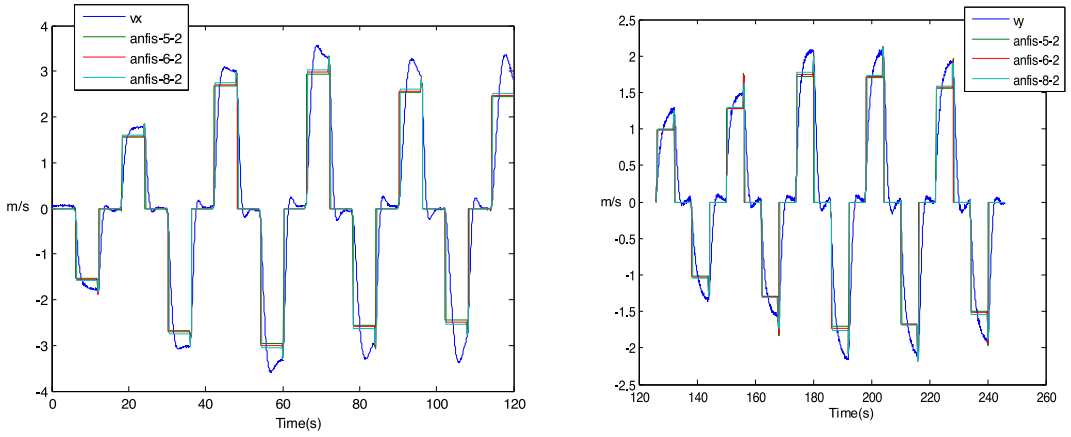

Fig. 12. Velocity in the *x*-axis (left) and in the *y*-axis (right) obtained by ANFIS.

to changes in the amplitude than the ANFIS models. The performance of ANFIS models would have been better if the number of partitions in layer 1 had been higher than 2. This number was not increased due to the increment of the computational cost of the network training.

4.2. Application of the hybrid method

We have combined different parametric and intelligent techniques to identify the UAV model. Again, for a train of different amplitude steps reference, the velocity in the *x*-axis (v_x) is shown on the left and the velocity in the *y*-axis (v_y) on the right of the figures.

Fig. 13 shows the best combinations of techniques. For each axis, the dark-blue line is the output of the real system; the green, red, light–blue, purple, and yellow lines represent the values of the output signal obtained by the models ARX-(10, 10, 1), RBF-(30, 200), ANFIS-(8, 2), ARX-(10, 10, 1)–RBF(5, 50), and ARX-(10, 10, 1)–ANFIS(5, 2), respectively. As we may have expected the signals obtained using only the ARX models are the worse for both axes. On the other hand, the hybrid methods, ARX–RB and ARX–ANFIS, provide better performance than the techniques individually applied, even if each technique was configured with the best parameters found (highest input size and more neurons in the hidden layer).

In the time domain, the system step response has been obtained for the different models. Fig. 14 shows the results for the same techniques and combinations of Fig. 12: ARX-(10, 10, 1), RBF-(30, 200), ANFIS-(8, 2), ARX-(10, 10, 1)–RBF(5, 50), and ARX-(10, 10, 1)–ANFIS(5, 2). In this case the better response for v_x is obtained by the combination of ARX and ANFIS, meanwhile for v_y the best result is given by the application of ARX and RB. These results match the quantitative errors presented in Table 1.

Fig. 15 shows the system response in the frequency domain, v_x on the left and v_y on the right. The dark-blue line represents the values of the frequency response estimated by the real data; green, red, light-blue, purple, and yellow lines represent the values of the estimated frequency response obtained with the models ARX-(10, 10, 1), RBF-(30, 200), ANFIS-(8, 2), ARX-(10, 10, 1)–RBF(5, 50), and ARX-(10, 10, 1)–ANFIS(5, 2), respectively. The results are qualitatively similar to the ones in the time domain. The best performance is obtained by the hybrid techniques. For the velocity in the *x*-axis the combination of ARX + ANFIS provides the best fitting to the estimated response from the real data. The best response for v_y is obtained by the combination of ARX + RB. The graphical behaviour is in line with the numerical estimation (Table 1).

4.3. Discussion of the results

The numerical results of the individual and hybrid application of the strategies are summarized in Table 1, in terms of the errors defined in (11)–(13) and (18). In order to do a fair comparison, the best configuration for each technique when applied individually has been selected.

In the case of the parametric error techniques, ARX and OE were finally applied because the ARMAX model gave very similar results and it was more complex. If we focus on ARX, the best model is generated by the configuration ARX-(10, 10, 10); if we look at the OE parametric structure, the best model is given by the configuration OE(6, 6, 1). Regarding the neural networks, the best model is the one obtained by the radial basis network with the

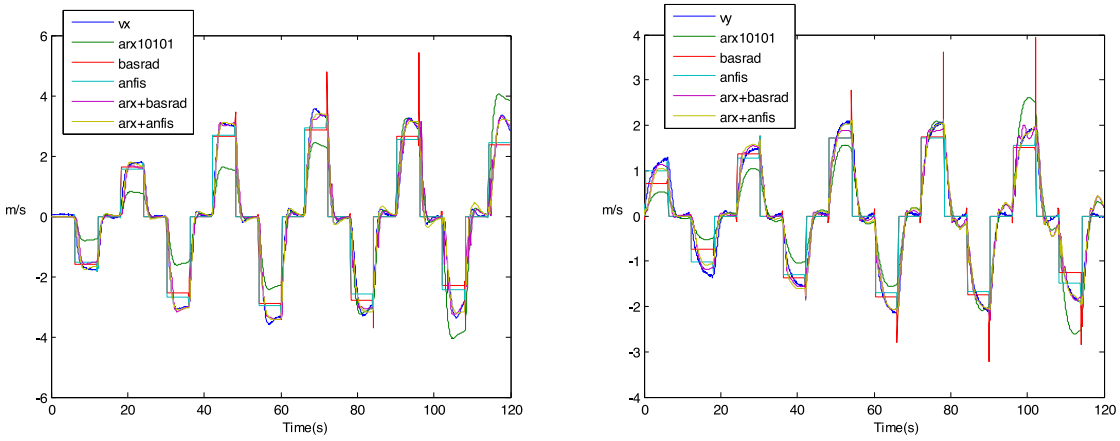

Fig. 13. Velocity in the *x*-axis and in the *y*-axis obtained by the application of different hybridizations. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)

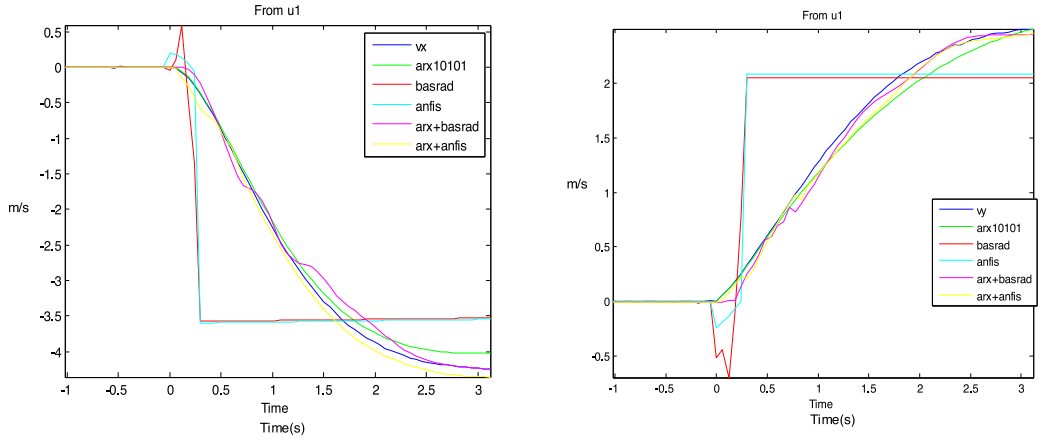

Fig. 14. Estimated step responses for v_x (left) and v_y (right) with different models.

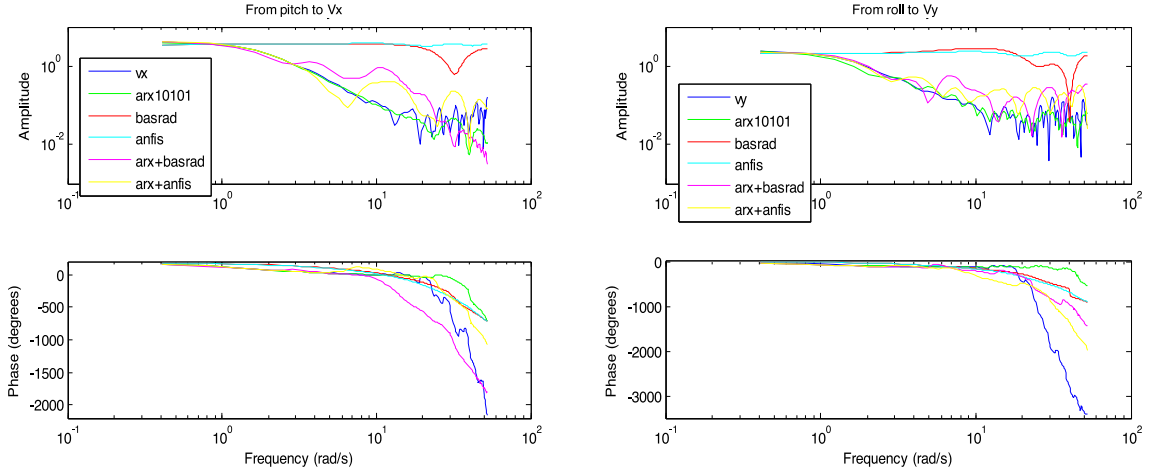

Fig. 15. Estimated frequency response of velocity in the *x*-axis and in the *y*-axis. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)

configuration (30, 200), meaning that the size of the input array is 30 and the number of neurons in the hidden layer is 200. On the other hand, the best configuration found for the neuro-fuzzy system was (8, 2), that is, the size of the input array is 8 and the number of partitions is 2. Therefore, these models have been introduced in the comparisons.

In Table 1, there are three main groups of rows, each one separated from the rest with a horizontal line. The first group of rows shows the results of the models obtained by the application of the selected techniques individually. The second group of rows presents the results of the models generated by the hybridization of intelligent and parametric techniques. The third group also shows the results of hybrid techniques but with a slight incre-

Table 1
Comparison in terms of MSE and TC for different techniques and combinations.

Technique	v_x				v_y				TC
	MSE_T	MSE_S	MSE_F	MSE_C	MSE_T	MSE_S	MSE_F	MSE_C	
ARX10, 10, 1	0.5056	**0.0122**	**0.0060**	0.1746	0.1326	0.0024	**0.0077**	0.0476	91.6
OE6, 6, 1	0.2915	0.0418	0.0352	0.1228	0.0820	**0.0010**	0.0085	**0.0305**	**54.56**
RB-(5, 30)	0.5097	1.1919	3.4022	1.7013	0.2345	0.3796	1.5013	0.7051	1497.9
RB-(30, 200)	**0.0240**	0.0171	0.2574	**0.0995**	**0.0159**	0.0145	0.2729	0.1011	38,136
ANFIS-(5, 2)	0.4816	1.1807	6.1789	2.6137	0.2106	0.3765	2.1761	0.9211	3743.8
ANFIS-(8, 2)	0.3363	0.8236	4.9864	2.0488	0.1531	0.2731	1.7381	0.7214	36,276
ARX–ANFIS(5, 2)	**0.0145**	**0.0087**	**0.0252**	**0.0161**	**0.0100**	0.0057	0.0198	0.0118	3835.4
OE–ANFIS(5, 2)	0.0295	0.1356	0.1433	0.1028	0.0103	0.0030	0.0281	0.0138	3798.4
ARX–RB(5, 30)	0.0572	0.0172	0.0774	0.0506	0.0144	**0.0015**	**0.0059**	**0.0073**	1589.5
OE–RB(5, 30)	0.0429	0.1163	0.0848	0.0813	0.0127	0.0039	0.0162	0.0109	**1552.5**
ARX–RB(5, 50)	0.0125	**0.0031**	0.0355	**0.0170**	0.0068	0.0036	0.0108	0.0071	**2356.6**
OE–RB(5, 50)	0.0286	0.1114	0.0754	0.0718	0.0071	0.0007	0.0110	0.0063	**2356.6**
ARX–RB(10, 50)	**0.0103**	0.0040	**0.0340**	0.0287	0.0059	0.0022	0.0108	0.0063	3764.1
OE–RB(1050)	0.0270	0.0863	0.0445	0.0526	**0.0057**	**0.0004**	**0.0078**	**0.0046**	3727.1

ment of the complexity of their configuration; the parameters of the neural network change from (5, 30) to either (5, 50) or (10, 50). For each column and each main row, the best result has been boldfaced, and for each column the best result has been underlined.

The last column of Table 1 represents the computational time (TC) as defined in Section 3.2.3. Analyzing this indicator, it is interesting to point out the difference between the value for soft computing techniques, such as RB-(30,200), and the corresponding hybrid method (ARX–RB), in this case 38136 against 2356.6. On the one hand, RB is more accurate than the parametric technique ARX (0.0995 vs. 0.1746 MSE_C), but it is much more complex in terms of computational time (38136 vs. 91.6). Nevertheless, with the hybridization ARX–RB the precision is now 0.0170 while the TC is decreased to 2356.6.

In general, see Table 1, the hybrid techniques provide better results than the techniques individually applied. Regarding v_x, the best result is the obtained by the combination of ARX-(10,10,1) and ANFIS-(5,2) with MSE_C error of 0.0161; for v_y, the best result is given by OE-(6,6,1) with the radial basis function (10,50), in this case, the error is 0.0046.

Another interesting way of seeing the improvements provided by the hybridization is to compare each individual technique with the corresponding hybrid one. For example ARX–RB(10,50) gives a MSE_C ten times lower than RB-(30,200). It is a clear example of an important reduction of complexity (from 15–200 to 5–50); indeed, the computational cost (TC) goes from 38,136 to 3764.1 (90% reduction). At the same time the accuracy increases: from 0.0995 to 0.0170 (see Table 1).

Another example of the efficiency of the hybridization is the synergy of OE-(6, 6, 1) and RB-(5, 30). Although it means an increment of 23 operations, the performance in terms of the MSE_c error goes from 1.7013 to 0.0813.

Applying ANFIS technique also gives similar improvements. The combination of OE-(6, 6, 1) and ANFIS-(5, 2) reduces the MSE_c from 2.6137 to 0.1028, while the complexity increases from 3743.8 to 3798.4, only an increment of 1.46%.

Even more, we have realized that when we place a parametric technique module between the input signal and the neural network (Fig. 7), we are reducing an undesirable effect that can be notice on the RBN and ANFIS response (Figs. 11 and 12). The output seems to be saturated and this is due to the input training vector. Nevertheless, this effect is corrected (Fig. 13) when a combination of parametric and intelligent techniques is used.

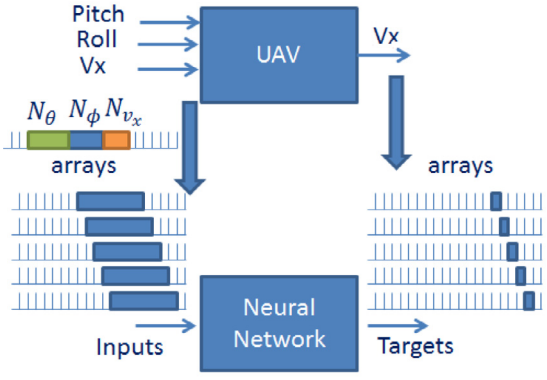

Fig. 16. On-line training procedure.

Therefore, considering the low increment of complexity introduced by the hybridization and the important improvement in terms of MSE error, it is much better to combine both the soft computing techniques with the parametric techniques than increasing the complexity of the networks. In other words, the improvement of the models accuracy produced by the hybridization of the techniques clearly compensates the low rising of the computational cost.

4.4. Influence of the off-line/on-line learning algorithm: adaptive neural networks

In addition, we are going to show the application of adaptive neural networks to the identification of the UAV model [52]. Using off-line learning methods, the first step is to generate a dataset with all the input/output pairs of data. Then, this set is split in order to use part of it for training and the other part for testing. But the selection of the data for training is a delicate task. Indeed, the accuracy of the model depends on the data used to train the network. Results prove that off-line methods may be too sensitive to the partition of the dataset. Another disadvantage of these off-line approaches is that they do not capture the dynamics when it is changing over time. Therefore, we will use an on-line learning procedure to let the networks identify the model, using new data every sample time (Fig. 16).

In this case, due to the adaptive learning, the function that estimates the system behaviour f is different at each sampling time. That is, f_{v_x} and f_{v_y} are continuously changing over time. Thus, the discrete output variables, velocity in the x-axis and in the y-axis,

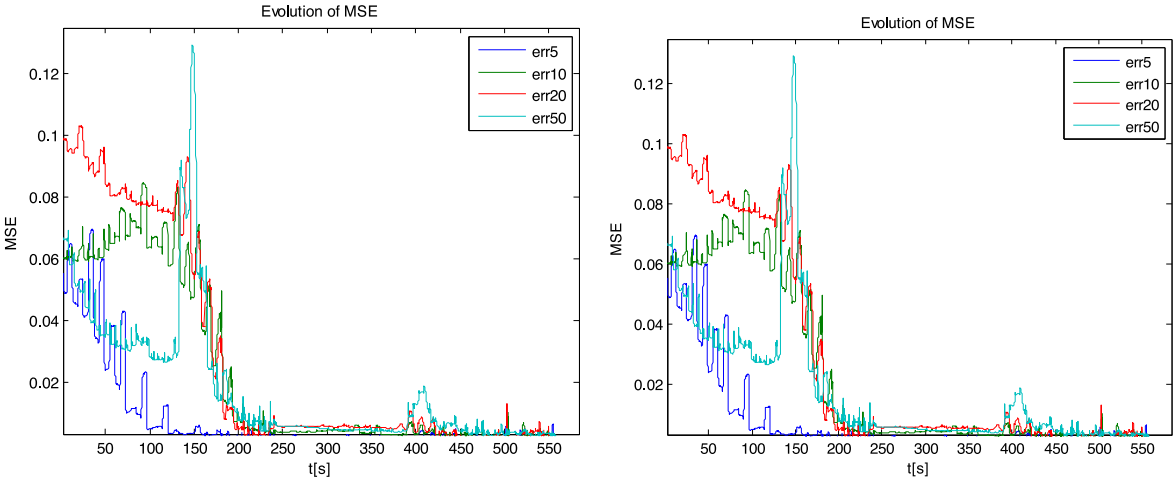

Fig. 17. MSE model error evolution of velocity in the *x*-axis (left), and in the *y*-axis (right). (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)

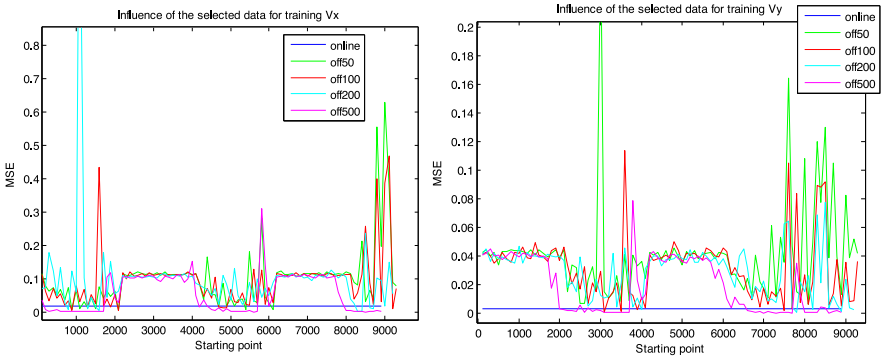

Fig. 18. Influence of selected training data on v_x (left) and on v_y (right). (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)

are given by the following expressions:

$$v_x[k+1] = f_{v_x}\begin{pmatrix} v_x[k] & u_\theta[k] & u_\phi[k] \\ v_x[k-1] & u_\theta[k-1] & u_\phi[k-1] \\ \cdots\cdots & \cdots\cdots & \cdots\cdots \\ v_x[k-N_{v_x}] & u_\theta[k-N_{u_\theta}] & u_\phi[k-N_{u_\phi}] \end{pmatrix} \quad (49)$$

$$v_y[k+1] = f_{v_y}\begin{pmatrix} v_y[k] & u_\theta[k] & u_\phi[k] \\ v_y[k-1] & u_\theta[k-1] & u_\phi[k-1] \\ \cdots\cdots & \cdots\cdots & \cdots\cdots \\ v_y[k-M_{v_y}] & u_\theta[k-M_{u_\theta}] & u_\phi[k-M_{u_\phi}] \end{pmatrix} \quad (50)$$

Fig. 17 shows the evolution of the model error using adaptive neural networks, MLP types, with different configurations. The number associated to each curve means the configuration; for example, "err5" (blue line) means that the parameters $(N_{v_x},\ N_{u_\theta},\ N_{u_\phi})$ are set to 5 for v_x. This value also corresponds to the parameters$(M_{v_y}, M_{u_\theta},\ M_{u_\phi})$ for the velocity in the *y*-axis. The MSE decreases as the neural network learns. The values of N_i and M_j are the number of consecutive samples for each system variable i and j used to train the network. That is, the learning improves the identified model continuously.

Fig. 18 shows how the selection of the training data influences the final result. The parameters for the velocity in both axes, $(N_{v_x},\ N_{u_\theta},\ N_{u_\phi})$ and $(M_{v_y}, M_{u_\theta},\ M_{u_\phi})$, were set to (5, 5, 5) in the on-line case. The number associated to each curve means the number of samples used to train the network (from 50 to 500). The samples were taken from a random point of the dataset, consecutively. This experiment was randomly executed 9000 times. The MSE error was calculated during each execution and its value is represented in the *y*-axis (normalized). The MSE error obtained with the adaptive network (on-line learning) is shown in blue; the MSE curves obtained by the application of the off-line methods are represented by different colours depending on the number of samples taken (for example, 50 samples, green line).

As it is possible to see, on-line MSE error is almost always smaller than the error of the off-line models. In addition, in these figures it may be noticed how the MSE is very dependent on the data selected for training.

Table 2 shows the comparison of the MSE error obtained by applying on-line and off-line training. Every column of each row has the same configuration, $(N_{v_x},\ N_{u_\theta},\ N_{u_\phi})$ and $(M_{v_y}, M_{u_\theta},\ M_{u_\phi})$. In this test, 200 samples have been used to train the off-line model. For each type of training, the best configuration has been

Table 2
Comparison of MSE for on-line and off-line techniques.

Configuration	v_x		v_y	
$N_{v_x},\ N_{u_\theta},\ N_{u_\phi}$	On-line	Off-line	On-line	Off-line
2, 2, 2	0.01552	0.09038	0.004223	0.01015
4, 4, 4	0.01101	0.11556	0.002758	0.04318
6, 6, 6	**0.00320**	0.06491	0.002689	0.02205
8, 8, 8	0.01478	0.05856	**0.001584**	0.02863
10, 10, 10	0.02060	0.05908	0.002710	0.02115

underlined. Besides, the best global results have been boldfaced. In two cases, the best configuration is (8, 8, 8); in the rest of cases the network configurations with values (2, 2, 2) and (6, 6, 6) have resulted with a lower error. In any case, the best total results have been obtained by applying on-line learning with the configuration (8, 8, 8). To summarize, in this experiment on-line models provide smaller MSE than off-line methods.

5. Conclusions

In this work, models of a quadrotor unmanned aerial vehicle have been generated by applying different identification techniques: parametric structures, soft computing techniques (neural networks and neuro-fuzzy inference systems), and the hybridization of them.

In order to compare the UAV models, some indicators have been defined: the error in the time and frequency domains, accuracy, and complexity in terms of computational cost (number of elemental operations). Analyzing the results, the hybrid strategies that combine parametric and intelligent techniques provide models whose balance between accuracy and complexity is much better than if they are individually applied.

Besides, off-line learning neural networks and adaptive networks with on-line learning strategies have been applied. This has allowed us to show the influence of the training dataset partition on the final model. The results prove that the network is able to learn on-line and to improve the identified model continuously.

These models, obtained from real data, are control-oriented. The final goal is to use them to design, simulate and implement different controllers and to test them on real systems, what is proposed as future work. Among others possible future lines we may highlight: development of training methods which use the analyzed equations of computational complexity to stop the training when the best accuracy–complexity relation is obtained, something like obtaining Pareto fronts; testing other topologies to generate hybrid strategies; development of controllers for the UAV under study and test them on the real platform.

Acknowledgement

The authors would like to acknowledge the data provided by the Control Engineering Group of the Spanish Committee of Automatic [47]. Author Matilde Santos would like to thank the Spanish Ministry of Science and Innovation (MICINN) for support under project DPI2013-46665-C2-1-R.

References

[1] Q. Zhu, A.T. Azar, Complex System Modelling and Control Through Intelligent Soft Computations, Springer, Germany, 2015.
[2] D.W. Hybertson, Model-Oriented Systems Engineering Science: A Unifying Framework For Traditional and Complex Systems, CRC Press, 2016.
[3] C.J. Puccia, L. Richard, Qualitative Modeling of Complex Systems, Harvard University Press, Cambridge, MA, 2013.
[4] G.C. Goodwin, R.L. Payne, Dynamic System Identification: Experiment Design and Data Analysis, Academic Press, New York, 1977.
[5] H.M.R. Ugalde, J.C. Carmona, V.M. Alvarado, J. Reyes-Reyes, Neural network design and model reduction approach for black box nonlinear system identification with reduced number of parameters, Neurocomputing 101 (2013) 170–180.
[6] M. Santos, R. López, J.M. de la Cruz, A neuro-fuzzy approach to fast ferry vertical motion modelling, Eng. Appl. Artif. Intell. 19 (3) (2006) 313–321.
[7] L. Weng, Q. Liu, M. Xia, Y.D. Song, Immune network-based swarm intelligence and its application to unmanned aerial vehicle (UAV) swarm coordination, Neurocomputing 125 (2014) 134–141.
[8] M. Santos, V. Lopez, F. Morata, Intelligent fuzzy controller of a quadrotor, in: Proceedings of the 2010 International Conference on Intelligent Systems and Knowledge Engineering (ISKE), IEEE, November 2010, pp. 141–146.
[9] N. Marichal, M. Tomas-Rodriguez, S. Castillo Rivera, A. Hernandez, Modeling and analysis of vibrations in a UAV helicopter with a vision system, Int. J. Adv. Rob. Syst. 9 (2012) 220.
[10] P. García-Auñón, M. Santos, J.M. de la Cruz, Parameter selection based on fuzzy logic to improve UAV path-following algorithms, Int. J. Appl. log. (2016), doi:10.1016/j.jal.2016.11.025.
[11] J. Menoyo-Larrazabal, M. Santos-Peñas, Intelligent rudder control of an unmanned surface vessel, Expert Syst. Appl. 55 (2016) 106–117.
[12] S Estellés-Martínez, M Tomás-Rodríguez, Three-dimensional trajectory tracking of a quadrotor through PVA control, Revista Iberoam. Autom. Inf. Indust. RIAI 11 (1) (2014) 54–67.
[13] N.V. Hoffer, C. Coopmans, A.M. Jensen, Y. Chen, A survey and categorization of small low-cost unmanned aerial vehicle system identification, J. Intell. Rob. Syst. 74 (1–2) (2014) 129–145.
[14] A. Gessow, G.C. Myers, Aerodynamics of the Helicopter, Frederick Ungar, 1952.
[15] A.R.S. Bramwell, Helicopter Dynamics, Edward Arnold Ltd. Publisher, London, 1976.
[16] G. Cai, B.M. Chen, K. Peng, M. Dong, T.H. Lee, Modeling and control of the yaw channel of a UAV helicopter, IEEE Trans. Indust. Electron. 55 (9) (2008) 3426–3434.
[17] A. Budiyono, K.J. Yoon, F.D. Daniel, Integrated identification modeling of rotorcraft-based unmanned aerial vehicle, in: Proceedings of the Seventeenth Mediterranean Conference on Control and Automation, 2009. MED'09, IEEE, 2009, June, pp. 898–903.
[18] A. El-Saadany, A. Medhat, Y.Z. Elhalwagy, Flight simulation model for small scaled rotor craft-based UAV, in: Proceedings of the Thirteenth International Conference on Aerospace Sciences and Aviation Technology, 2009, May Paper: ASAT-13-CT-31.
[19] J. del-Cerro, A. Martinez, A. Barrientos, Modeling and control prototyping of unmanned helicopters, in: T.M. Lam (Ed.), Aerial Vehicles, INTECH Open Access Publisher, 2009, pp. 87–106.
[20] P. Castillo, R. Lozano, A.E. Dzul, Modeling and control of mini-helicopters, Modeling and Control of Mini-Flying Machines: Advances in Industrial Control, Springer, 2005, pp. 81–119.
[21] H. Wu, D. Sun, Z. Zhou, Model identification of a micro air vehicle in loitering flight based on attitude performance evaluation, IEEE Trans. Robot. 20 (4) (2004) 702–712.
[22] J.C. Morris, M. Van Nieuwatadt, P. Bendotti, Identification and control of a model helicopter in hover, in: Proceedings of the American Control Conference, 2, June 1994, p. 1236.
[23] B. Mettler, M.B. Tischler, T. Kanade, System identification of small-size unmanned helicopter dynamics, in: Annual Forum Proceedings, 2, American Helicopter Society, May 1999, pp. 1706–1717.
[24] M.B. Tischler, M.G. Cauffman, Frequency-response method for rotorcraft system identification: flight applications to BO105 coupled rotor/fuselage dynamics, J. Am. Helicopter Soc. 37 (3) (1992) 3–17.
[25] M.B. Tischler, System identification methods for aircraft flight control development and validation, Adv. Aircr. Flight Control (1996) 35–69.
[26] S.M. Ahmad, M.H. Shaheed, A.J. Chipperfield, M.O. Tokhi, Nonlinear modeling of a twin rotor MIMO system using radial basis function networks, in: Proceedings of the IEEE 2000 National Aerospace and Electronics Conference. NAECON 2000., IEEE, 2000, pp. 313–320.
[27] S. Hashimoto, T. Ogawa, S. Adachi, A. Tan, G. Miyamori, System identification experiments on a large-scale unmanned helicopter for autonomous flight, in: Proceedings of the 2000 IEEE International Conference on Control Applications, 2000, IEEE, 2000, pp. 850–855.
[28] W. Yuan, J. Katupitiya, A time-domain grey-box system identification procedure for scale model helicopters, in: Proceedings of the 2011 Australasian Conference on Robotics and Automation, 2011, December.
[29] M. Manaı, A. Desbiens, E. Gagnon, Identification of a UAV and design of a hardware-in-the-loop system for nonlinear control purposes, in: Proceedings of the AIAA Guidance, Navigation, and Control Conference, 5, AAIA, 2005, pp. 2005–6483.
[30] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Netw. 2 (5) (1989) 359–366.
[31] R. San Martin, A. Barrientos, P. Gutiérrez, J. del Cerro, Unmanned aerial vehicle (UAV) modeling based on supervised neural networks, in: Proceedings of the International Conference on Robotics and Automation, ICRA., 2006, May, pp. 2497–2502.
[32] R.S. San Martin, A. Barrientos, P. Gutierrez, J.D. Cerro, Neural networks training architecture for UAV modeling, in: Proceedings of the World Automation Congress, 2006. WAC'06, IEEE, 2006, July, pp. 1–6.
[33] V. Puttige, S. Anavatti, Real-time system identification of unmanned aerial vehicles: a multi-network approach, J. Comput. 3 (7) (2008) 31–38.
[34] I.E. Putro, A. Budiyono, K.J. Yoon, D.H. Kim, Modeling of unmanned small scale rotorcraft based on neural network identification, in: Proceedings of the IEEE International Conference on Robotics and Biomimetics, 2008. ROBIO 2008., IEEE, February 2009, pp. 1938–1943.
[35] Z. Taha, A. Deboucha, M.B. Dahari, Small-scale helicopter system identification model using recurrent neural networks, in: Proceedings of the TENCON 2010 – 2010 IEEE Region 10 Conference, IEEE, November 2010, pp. 1393–1397.
[36] Q. Lin, Z. Cai, Y. Wang, J. Yang, L. Chen, Adaptive flight control design for quadrotor UAV based on dynamic inversion and neural networks, in: Proceedings of the 2013 Third International Conference on Instrumentation, Measurement, Computer, Communication and Control (IMCCC), IEEE, September 2013, pp. 1461–1466.
[37] K.S. Narendra, K. Parthasarathy, Identification and control of dynamical systems using neural networks, IEEE Trans. Neural Netw. 1 (1) (1990) 4–27.
[38] S.S. Shamsudin, X. Chen, Identification of an unmanned helicopter system us-

ing optimised neural network structure, Int. J. Model. Identif. Control 17 (3) (2012) 223–241.
[39] Y. Li, N. Sundararajan, P. Saratchandran, Neuro-controller design for nonlinear fighter aircraft maneuver using fully tuned RBF networks, Automatica 37 (8) (2001) 1293–1301.
[40] I.B. Tijani, R. Akmeliawati, A. Legowo, A. Budiyono, Nonlinear identification of a small scale unmanned helicopter using optimized NARX network with multiobjective differential evolution, Eng. Appl. Artif. Intell. 33 (2014) 99–115.
[41] K.S. Hatamleh, M. Al-Shabi, A. Al-Ghasem, A.A. Asad, Unmanned aerial vehicles parameter estimation using artificial neural networks and iterative bi-section shooting method, Appl. Soft Comput. 36 (2015) 457–467.
[42] H. Jin, X. Li, X. Zhang, Y. Li, Study on the modeling methods of the quad-rotor unmanned helicopter, in: Proceedings of the 2015 Twenty-Seventh Chinese Control and Decision Conference (CCDC), IEEE, 2015, May, pp. 4835–4839.
[43] V.R. Puttige, S.G. Anavatti, Real-time neural network based online identification technique for a UAV platform, in: Proceedings of the International Conference on Computational Intelligence for Modeling Control and Automation, 2006 and International Conference on Intelligent Agents Web Technologies and Internet Commerce, IEEE, 2006, November, p. 92.
[44] S.A. Salman, V.R. Puttige, S.G. Anavatti, Real-time validation and comparison of fuzzy identification and state-space identification for a UAV platform, in: Proceedings of the 2006 IEEE Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, and 2006 IEEE International Symposium on Intelligent Control,, IEEE, 2006, October, pp. 2138–2143.
[45] M. Mahfouz, M. Ashry, G. Elnashar, Design and control of quadrotor helicopters based on adaptive neuro-fuzzy inference system, Int. J. Eng. Res. Technol. 2 (12) (2013) 479–485.
[46] Parrot. www.parrot.com (2013)
[47] CEA (2016). Control Engineering Group of the Spanish Committee of Automatic: www.ceautomatica.es/og/ingenieria-de-control
[48] D.J. Struik, A Source Book in Mathematics, 1200–1800, Princeton University Press, 2014.
[49] E. Walter, L. Pronzato, Identification of Parametric Models, 8, Springer, 1997.
[50] J. Park, I.W. Sandberg, Universal approximation using radial-basis-function networks, Neural Comput. 3 (2) (1991) 246–257.
[51] J.S.R. Jang, ANFIS: adaptive-network-based fuzzy inference system, IEEE Trans. Syst. Man Cybern. 23 (3) (1993) 665–685.
[52] J.E. Sierra, M. Santos, Adaptive neural control-oriented models of unmanned aerial vehicles, in: Proceedings of the Tenth International Conference on Soft Computing Models in Industrial and Environmental Applications, Springer International Publishing, 2015, pp. 329–337.

Jesus Enrique Sierra Garcia was born in Burgos, Spain. He received his M.Sc. degrees in Electronics and Telecommunications from the University of Valladolid (UVA) in 2007 and 2015, respectively, and his M.Sc degree in Control Engineering from the National University for Distance Education (UNED) in 2014. Since 2012 he has been with the Civil Engineering Department at the University of Burgos, where he currently is a Lecturer in Languages and Computer Systems. Since 2013 he has been working with the R&D department of ASTI, where he is Technology Development Manager and Head of Innovation. His major research interests are: Intelligent Control (fuzzy and neuro-fuzzy), Signal Processing, Modelling and Simulation.

Matilde Santos Peñas was born in Madrid, Spain. She received her B.Sc. and M.Sc. degrees in Physics (Computer Engineering) and her Ph.D. in Physics in 1994, from the University Complutense of Madrid (UCM). Since 1986 she has been with the Department of Computer Architecture and Systems Engineering at the UCM, where she currently is a Full Professor in System Engineering and Automatic Control. Her major research interests are: Intelligent Control (fuzzy and neuro-fuzzy), Signal Processing, Modelling and Simulation.

2 Wind and payload disturbance rejection control based on adaptive neural estimators: application on quadrotors

BIBLIOGRAPHIC DESCRIPTION

Title

Wind and payload disturbance rejection control based on adaptive neural estimators: application on quadrotors.

Citation

Sierra. J.E., Santos, M. (2019). Wind and payload disturbance rejection control based on adaptive neural estimators: application on quadrotors. Complexity, Article ID 6460156, 20 pages.

Abstract

In this work a new intelligent control strategy based on neural networks is proposed to cope with some external disturbances that can affect quadrotor unmanned aerial vehicles (UAV) dynamics. Specifically, the variation of the system mass during logistic tasks and the influence of the wind are considered. An adaptive neuro mass estimator and an adaptive neuro disturbance estimator complement the action of a set of PID controllers, stabilizing the UAV and improving the system performance. The control strategy has been extensively tested with different trajectories: linear, helical, circular, and even a lemniscate one. During the experiments the mass of the UAV is triplicated and winds of 6 and 9 in the Beaufort's scale are introduced. Simulation results show how the on-line learning of the estimator increases the robustness of the controller, reducing the effects of the changes in the mass and of the wind on the quadrotor.

References

San Juan, V. et al(2018); Szabolcsi, R.(2015); Grassi, R. et al(2017); Ruiz, A. G. E. et al(2016); Santos, M. et al(2010); Wu, Z. et al(2017);Toriz, P. A. et al(2017);Fahmy, A. A. et al(2015); García-Auñón, P. et al(2017); Zhou, Z. et al(2018); Zhang, X. et al(2014);Santos, M. (2011);Santos, M. et al(2006);Min, B.-C. et al (2011);Wang, C. et al(2014);Wang, C. et al(2016);Sreenath, K. et al(2013, May);Sreenath, K. et al(2013);Lee, S. et al(2017, June); Palunko, I. et al(2011, August);Palunko, I. et al(2012, May);Kotaru, P. et al(2017,

May);Yüksel, B. et al(2014, July);Escareño, J. et al(2013);Alexis, K. et al(2011);Cabecinhas, D. et al(2015); Besnard, L. et al(2012);Sierra. J.E. et al(2018);Bansal, S. et al(2016);Boudjedir, H. et al(2014);Bakshi, N.A. et al(2016); Sierra, J. E. et al(2018);Shastry, A. K. et al(2016); Yañez-Badillo, H. et al(2017);Nicol, C. et al(2008, May);Etele, J. (2006)

Impact Factor

Complexity has an impact factor of 1.824 according to Thomson Reuters Journal Citation Reports (2017).

Hindawi
Complexity
Volume 2019, Article ID 6460156, 20 pages
https://doi.org/10.1155/2019/6460156

Research Article

Wind and Payload Disturbance Rejection Control Based on Adaptive Neural Estimators: Application on Quadrotors

Jesús Enrique Sierra and Matilde Santos

Computer Science Faculty, Complutense University of Madrid, 28040 Madrid, Spain

Correspondence should be addressed to Jesús Enrique Sierra; jesier01@ucm.es and Matilde Santos; msantos@ucm.es

Received 29 July 2018; Revised 7 December 2018; Accepted 12 December 2018; Published 3 January 2019

Academic Editor: Dan Selişteanu

In this work, a new intelligent control strategy based on neural networks is proposed to cope with some external disturbances that can affect quadrotor unmanned aerial vehicles (UAV) dynamics. Specifically, the variation of the system mass during logistic tasks and the influence of the wind are considered. An adaptive neuromass estimator and an adaptive neural disturbance estimator complement the action of a set of PID controllers, stabilizing the UAV and improving the system performance. The control strategy has been extensively tested with different trajectories: linear, helical, circular, and even a lemniscate one. During the experiments, the mass of the UAV is triplicated and winds of 6 and 9 in Beaufort's scale are introduced. Simulation results show how the online learning of the estimator increases the robustness of the controller, reducing the effects of the changes in the mass and of the wind on the quadrotor.

1. Introduction

In recent years, new and valuable applications of unmanned aerial vehicles (UAV) have emerged in different sectors such as defense, security, construction, agriculture, entertainment, and shipping [1–3]. These and other applications demand the design of efficient and robust controllers for those autonomous vehicles. That is why the modelling and control of these complex and unstable systems still motivate the research and interest of the scientific community [4–11].

Nevertheless, the modelling and control of quadrotor vehicles are not an easy task. The complexity comes from the randomness of the airstreams and of the exogenous forces, the high nonlinearity dynamics, the coupling between the internal variables, the uncertainty of the measurements, etc. These factors make the techniques based on artificial intelligence a promising approach for the identification and control of these systems [12].

Moreover, these intelligent strategies are especially interesting when the model parameters change while the system is working [13]. For example, the total mass will undergo variations when the vehicle is performing logistic tasks, since the mass depends on the loads that are shipped.

There are few works that study the effect of the payload variation on the quadrotor dynamics and that take it into account. In [14], an adaptive control is used to mitigate the impact of the parameter variation by estimating them under guaranteed performance. In [15], least square and gradient methods are implemented for adaptive parameter estimation, which are used to update the control output to the current UAV mass and inertia moment. Recently, Wang [16] applied again the same strategy to estimate the variations in the payload and the effect of the wind gusts.

We propose to use estimators based on neural networks. They offer three main advantages with respect to other techniques: the fact that knowledge about the internal structure of the system to be estimated is not necessary, unlike other estimation techniques such as PEM (parametric error models), Hammerstein-Weiner, and Volterra; its online learning ability; and the easy parallelization. Even more, using adaptive neural networks allows the online learning of the estimation not to be so biased by the selection of the training data. This is a relevant advantage against other offline estimation methods which are very sensitive to the dataset used during the training.

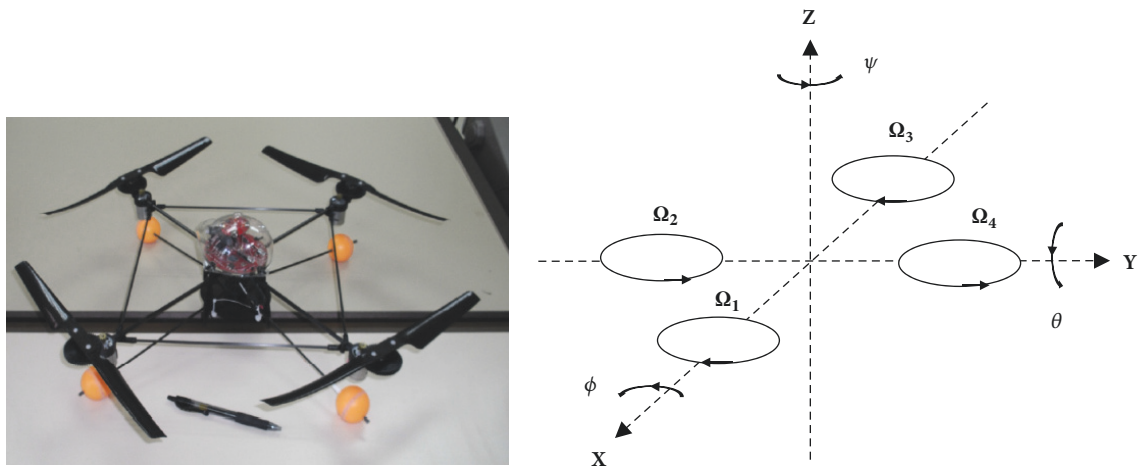

Figure 1: A quadrotor vehicle (left) and UAV's coordinate system (right).

There are other papers more focused on a closely related problem, the UAV stabilization and path tracking when its center of gravity changes, and what happens by the manipulation of suspended loads by cables. In [17], using the flatness property, a trajectory generation method is presented that enables finding nominal trajectories with various constraints that result in minimal load swing. Also, the same authors present a very interesting cooperative control strategy to manipulate suspended loads by several quadrotors at the same time [18]. In [19], a controller based on a set of connected PD regulators is used to tackle this issue. An adaptive tracking controller based on output feedback linearization is used in [20]; this controller compensates for dynamical changes in the center of gravity of the quadrotor. In [21], this issue is again addressed using the dynamic programming approach. In [22], the complexity of the aerial vehicle is incremented by considering the elasticity of the cable in the system equations.

Other studies are focused on the rejection of wind disturbances. In [23], Lyapunov-based observers are used to estimate the external force disturbances. In [24], a control strategy based on sliding mode and adaptive control techniques is proposed to deal with slow and fast time-varying wind conditions. In [25], a switching model predictive attitude controller for an unmanned aerial vehicle subject to atmospheric disturbances is presented. In [26], a nonlinear adaptive state feedback controller for thrust and torque actuation is designed, so that it guarantees global convergence of the closed-loop path following in the presence of constant wind disturbances. In [27], a sliding mode control driven by sliding mode disturbance observer (SMC-SMDO) approach is used to design a robust flight controller for a small quadrotor.

So far, the studies are mainly focused on payload variations and on wind disturbances rejection. Only few recent papers have been found that address both problems at the same time, such as in [16]. This challenging issue demands further research. Furthermore, although there are some papers where neural networks are applied to model quadrotors [28, 29], and to control them [30–33], these techniques have not been explored to solve this specific research problem.

Therefore, in this work, we propose the design of an intelligent control strategy based on neural networks to cope with these external disturbances, payload changes and wind, that can affect quadrotor dynamics. The final goal is to stabilize the UAV and to improve the system performance. The control strategy has been extensively tested by simulation with different trajectories. Indeed, the online learning estimator that has been implemented increases the robustness of the controller, reducing the effects produced for these variations.

The paper is organized as follows. In Section 2, the dynamic behavior of the system is described. The design of the controller and the adaptive neural estimators are presented in Section 3. Simulation results are discussed in Section 4. The document ends with the conclusions and future works.

2. System Model

A quadrotor vehicle is composed by four perpendicular arms, each one with a motor and a propeller (Figure 1, left). The four motors drive the lift and direction control.

The UAV absolute position is described by three coordinates, (x, y, z), and the attitude is given by the three Euler's angles (ϕ, θ, ψ), under conditions: $(-\pi \leq \psi < \pi)$ for the yaw angle, $(-\pi/2 \leq \phi < \pi/2)$ for the roll, and $(-\pi/2 \leq \theta < \pi/2)$ for the pitch, all angles in radians.

The system is based on two couples of propellers opposed to each other, (1, 3) and (2, 4) (Figure 1, right). To keep the balance of the system, one pair of motors turns clockwise while the other one spins counterclockwise. The increment of the speed of rotor 3 with respect to rotor 1 produces a positive pitch ($\theta > 0$), while increasing the speed of rotor 4 regarding rotor 2 produces a positive roll ($\phi > 0$). The increment of the speeds of rotors 1 + 3 with respect to rotors 2 + 4 produces a positive yaw ($\psi > 0$).

Regarding the UAV modelling, in the related literature, there are typically two approaches to obtain the mathematical model of the quadrotor: the Lagrangian method and the one based on the representation of the translational and angular dynamics. As we need to develop a control-oriented model, we have used the latter, that is, the Newton-Euler method, which describes the dynamic systems in terms of force and momentum.

The Newton dynamic equation states that the sum of forces applied in a system is equal to the variation of the lineal momentum:

$$\sum F_i = \frac{d}{dt}(mv). \tag{1}$$

While the mass is constant, this equation is equivalent to

$$\sum F_i = m\dot{v}. \tag{2}$$

In our case, the forces F_i are the vector of forces T produced by the rotors and by the gravity, considering an Earth reference system; thus, the previous equations give the translational dynamic (3) considering the assumption $m\dot{v} \gg \dot{m}v$, implicitly assumed in other papers [33]:

$$m\dot{v} = RT - mge_3, \tag{3}$$

where m is the mass of the quadrotor in Kg, R is the rotation matrix which is dimensionless, g is the gravitational acceleration in $\mathrm{m{\cdot}s^{-2}}$, T is a vector of forces in N, and $e_3 = [0, 0, 1]^T$ is a unit vector which describes the rotor orientation.

The Euler dynamic equation expresses that the sum of torques is equivalent to the variation of the angular moment:

$$\sum \mu_i = \frac{d}{dt}(J \times w) = \frac{d}{dt}J \times w + J \times \frac{d}{dt}w. \tag{4}$$

This sum of torques represents the vector of torques τ (N·m) in the three axes produced by the rotors, that is, the angular dynamic of system (5) [34], considering the assumption $J \times (d/dt)w \gg (d/dt)J \times w$, also implicitly assumed in other papers [33]:

$$\tau = J\dot{\omega} + \omega \times J\omega, \tag{5}$$

where J is the inertia tensor in $\mathrm{Kg{\cdot}m^2}$ (6), ω is the angular velocities vector in rad/s, and $\times$ represents the vector product:

$$J = \begin{pmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{pmatrix}. \tag{6}$$

The vectors τ (7) and T (8) are a function of the velocities of the propellers:

$$\tau = \begin{pmatrix} bl\left(\Omega_4^2 - \Omega_2^2\right) \\ bl\left(\Omega_3^2 - \Omega_1^2\right) \\ d\left(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 - \Omega_3^2\right) \end{pmatrix} \tag{7}$$

$$T = \begin{pmatrix} 0 \\ 0 \\ b\left(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2\right) \end{pmatrix}, \tag{8}$$

where b is the thrust coefficient in $\mathrm{N{\cdot}s^2}$, d is the drag coefficient, l is the longitude of each arm in m, and $\Omega_1, \ldots, \Omega_4$ are the velocities in rad/s of the rotors 1 to 4, respectively.

To simplify the calculations, instead of using the speed of the rotors, it is possible to define a set of control signals u_1, u_2, u_3 y u_4 as follows (9):

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -1 & 0 & 1 \\ -1 & 0 & 1 & 0 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix}. \tag{9}$$

This matrix is invertible, so it is possible to generate speed references for the rotors from a set of control signals.

Finally, from (1) to (9), the following system of equations is derived:

$$\ddot{\phi} = \frac{\dot{\theta}\dot{\psi}\left(I_y - I_z\right)}{I_x} + \left(\frac{lb}{I_x}\right)u_2 \tag{10}$$

$$\ddot{\theta} = \frac{\dot{\phi}\dot{\psi}\left(I_z - I_x\right)}{I_y} + \left(\frac{lb}{I_y}\right)u_3 \tag{11}$$

$$\ddot{\psi} = \frac{\dot{\phi}\dot{\theta}\left(I_x - I_y\right)}{I_z} + \left(\frac{d}{I_z}\right)u_4 \tag{12}$$

$$\ddot{X} = -\left(\sin\theta\cos\phi\right)\left(\frac{b}{m}\right)u_1 \tag{13}$$

$$\ddot{Y} = \left(\sin\phi\right)\left(\frac{b}{m}\right)u_1 \tag{14}$$

$$\ddot{Z} = -g + \left(\cos\theta\cos\phi\right)\left(\frac{b}{m}\right)u_1. \tag{15}$$

The constants of (10) to (15) that are used during the simulations are listed in Table 1. The values have been extracted from [35].

3. Control Strategy Design

3.1. First Approach: Adaptive Inverse Controller. There are different control strategies based on neural networks [29–34]. In a first approach [32], a variant of the generalized learning algorithm (GLA) was used to control this system (Figure 2). The procedure was as follows.

The first step is the application of the GLA algorithm to offline train the neural network in order to identify the inverse dynamic of the plant (Figure 2). Once the network has been offline trained, it is placed in cascade connection with the plant and a PID controller. In Figure 3, this control strategy is shown for the altitude variable of the UAV. Then, the configuration of the network is online refined. In order to do this, during each control interval, two processes are sequentially applied to the network (first the simulation, later the online learning):

(1) **Simulation:** The output of the PID, $PID_{OUT}(t_i)$, feeds one of the inputs of the artificial neural network; the

Table 1: Parameter values of the model.

Parameter	Description	Value/Units
l	Longitude of an arm	0.232 m
m	Mass of the quadrotor	0.52 Kg
d	Drag coefficient	$7.5e^{-7}$ N.m.s^2
b	Thrust coefficient	$3.13e^{-5}$ N.s^2
I_x	Inertia in X	$6.228e^{-3}$ Kg.m^2
I_y	Inertia in Y	$6.225e^{-3}$ Kg.m^2
I_z	Inertia in Z	$1.121e^{-2}$ Kg.m^2
ρ_{air}	Density of the air	1.2 Kg/m^3
A	Area in the direction of the wind	0.0186 m^2
Cd	Wind drag coefficient	1

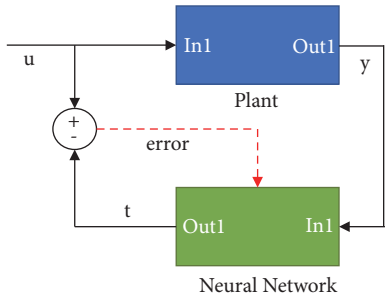

Figure 2: Offline training to identify the plant inverse dynamic.

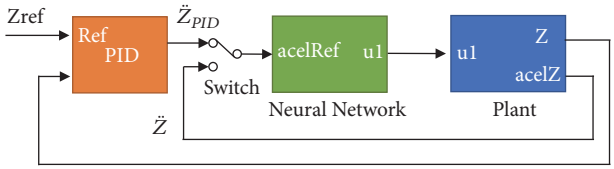

Figure 3: Adaptive altitude neurocontrol strategy.

rest of the inputs are past values of the plant output, $PLANT_{OUT}(t_i - j * Ts)$. The network generates the control input, u_1 (16), which is the input of the plant (Figure 3, switch in the upper position):

$$u_1(t_i) = f_{NET}(PID_{OUT}(t_i), PLANT_{OUT}(t_i - j * Ts), par_{NET}(t_i - Ts)) \quad j = 1 \ldots (N_{inputs} - 1), \tag{16}$$

where Ts is the sampling time in seconds, par_{NET} denotes the configuration parameters of the network, and N_{inputs} means the number of inputs of the neural network.

(2) **Online learning:** The neural network is trained again with the current and previous outputs of the plant, in order to generate the control output, u_1, obtaining the new configuration parameters par_{NET} (17). The network input dataset is made up of the past values of the plant output, $PLANT_{OUT}(t_i - j * Ts)$. The output dataset is the current value of the plant input $u_1(t_i)$ (Figure 3, switch in the lower position):

$$par_{NET}(t_i) = f(PLANT_{OUT}(t_i - j * Ts), u_1(t_i), par_{NET}(t_i - Ts)), \quad j = 0 \ldots (N_{inputs} - 1). \tag{17}$$

In order to test the validity of this first approach, we firstly focused only on the altitude control. UAVs are normally provided with accelerometers, so it is assumed that the acceleration in the z-axis ($\ddot{Z}$) is available. The network must be able to simulate the control signal u_1 by using acceleration measurements.

In this example, PID_{OUT} (16) is the reference of the acceleration in the z-axis, $\ddot{Z}_{PID}$; $PLANT_{OUT}$ is the acceleration in the z-axis in m·s^{-2}, and u_1 is the control signal. Thanks to the artificial neural network, the PID does not need to include the plant gain. The network is able to learn the plant gain and work with it. In other words, with this approach, it is not necessary to know the system parameters to control it [32].

3.2. New Advanced Strategy: Controller with Adaptive Neural Estimators. The generalized learning algorithm proposed in the previous section is especially useful when there is not any knowledge about the real dynamic of the plant (black-box system). However, when some knowledge about the system dynamics is available, even if it is not complete or accurate, it is positive to include it in some way in the controller. In this section, the system equations ((10) to (15)) are introduced in the controller, and the neural networks are focused on the uncertain terms: the mass, the wind disturbance, and the nonmodelled dynamics.

The control system has been designed to track trajectories defined by tuples of three coordinates $(Xref, Yref, Zref)$.

The model of the whole system with the controllers is shown in Figure 4. For the sake of simplicity, several ports that route the signals haven been introduced to reduce the number of lines in the diagram.

The UAV model control inputs are the four control signals, u_i, $i = 1, .., 4$, that represent the power of the rotors. Four main controllers are defined to obtain these model inputs, the controllers of the X, Y, and Z coordinates, and

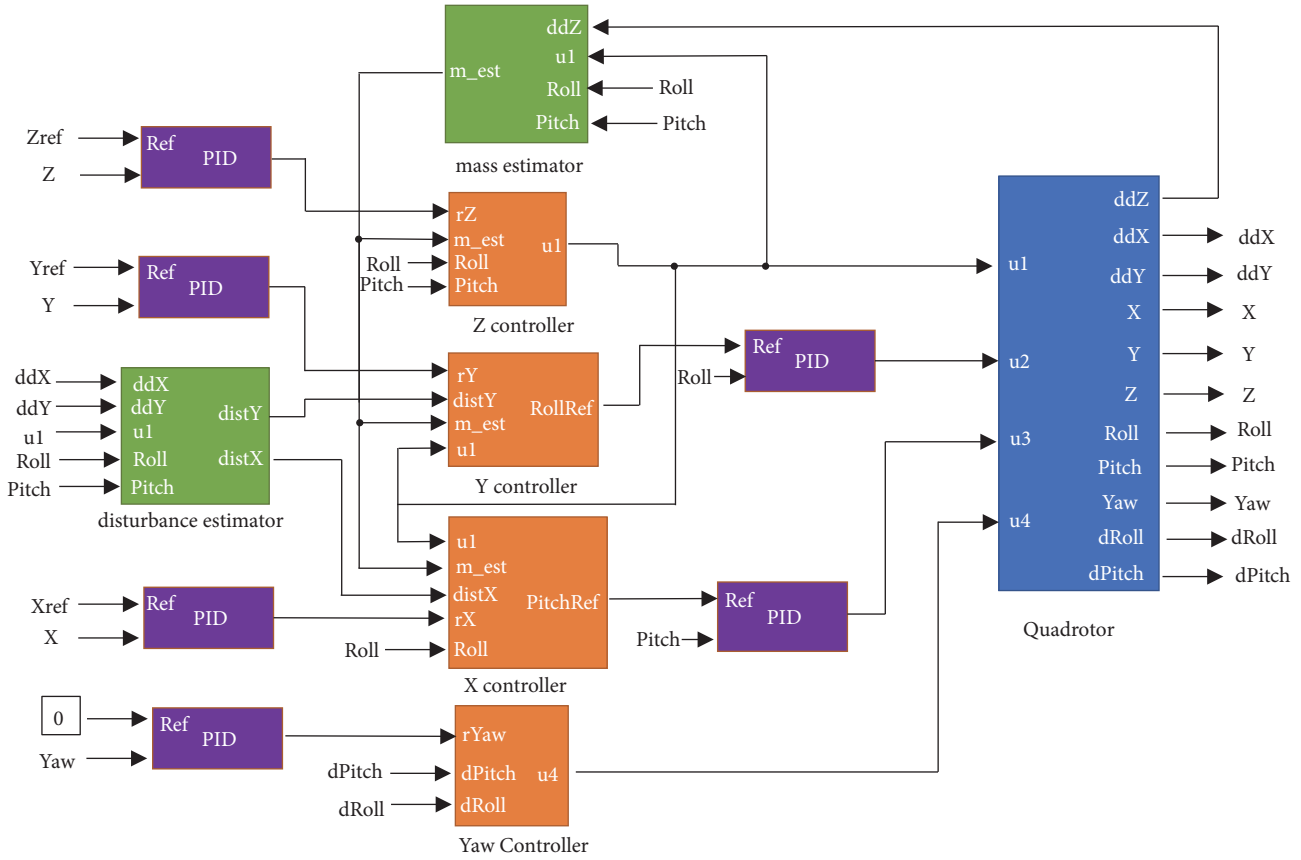

Figure 4: Control system with mass and disturbance estimators.

another for the yaw angle. This is because the pitch and roll angles are used to track the *Xref* and *Yref* reference coordinates. The control of Z is carried out by the control signal u_1. The Y controller generates the roll reference, and the tracking of the roll angle is performed by the control signal u_2. In the same way, the controller of X generates the pitch angle reference value, and control signal u_3 will be in charge of getting this value. The control signal u_4 is used to stabilize the yaw angle around zero.

The aim of the PIDs controllers that appear in Figure 4 is to generate the acceleration references to make the attitude error $(\phi_r - \phi, \theta_r - \theta, \psi_r - \psi)$ and the tracking error converge to zero. The rest of the controllers are used to compensate the nonlinearities of the system. The PID tuning parameters have been set by trial and error and the values of the gains are shown in Table 2.

A mass estimator and a wind disturbance estimator have been added (Figure 4). They are implemented by neural networks. These estimators are used to compensate the variations of the system's mass and the influence of the wind disturbances. These adaptive neural estimators feed the inputs of the X, Y, and Z controllers.

The controllers are defined as follows.

The performance of the Z controller is given by

$$u_1(t_i) = \begin{cases} (rZ(t_i) + g)\left(\dfrac{m_{est}(t_{i-1})}{b}\right) & \cos\theta_{i-1}\cos\phi_{i-1} = 0 \\ (rZ(t_i) + g)\left(\dfrac{m_{est}(t_{i-1})}{b}\right)\left(\dfrac{1}{\cos\theta_{i-1}\cos\phi_{i-1}}\right) & \cos\theta_{i-1}\cos\phi_{i-1} \neq 0, \end{cases} \tag{18}$$

where rZ is the output of the Z PID controller, m_{est} is the estimation of the mass in Kg, $(\phi_{i-1}, \theta_{i-1})$ represents the roll and pitch signals at time t_{i-1}, and the rest of the parameters and variables have been previously defined. In (18), it is possible to see how there is a discontinuity at $\cos\theta\cos\phi = 0$ that has been taken into account.

Table 2: PID parameters for each variable.

Signal	Kp	Kd	Ki
X	8	8	0
Y	8	8	0
Z	2	2	0.9
Roll	8	8	0
Pitch	8	8	0
Yaw	8	8	0

The Y controller is defined by

$$\phi_R(t_i) = \begin{cases} \phi_R(t_{i-1}) & u_1(t_i) = 0 \\ \operatorname{asin}\left(\left(rY(t_i) + distY_{est}(t_{i-1}) \right) \left(\frac{m_{est}(t_{i-1})}{bu_1(t_i)} \right) \right) & u_1(t_i) \neq 0, \end{cases} \tag{19}$$

where ϕ_R is the reference of the roll PID controller in rad, rY is the output of the Y PID controller in $\mathrm{m{\cdot}s^{-2}}$, $distY_{est}$ is the estimation of the wind disturbance in the y-axis in $\mathrm{m{\cdot}s^{-2}}$, and the rest of the parameters and variables have been already defined. The discontinuity at $u_1(t_i) = 0$ has been considered.

The control of the X coordinate is given by the expression:

$$\theta_R(t_i) = \begin{cases} \theta_R(t_{i-1}) & u_1(t_i)\cos\phi_{i-1} = 0 \\ -\operatorname{asin}\left(\left(rX(t_i) + distX_{est}(t_{i-1}) \right) \left(\frac{m_{est}(t_{i-1})}{bu_1(t_i)\cos\phi_{i-1}} \right) \right) & u_1(t_i)\cos\phi_{i-1} \neq 0, \end{cases} \tag{20}$$

where θ_R is the reference of the pitch PID controller, rX is the output of the X PID controller in $\mathrm{m{\cdot}s^{-2}}$, $distX_{est}$ is the estimation of the wind disturbance in the X coordinate in $\mathrm{m{\cdot}s^{-2}}$, and the rest of the parameters and variables have been already cited. Again, there is a discontinuity at $u_1(t_i)\cos\phi_{i-1} = 0$.

3.3. Adaptive Neural Estimator for Disturbances. The control scheme proposed in Figure 4 of this paper uses three different estimations: the estimation of the total mass of the system, the estimation of the disturbances in the X coordinate, and the estimation of the disturbances in the Y coordinate. These approximations, as may be observed in (18)-(20), feed the inputs of the different nonlinear controllers in order to reject the effect of the changes in the mass (m_{est}) and the wind external disturbances ($distX_{est}$ and $distY_{est}$).

These parameters that affect the dynamic of the UAV must be estimated if there are no sensors that could measure these disturbances, as it is the case. Although the estimation of each one of these variables (m_{est}, $distX_{est}$, and $distY_{est}$) has been implemented in a different function block, the inner structure of all of them is the same. They differ in some parameters of the configuration and the input and output signals. Thus, they can be jointly explained.

The **estimator** is based on an artificial neural network with online learning. Thus, there is one neural network to, let say, model each of the three parameters or disturbances considered. The following figure represents its generic structure (Figure 5). It is based on the diagram shown in [28], but in this case a new parameter model has been included. Furthermore, in this case, the inputs of the neural network are the outputs of the UAV, and its output are the output of the parameter model (when in [28], the output of the neural network is fed by the outputs of the UAV).

The parameter model receives the inputs and outputs of the UAV. The output of the parameter model is used as target output of the neural network during the training. The output of the proposed estimator is always the output of the neural network. The parameter model is needed because we are using supervised artificial networks and the inputs, and their corresponding outputs must be known.

We will use one parameter model for the mass, one parameter model for the wind disturbance in the x-axis, and another for the y-axis. The complete process is further explained below.

Each element of the dataset used to offline train the network is composed of the following:

(i) Target: Parameter (disturbance) value calculated with the model at instant t_i, param(t_i)

```
Net= configureNet ()
Dataset= ∅
For i=1 to toff        #Generating the dataset for off-line learning
     inputNet =[out1(i-1),... out1(i-n1), out2(i-1),..., outM (i-1).... outM (i- nM )]
     ParM (i)=model(out1(i-1),....., outM (i-1)])
     Element = { parM (i), inputNet }
     Dataset=Dataset ∪ Element
endFor
Net= offlineTraining (Net, Dataset)
For i= toff to tend      #on-line learning
     inputNet =[out1(i-1),... out1(i-n1), out2(i-1),..., outM (i-1).... outM (i- nM )]
     ParS (i)=simulate(Net, inputNet )
     [out1(i),....., outM (i)]= executeUAV ( ParS (i))          #Execute UAV + Controller
     ParM (i)=model(out1(i-1),....., outM (i-1)])
     If ∃ ParM(i) then
          Element= { parM (i), inputNet }
          Net= onlineTraining (Net, Element)
     endIf
endFor
```

Pseudocode 1

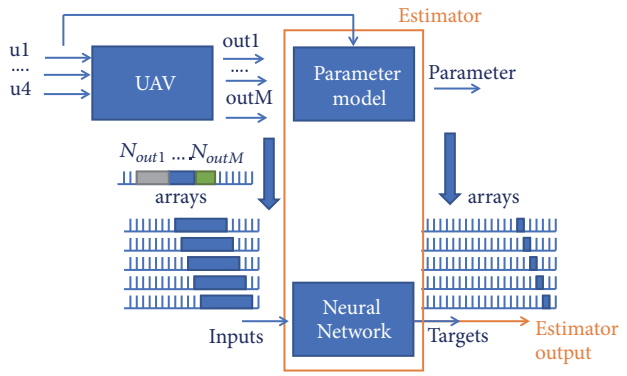

Figure 5: Neural estimator structure and configuration.

(ii) Network inputs: for each output signal i, the previous N_i values to t_i are collected and structured as an array. A theoretical example would be the following: At t_i=10 and for M = 3 outputs with a configuration Nout1=3, Nout2=2, Nout3=1, the network inputs are [out1(7), out1(8), out1(9), out2(8), out2(9), out3(9)] and the target is: param(10).

If the parameter cannot be calculated (division by zero, squared root of negative numbers, or any other singularity), that element is not included into the training dataset.

In this offline learning, the training dataset has as many elements as previous instants of time are considered. Longer time will normally produce better accuracy but will require more computational effort. The selection of the data for training is a delicate task. Indeed, the accuracy of the model depends on the data used to train the network. Another disadvantage of exclusively using offline approaches is that they do not capture the dynamics when it is changing over time [28]. For these reasons, in our proposal, we use adaptive learning for the mass and wind disturbance estimators' calculation.

Once the offline learning has finished, at each instant of time, a new training element is added {target, network inputs}, as has been previously explained. The target is obtained as the output of the model and the network inputs the output signals of the UAV. This new element is used to teach the networks how to adapt its parameters according to the new input. That is, the function to estimate the parameter is continuously changing over time.

Pseudocode 1 details the algorithm which relates to offline learning, the simulation, the online learning, and how the parameters are updated.

As it has been commented before, when it is not possible to measure or to calculate the parameter with the model, the input values are not defined, or there is a singularity in the calculation, the artificial neural network of our approach is very useful to estimate the disturbance. In these situations, we could say roughly speaking that the network generates new knowledge.

Once the inner structure of the estimator has been introduced, we explain the specific configurations for the mass and the wind disturbances.

Mass Estimation

(i) Target output: The mass is approximated with the model:

$$\widetilde{m}(t_i) = \frac{\ddot{Z}(t_{i-1}) + g}{u_1(t_{i-1}) \cos\theta_{i-1} \cos\phi_{i-1}}. \quad (21)$$

(ii) Network inputs:

(a) Acceleration in the z-axis $\ddot{Z}(t_{i-1})$

(b) Roll and pitch cosine angles multiplied $(\cos\theta_{i-1} \cos\phi_{i-1})$.

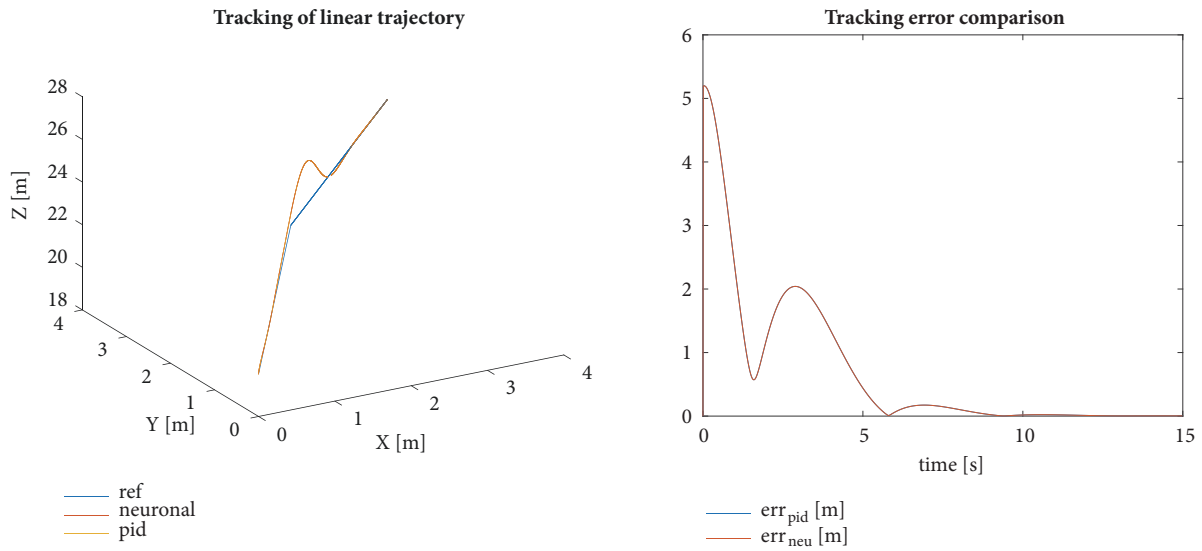

Figure 6: UAV tracking of a linear trajectory (left) and its tracking error (right).

Disturbance in X-Axis Estimation

(i) Target output: The disturbance is approximated with the model:

$$\widetilde{distX}(t_i) = -(\sin\theta_{i-1}\cos\phi_{i-1})\frac{bu_1(t_{i-1})}{m_{est}(t_{i-1})} - \ddot{X}(t_{i-1}). \quad (22)$$

(ii) Network inputs:

(a) Acceleration in the x-axis $\ddot{X}(t_{i-1})$
(b) Roll angle θ_{i-1}
(c) Pitch angle ϕ_{i-1}.

Disturbance in Y-Axis Estimation

(i) Target output: The disturbance is approximated with the model:

$$\widetilde{distY}(t_i) = \sin\phi_{i-1}\frac{bu_1(t_{i-1})}{m_{est}(t_{i-1})} - \ddot{Y}(t_{i-1}). \quad (23)$$

(ii) Network inputs:

(a) Acceleration in the y-axis $\ddot{Y}(t_{i-1})$
(b) Roll angle θ_{i-1}
(c) Pitch angle ϕ_{i-1}.

For the three different estimators, the artificial network implemented is a multilayer perceptron (MLP) with a hidden layer. The number of neurons of the hidden layer has been set to 20. The Levenberg-Marquardt algorithm with μ=0.001 has been used for the training. The network is offline trained for the first 2 seconds and then the online learning is applied for the remaining 13 seconds.

4. Results and Discussion

Simulation results have been obtained with Matlab/Simulink software. The duration of each simulation is 15 s. The controller is offline trained during the first 2 s. Then, the online learning algorithms are applied for the remaining 13 s. The sample time Ts is set to 10 ms.

In the experiments, in order to simplify the system, the yaw angle is set to 0, but it could be set to any other value by the user.

4.1. Trajectory Tracking without Disturbances. The control system has been first tested and validated with several trajectories without considering any disturbances. For each trajectory, the path followed by the UAV with adaptive neural estimators (red line), without them (yellow line), the reference (blue line), and the tracking error, are shown (Figures 6–11). The tracking error is calculated by the following equation:

$$tERR(t_i) = \sqrt{(X_i - Xref_i)^2 + (Y_i - Yref_i)^2 + (Z_i - Zref_i)^2}, \quad (24)$$

where $Xref_i, Yref_i, Zref_i$ are the references for X, Y, and Z in t_i, respectively.

The trajectories used to validate the controller are described by the following equations, with the corresponding parameters:

Linear:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.2t + 1 \\ 0.2t + 1 \\ 5 * step(t) + Z_0 \end{bmatrix}. \quad (25)$$

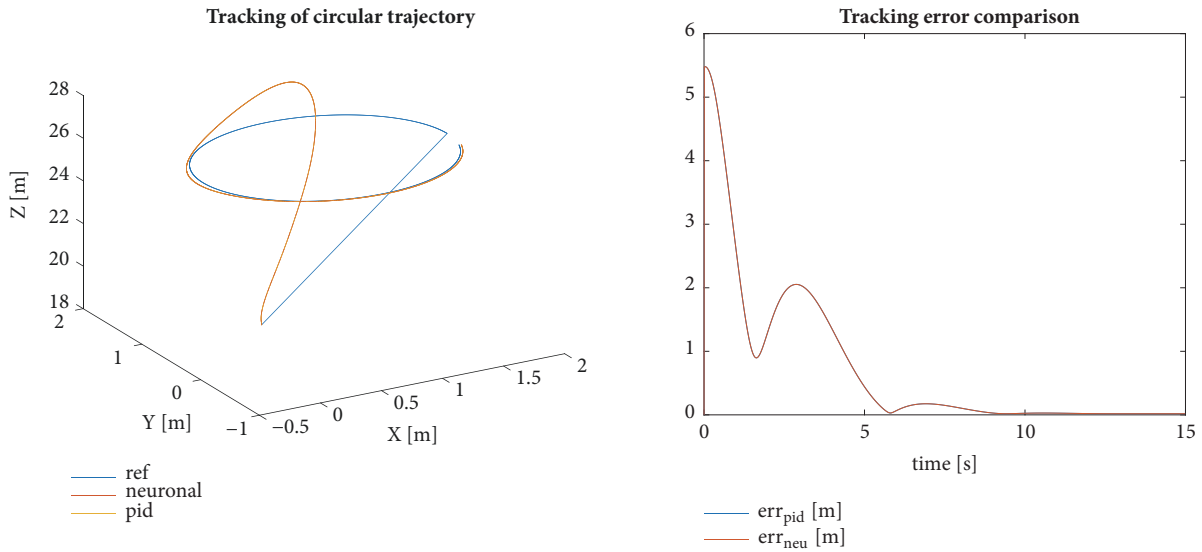

Figure 7: UAV tracking of a circular trajectory (left) and its tracking error (right).

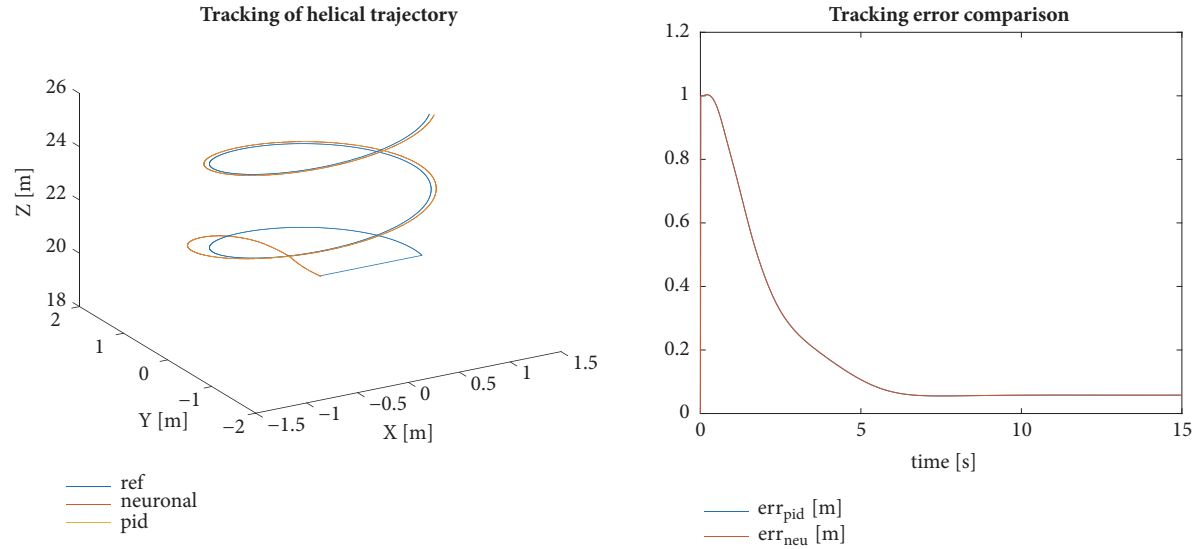

Figure 8: UAV tracking of a helical trajectory (left) and its tracking error (right).

Circular:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \cos(0.4t) + 1 \\ \sin(0.4t) + 1 \\ 5 * step(t) + Z_0 \end{bmatrix}. \tag{26}$$

Helical:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \cos(0.8t) \\ \sin(0.8t) \\ 0.4t + Z_0 \end{bmatrix}. \tag{27}$$

Cyclic helical:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \cos(1.6t) \\ \sin(1.6t) \\ \sin(0.4t) + Z_0 \end{bmatrix}. \tag{28}$$

Lemniscate:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \dfrac{\cos(0.4t)}{1 + \sin(0.4t)^2} \\ \dfrac{\cos(0.4t)\sin(0.4t)}{1 + \sin(0.4t)^2} \\ Z_0 \end{bmatrix}. \tag{29}$$

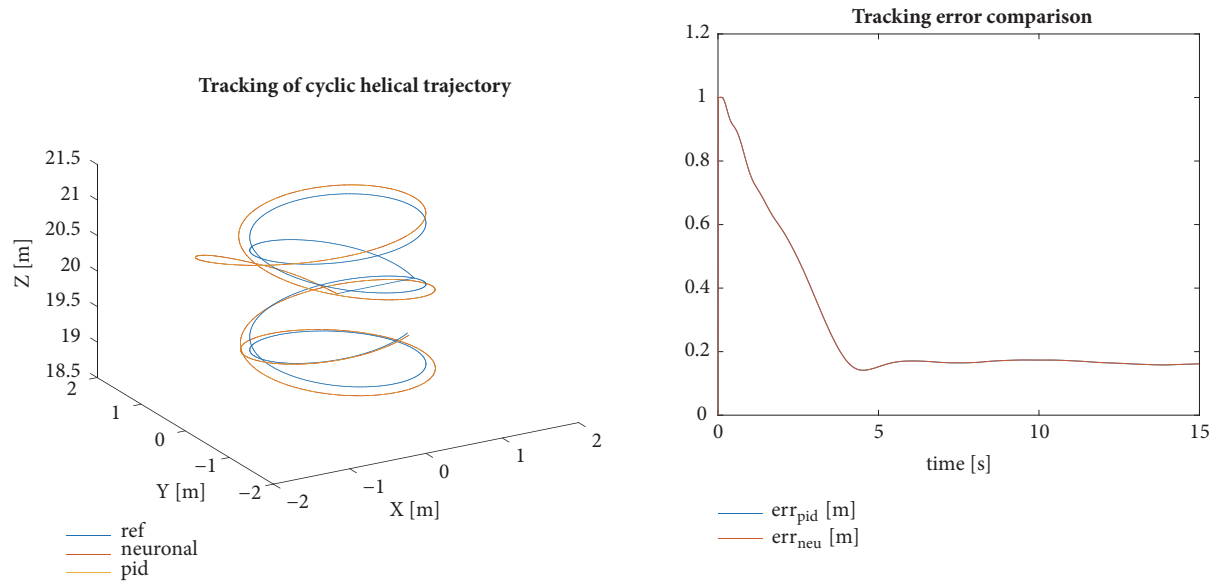

Figure 9: UAV tracking of a cyclic helical trajectory (left) and its tracking error (right).

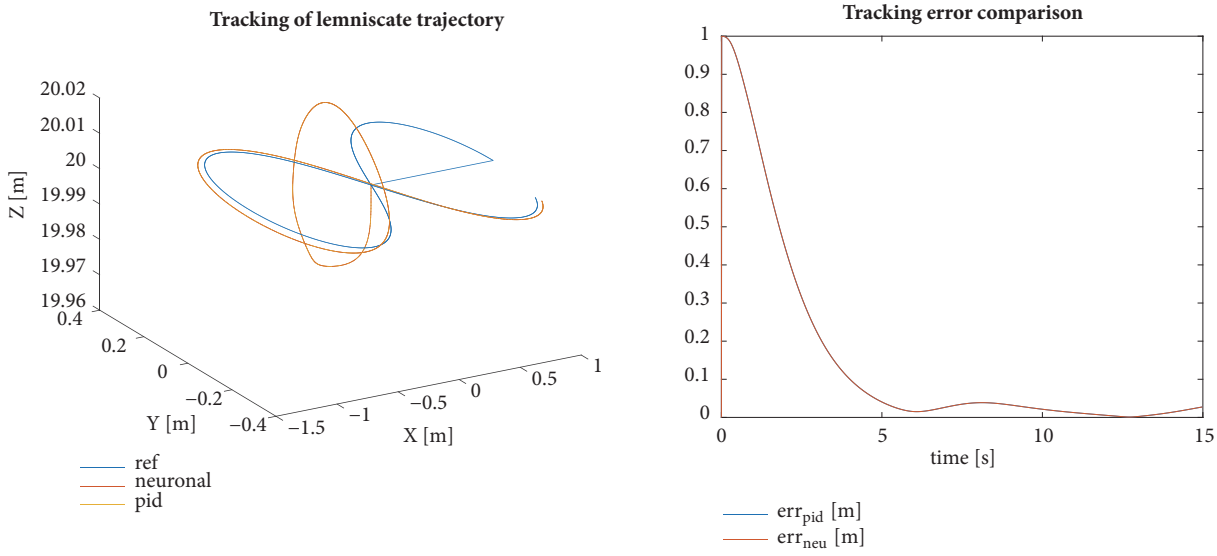

Figure 10: UAV tracking of a lemniscate trajectory (left) and its tracking error (right).

Helical lemniscate:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \dfrac{\cos(0.8t)}{1+\sin(0.8t)^2} \\ \dfrac{\cos(0.8t)\sin(0.8t)}{1+\sin(0.8t)^2} \\ 0.1t + Z_0 \end{bmatrix}. \quad (30)$$

In Figures 6–11, it is possible to observe how the tracking error decreases along the time. In t = 0, the error is high because the value of the reference starts in a very different value than the initial position of the system, that is, $(X_0, Y_0, Z_0) = (0, 0, 20)$. Then, the controller starts to work, and the tracking error is reduced.

For example, in the linear trajectory (Figure 6), the reference in t_0 is (1, 1, 25); thus, the initial tracking error at t=0 is $\sqrt{(1+1+25)}$ =5.196.

For the same reason, the initial error of the helical and lemniscate trajectories is much lower. The initial value reference is (1, 0, 20), and thus the initial error is 1.

In these figures, there is not any significant visual difference between the results with and without the neural network due to the fact that if the PIDs are well tuned and there are no changes either in the mass or in the external disturbances, as it is the case in this section, the neural networks do not provide relevant advantages. Nevertheless, the networks in the control scheme of Figure 4 show their full potential, when they must

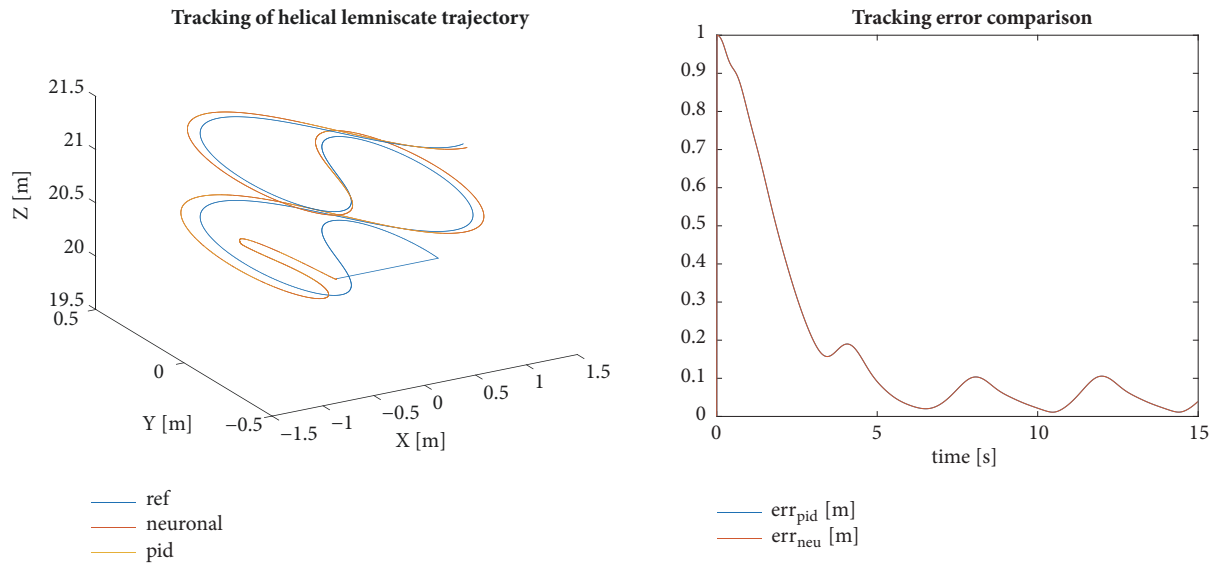

Figure 11: UAV tracking of a helical lemniscate trajectory (left) and its tracking error (right).

tackle disturbances, as it can be seen in the following sections. Consequently, the red and yellow lines seem to be overlapped.

Other appreciable result is that the stationary tracking error is higher when the trajectory has helical component. Predictably, the circular and helical trajectories show a cyclic nature.

As a conclusion, the controller performs a good tracking for a wide range of different trajectories. Thus, it can be said that the control strategy has been validated.

4.2. Control Robustness with Mass Variations

4.2.1. Mass Disturbance Model. Adding a payload in the quadrotor typically has three effects: the total mass of the system is increased, and the centre of the gravity can be modified and therefore also the inertia. In this work, we assume that the payload is an isotropic symmetric rigid solid attached to the UAV, not suspended, with dimensions much smaller than the dimensions of the quadrotor. The distance between the centre of gravity of the UAV and the centre of gravity of the load depends on the shape and the weight distribution of the manipulator and the shape and weight distribution of the load. In our experiment, we assume that this distance is zero. Under these circumstances, the effect of the inertia and centre of gravity variation can be neglected. Therefore, our paper is only addressing the effect of the mass variation.

When the UAV is performing a logistic task, there are two possible stable states regarding the mass: one is before the load is in contact with the quadrotor, and then only the mass of the UAV is considered; the second one is when the payload mass is part of the system, and the sum of both masses is then considered as an only system. Between these two states, there may be several profiles of mass variation depending on the grasping technology and the properties of the surface of the load.

We assume that the grasping and the load are nondeformable; thus, one step profile may be applied regarding the mass disturbance, as in other papers [16].

The mass variation is simulated by adding a new term $dist_m$ (34) to (13) to (15), resulting in (31) to (34). The modelling of the mass disturbance is a step function. The total mass then is triplicated at t = 4s (34) meaning that a payload has been attached to the UAV. After 4 sec, the total mass of the system is $m + m_L = 3m$, with m being the mass of the quadrotor and the mass of the load $m_L = 2m$. Even if it can be considered a simple approach of dealing with this disturbance, the final effect is well represented:

$$\ddot{X} = -(\sin\theta\cos\phi)\left(\frac{b}{(m + dist_m)}\right)u_1 \tag{31}$$

$$\ddot{Y} = (\sin\phi)\left(\frac{b}{(m + dist_m)}\right)u_1 \tag{32}$$

$$\ddot{Z} = -g + (\cos\theta\cos\phi)\left(\frac{b}{(m + dist_m)}\right)u_1 \tag{33}$$

$$dist_m = m_L \cdot step\,(t - 4)\,. \tag{34}$$

This experiment represents a possible situation while performing a logistic task, where the payload is heavier than the quadrotor itself.

4.2.2. Simulation Results with Mass Variation. Now, we test the control proposal with mass disturbances for the helical lemniscate trajectory because it is the most challenging one.

Figure 12, left, shows the reference trajectory in blue, the trajectory obtained by the controller with the neuromass estimator in red, and without the mass estimation in yellow (only PID). It is possible to observe how the trajectory obtained without the adaptive neural estimators (yellow line) moves away the reference even if it later comes closer to it.

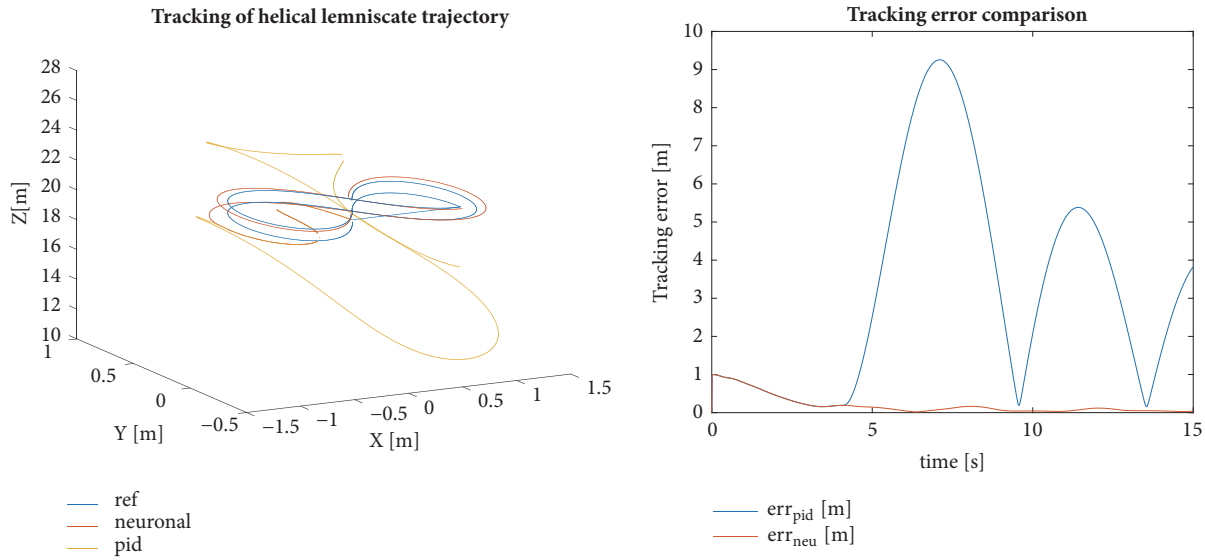

FIGURE 12: UAV tracking of a helical lemniscate trajectory with mass variation (left) and its tracking error (right).

At the right side of Figure 12, the tracking error is presented (red line with mass estimation and blue line without mass estimation). In both figures, it is possible to observe how the performance with the mass estimator is much better. The tracking error is mostly the same until the mass changes at t = 4 s, but then the error without mass estimation increases significantly.

The trajectory tracking of each coordinate has been represented in a separated figure to study the performance of the controller. Figure 13, top left, shows the tracking of the X coordinate, at the top, right, the tracking of the Y coordinate, and at the bottom the tracking of the Z coordinate, with reference (magenta line), the trajectory with mass estimator (red line), and the trajectory without mass estimator (blue line). For every coordinate, it is possible to see how the trajectory obtained by the controller with the mass estimator fits better the reference. Even though, the deviation mainly comes from the Z coordinate due to the fact that the acceleration in the z-axis is more sensible to changes in the mass.

Figure 14 shows the mass obtained by the neuromass estimator in this experiment (real mass in blue and the estimated mass in red). The estimate of the mass is very similar to the real one but slightly noisy. Clearly, it can be appreciated that the mass is triplicated following a step variation profile.

The controller is robust even with other types of variations in the mass like a sine function, for instance, if the mass changes according to

$$\begin{aligned} m(t) &= m_{model}\left(step(t) - step(t-4)\right) \\ &\quad + step(t-4)\left(2 \cdot m_{model} + m_{model} \cdot \sin\left(\frac{2\pi}{8}t\right)\right). \end{aligned} \tag{35}$$

The results of considering this mass variation profile are shown in Figure 15. The tracking error of each component is like the previous ones, but the system experiences a stronger change with the sinusoidal reference profile. In this case, the tracking error of the system without the adaptive neural estimators tends to increase; meanwhile, in the case of the step change, the tracking error decreased over the time.

Therefore, it can be seen how the controller with the neural estimator also works well with different mass change profiles.

Figure 16 shows the mass obtained by the neuromass estimator (red line) in this case and the corresponding real mass (blue line). The estimate of the mass is again very similar to the real one but slightly noisy but now the sinusoidal nature of the profile can be clearly observed.

The controller has been extensively tested for different trajectories and the MSE numerical results are summarized in Table 3 for the different trajectories and for each coordinate, with the neural estimator (neuro) and without it (PID). The last column shows the absolute error. The best result for each component and trajectory is boldfaced.

Table 4 compares the values of the maximum error in this experiment for the same cases with before.

In Tables 3 and 4, it is possible to observe that the controller with adaptive neural estimators provides less or equal tracking error for every tested trajectory. A general trend in the "PID column" (without mass disturbance estimation) is that the worst tracking error is obtained for the Z component. One of the reasons of this may be the fact that the maximum amplitude of the reference signal is in the z-axis.

4.3. Control Robustness with Wind Disturbances

4.3.1. Wind Disturbance Model. The effect of the wind in an UAV in flight can be considered as a drag force and torque

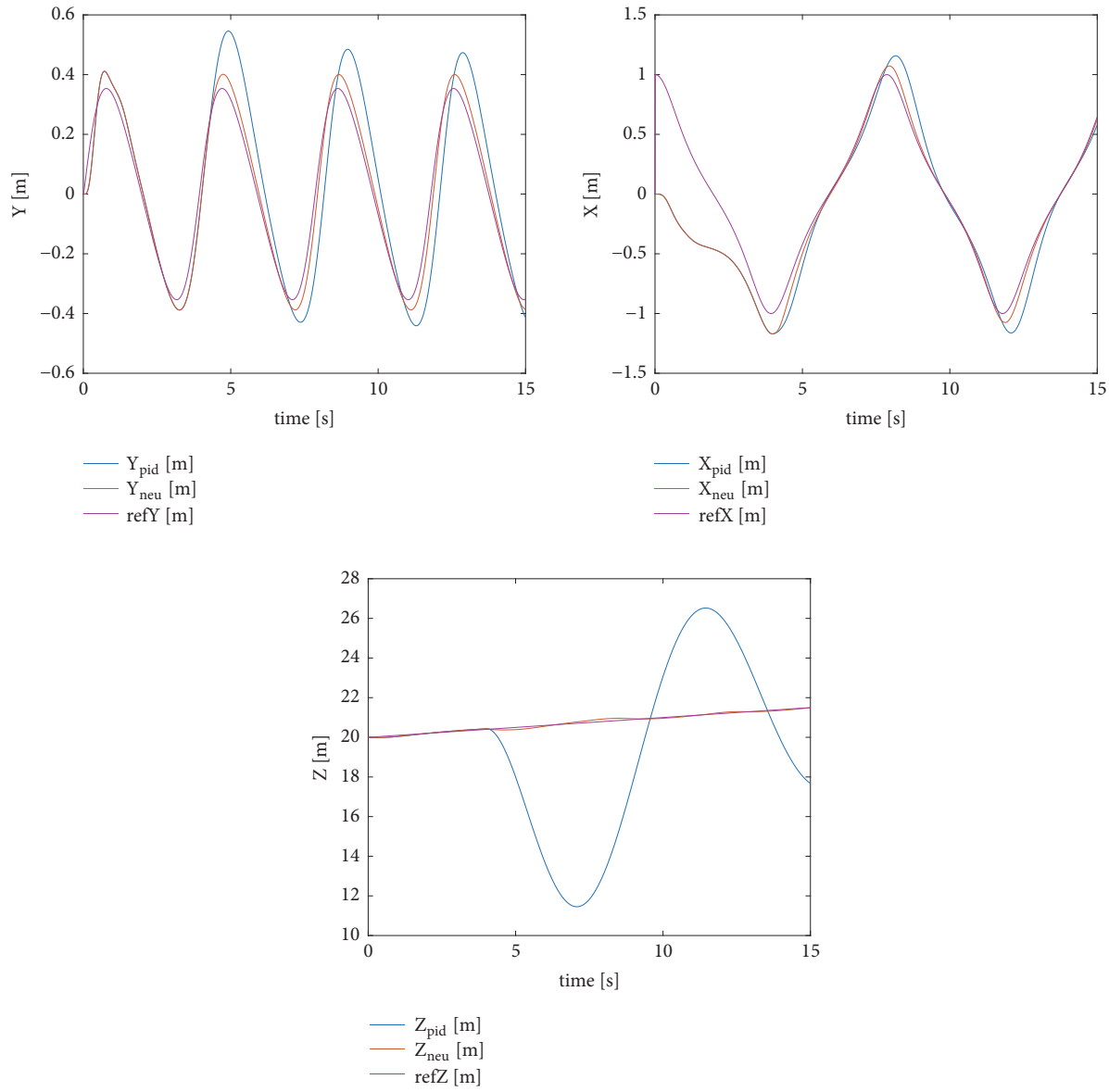

Figure 13: UAV tracking trajectory of X, Y, and Z coordinates, respectively, with mass variation.

depending on the aerodynamics properties of the object and the characteristics of the air flow.

The wind speed is a vector field; that is, its value may be different in each coordinate (x, y, z) of the space. For small UAVs like the one we are using in the experiments, we can assume a planar air flow, so the wind is the same in the region of the space where the UAV is flying. Under these circumstances, the wind influence in the torque can be neglected.

On the other hand, the following equation represents the drag force [36]:

$$\overline{F_D} = \frac{\overline{A} \cdot C_d \cdot \rho_{air} \cdot \overline{V}^2}{2}, \tag{36}$$

where C_d is the drag coefficient, which is usually determined experimentally and collects the complex dependency, A is the area exposed to the wind, ρ_{air} is the density of the air, and V is the relative velocity of the flying object with respect to the wind. Since the air flow is planar, it can be assumed that the drag force is fully exerted in the centre of the gravity of the vehicle.

In our work, this equation is divided by the mass. This way the expression is transformed to an acceleration ((40) to (43)) that can be easily introduced in the equations of the translational dynamics ((37) to (39)), where wind disturbances have been represented as $dist_wX$, $dist_wY$, and $dist_wZ$.

As the wind speed is a vectorial magnitude, in our experiment, we suppose the wind components X and Y are

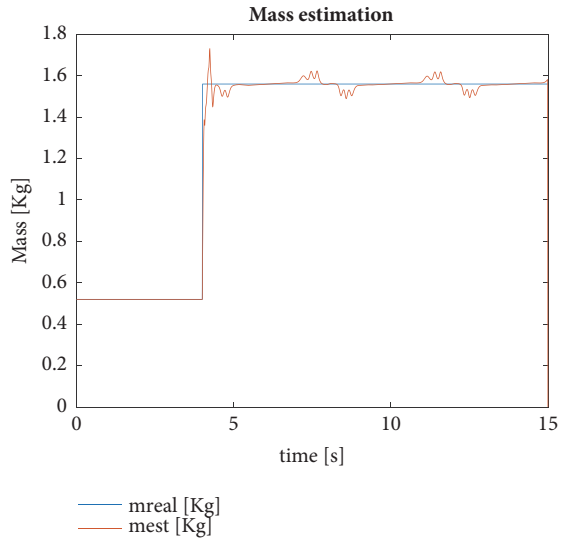

Figure 14: Mass obtained by the neural estimator.

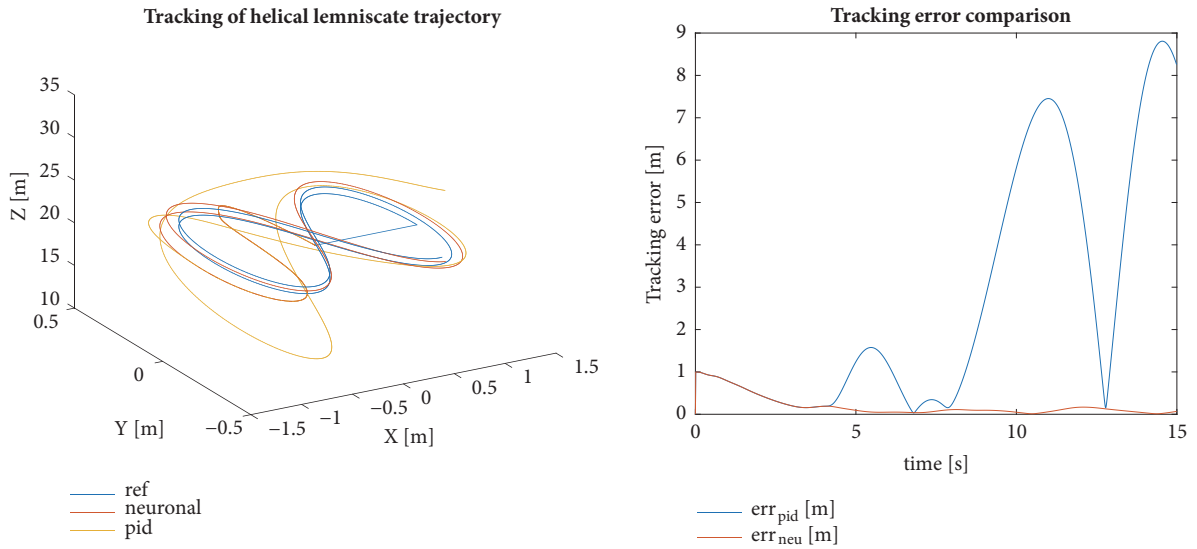

Figure 15: UAV tracking of a helical lemniscate trajectory with sinusoidal mass variation (left) and its tracking error (right).

twice the Z one. Due to this assumption, there appears a factor of 2 multiplying the wind speed in (41) and (42):

$$\ddot{X} = -dist_w X - (\sin\theta\cos\phi)\left(\frac{b}{m}\right)u_1 \tag{37}$$

$$\ddot{Y} = -dist_w Y + (\sin\phi)\left(\frac{b}{m}\right)u_1 \tag{38}$$

$$\ddot{Z} = -dist_w Z - g + (\cos\theta\cos\phi)\left(\frac{b}{m}\right)u_1. \tag{39}$$

In the free atmosphere, the wind is a balance between the Coriolis, centrifugal, and pressure forces acting on the air mass. But in the boundary layer, the mean wind velocity is also a function of the height [37], and it can be expressed as a logarithmic function. This fact is shown in the following:

$$v_w(Z) = v_{w(Z=20)} \cdot \frac{\log(Z/C)}{\log(20/C)} \tag{40}$$

$$dist_w X = \operatorname{sgn}(v_w(Z)) \cdot \rho_{air} \cdot A_x \cdot Cd \cdot \frac{\left(\dot{X} - 2 * v_w(Z)\right)^2}{(2m)} \tag{41}$$

$$dist_w Y = \operatorname{sgn}(v_w(Z)) \cdot \rho_{air} \cdot A_y \cdot Cd \cdot \frac{\left(\dot{Y} - 2 * v_w(Z)\right)^2}{(2m)} \tag{42}$$

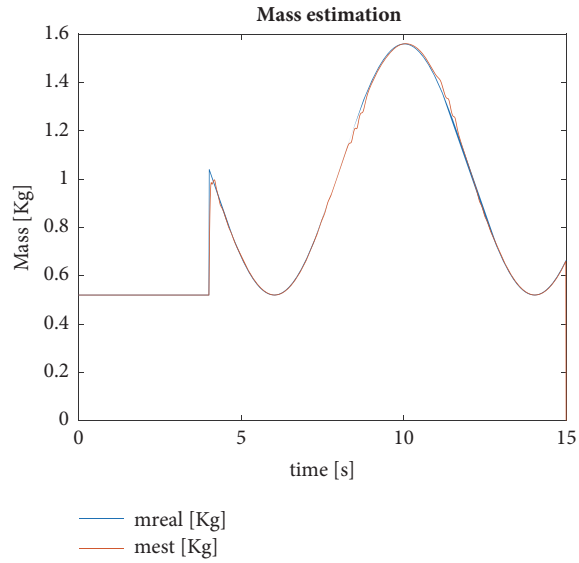

Figure 16: Mass obtained by the neural estimator with a sinusoidal mass change profile.

Table 3: Comparison of the MSE of the tracking error for different trajectories with mass variation.

Trajectory	MSE_X		MSE_Y		MSE_Z		MSE_T	
	Neuro	PID	Neuro	PID	Neuro	PID	Neuro	PID
Linear	0.0707	0.0707	0.0708	0.0708	**1.6099**	21.9021	**0.6635**	3.8267
Circular	**0.2766**	0.2775	**0.0720**	0.0729	**1.6101**	21.9026	**0.7076**	3.8657
Helical	**0.0934**	0.1053	**0.0018**	0.0148	**0.0029**	18.2737	**0.1899**	3.2131
Cyclic helical	**0.1067**	0.2145	**0.0126**	0.1212	**0.0046**	17.5445	**0.2715**	3.1897
Lemniscate	**0.0902**	0.0916	**0.003**	0.0020	**0.0010**	18.2297	**0.1589**	3.1981
Helical lemniscate	**0.0923**	0.1060	**0.0018**	0.0166	**0.0034**	18.2335	**0.1956**	3.2087

$$dist_w Z = \operatorname{sgn}\left(v_w(Z)\right) \cdot \rho_{air} \cdot A_z \cdot Cd \cdot \frac{\left(\dot{Z} - v_w(Z)\right)^2}{(2m)}, \tag{43}$$

where $v_{w(Z=20)}$ is the wind speed at 20 m of altitude in m/s, $v_w(Z)$ is the wind speed at Z altitude in m/s, C is a constant related to the flight (in this experiment, the value is set to 1.5) which is dimensionless, ρ_{air} is the air density in Kg·m^3, A_x to A_z are the effective area of the quadrotor exposed to each component of the wind in m^2, Cd is the drag coefficient with respect to the wind which is dimensionless, $\dot{X}$, $\dot{Y}$, and $\dot{Z}$ are the velocities in the x-axis, y-axis, and z-axis in m/s, and sgn denotes the sign function.

The mean wind velocity can be considered constant during the experiments, but not its instant value. The most common assumption is to consider the wind turbulence as a stationary Gaussian random process [37]. Considering this fact, in our experiment, the wind speed is simulated by a step with Gaussian noise at t = 4 s. The SNR between the average wind and the noise is 10 dB. The average wind speed is 12 m/s in the z-axis and 24 m/s in the x-axis and y-axis. These values match numbers 6 and 9 in Beaufort's scale (strong breeze and strong gale) [38].

4.3.2. Simulation Results with Wind Variation. Figure 17, left, shows the tracking of the trajectory, the reference in blue, the trajectory with adaptive neural estimators in red, and without estimators in yellow. At the right side of Figure 17, the tracking error is shown with estimators (red line) and without estimators (blue line). In both figures, it is possible to see how the performance of the control strategy with the neural estimators is much better. Like in the case of the mass variation, the tracking error is the same until t = 4 s because before there is no wind disturbance, but from that moment on, the tracking error of the controller without the estimators increases significantly.

If the tracking according to each coordinate, X, Y, and Z, is represented (Figure 18), the contribution to the tracking error seems to be more balanced (reference, magenta; trajectory with adaptive neural estimators, red; trajectory without neural estimators, blue). For every coordinate, it is possible to see how the trajectory obtained by the controller with the neural estimators better fits the reference. Figure 18 also shows how the disturbance produces an important deviation around t=7s due to the big peak of disturbance in the y-axis which cannot be compensated without the adaptive neural estimator and it becomes a stationary error. This deviation can be also easily observed in Figure 17 in the PID line.

Table 4: Comparison of the MAX of the tracking error for different trajectories with mass variation.

Trajectory	MAX_X		MAX_Y		MAX_Z		MAX_T	
	Neuro	PID	Neuro	PID	Neuro	PID	Neuro	PID
Linear	1.0270	1.0270	1.0270	1.0270	**5.0010**	9.7835	**5.1986**	9.7835
Circular	2	2	1.0562	1.0562	**5.0010**	9.7836	**5.4796**	9.7837
Helical	1	1	**0.1207**	0.2074	**0.1582**	9.2655	**1.0033**	9.2675
Cyclic helical	1	1	**0.2060**	0.6163	**0.1591**	9.1763	**1.0006**	9.1937
Lemniscate	1	1	**0.0614**	0.0962	**0.0913**	9.2541	**1**	9.2544
Helical lemniscate	1	1	**0.1128**	0.2703	**0.1322**	9.2549	**1**	9.2553

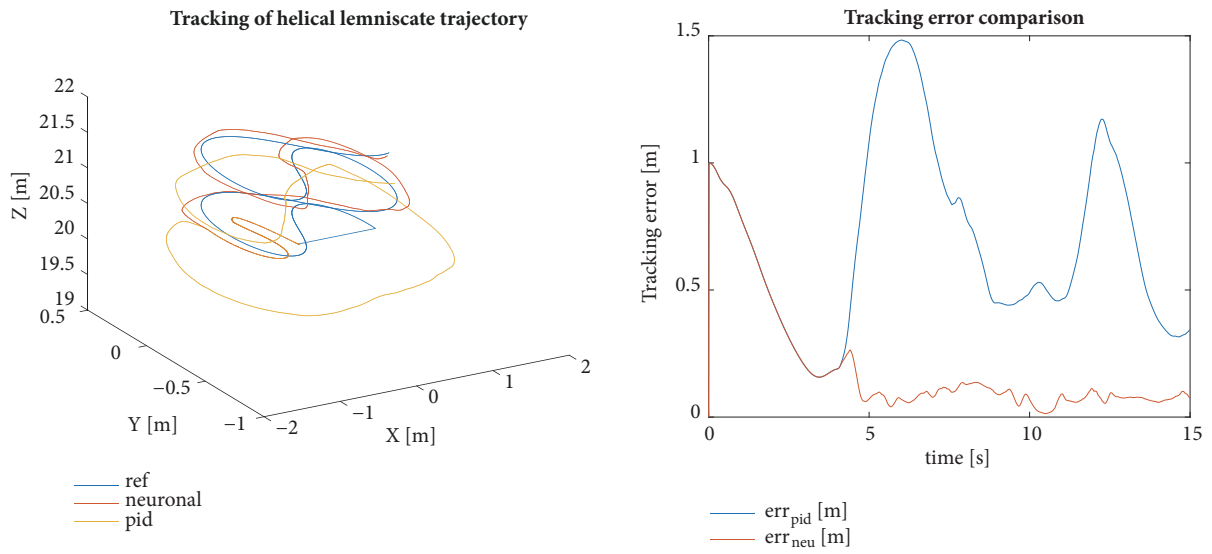

Figure 17: UAV tracking of a helical lemniscate trajectory with wind disturbance (left) and its tracking error (right).

Another interesting result which can be drawn from the previous figures is that we are not using a wind disturbance estimator for the Z coordinate (see Figure 4) but the tracking is still good enough. The reason is that adaptive neuromass estimator senses the wind disturbances in the z-axis as a virtual mass variation and it can compensate it. This is shown in Figure 19, where the estimate of the wind disturbance in the x-, y-, and z-axis is interpreted by the mass estimator as a mass variation. Indeed, the neural estimator (red line) fits reasonably well the real disturbances (blue line).

The controller has been extensively tested for different trajectories and the MSE numerical results are summarized in Table 5 for the different trajectories and for each coordinate, with the neural estimator (neuro) and without it (PID). The last column shows the absolute error. The best result for each component and trajectory is boldfaced.

Table 6 compares the values of the maximum error in this experiment for the same cases with before.

According to Tables 5 and 6, the controller with adaptive neural estimators provides less or equal tracking error for every tested trajectory. The minimal tracking error is achieved for the lemniscate trajectory in the z-axis, with a very small error value of 0.0005, 600 times less than the error value without the neural estimator. The maximum tracking error is obtained with the circular trajectory due to the high error of the initial conditions.

The controller with the adaptive neural estimators works also well when the wind follows other different profiles, such as a sinusoidal function. In the next experiment, the wind average speed at Z = 20 m has been set to

$$v_{w(Z=20)} = 12 + 6 * \sin\left(\frac{2\pi}{4}t\right). \quad (44)$$

Figure 20 shows the results. It can be seen how the performance of the controller without estimators is much worse than with the step wind profile. Nevertheless, the performance of the neurocontroller is like the previous ones. It is also possible to see the sinusoidal shape of the tracking error in Figure 20, right.

5. Conclusions

Intelligent control strategies are especially useful when the parameters change while the system is performing some tasks, and when the external disturbance are relevant, due to

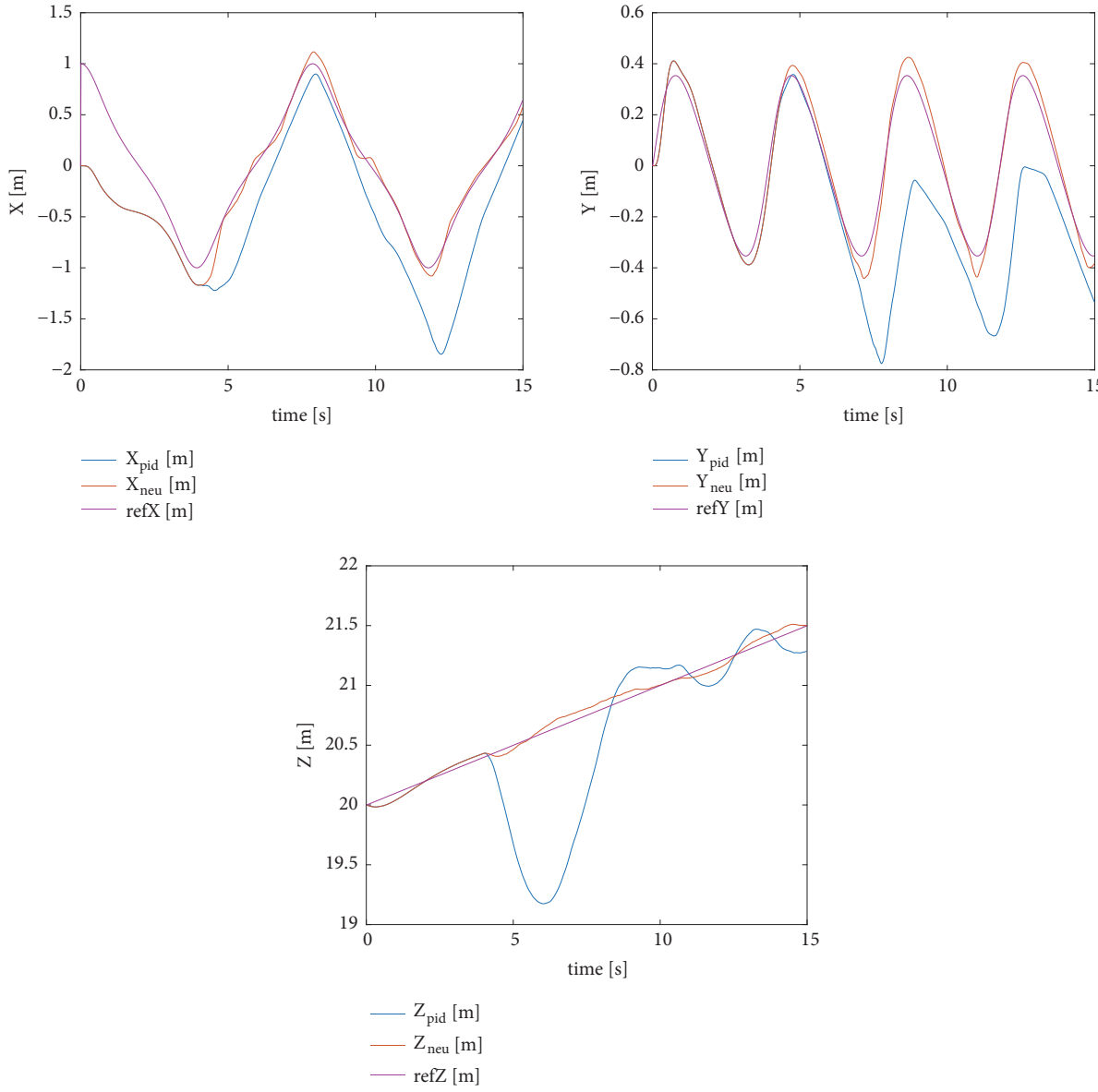

Figure 18: UAV tracking of X, Y, and Z coordinates, respectively, with wind disturbances.

Table 5: Comparison of the MSE of the tracking error for different trajectories with wind disturbances.

Trajectory	MSE_X		MSE_Y		MSE_Z		MSE_T	
	Neuro	PID	Neuro	PID	Neuro	PID	Neuro	PID
Linear	0.0707	0.0707	0.0708	0.0708	**1.6163**	2.1214	**0.6730**	1.0130
Circular	**0.2805**	0.8612	**0.0739**	0.2596	**1.6186**	2.2711	**0.7591**	1.6014
Helical	**0.0951**	0.4850	**0.0042**	0.2009	**0.0048**	0.2847	**0.2181**	0.8666
Cyclic helical	**0.1053**	0.3435	**0.0132**	0.2000	**0.0069**	0.2596	**0.2746**	0.8084
Lemniscate	**0.0911**	0.1928	**0.0011**	0.0335	**0.0005**	0.3036	**0.1748**	0.6008
Helical lemniscate	**0.0940**	0.2614	**0.0032**	0.0695	**0.0010**	0.2460	**0.2015**	0.6903

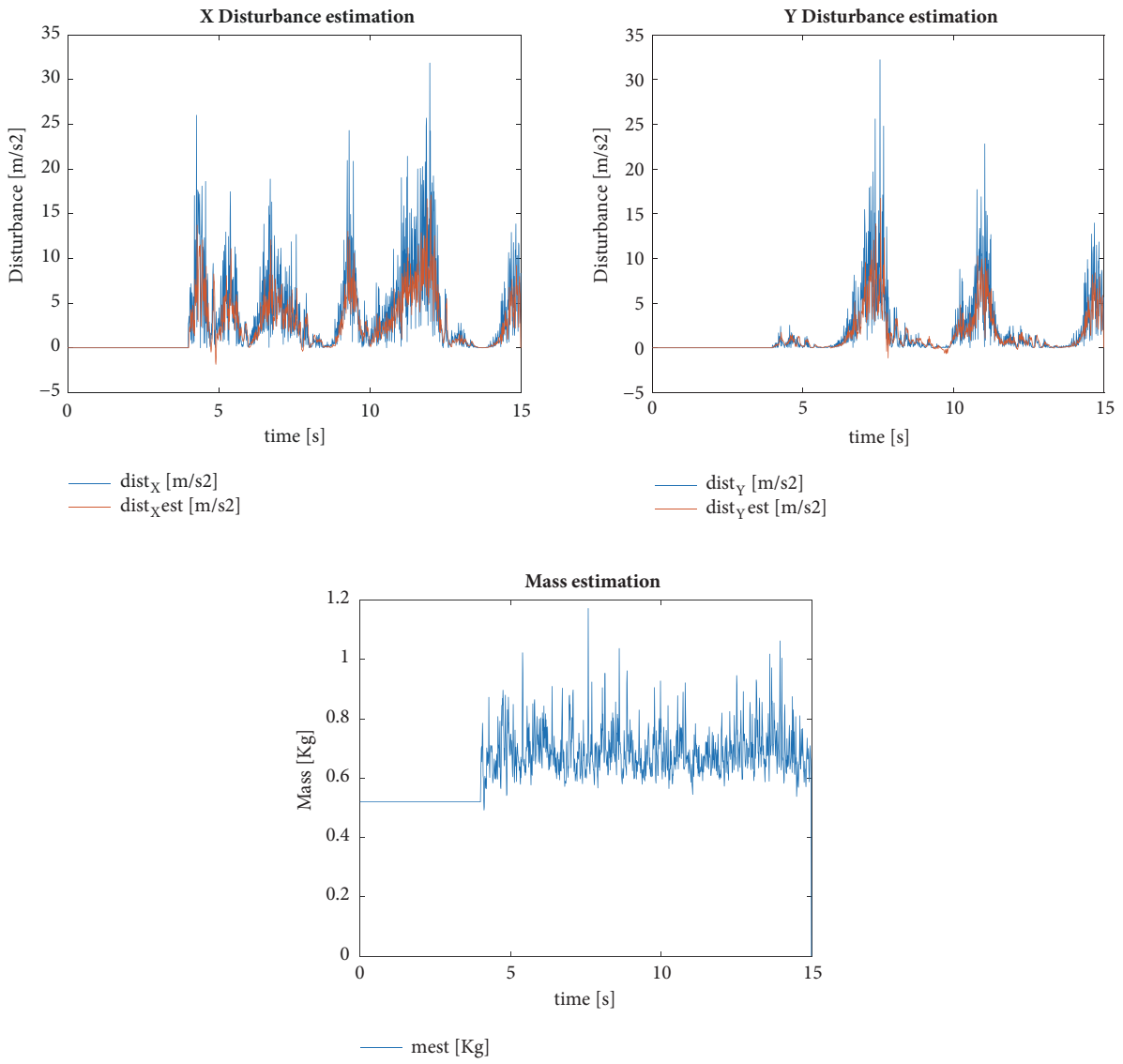

Figure 19: Wind disturbance estimation in the x- and y-axis, top, and mass estimation (bottom).

Table 6: Comparison of the MAX of the tracking error for different trajectories with wind disturbances.

Trajectory	MAX_X		MAX_Y		MAX_Z		MAX_T	
	Neuro	PID	Neuro	PID	Neuro	PID	Neuro	PID
Linear	1.0270	1.0270	1.0270	1.0270	5.0010	5.0010	5.1986	5.1986
Circular	2	2	**1.0562**	1.1864	5.0010	5.0010	5.4796	5.4796
Helical	**1**	1.4086	**0.1834**	0.9557	**0.1582**	1.4093	**1.0033**	1.6068
Cyclic helical	**1**	1.0948	**0.2060**	1.0506	**0.1591**	1.5356	**1.0006**	1.6874
Lemniscate	1	1	**0.1035**	0.5375	**0.0710**	1.6160	**1**	1.6166
Helical lemniscate	**1**	1.0582	**0.1228**	0.7613	**0.0771**	1.3190	**1**	1.3371

the ability to learn and adapt to the changing conditions. This is a common situation in many UAV applications.

In this work, a new intelligent control strategy based on neural networks has been proposed. It includes the design of neural networks that estimate the system parameter variations. They allow the UAV to follow different trajectories with small tracking error when disturbances due to mass changes and wind are included.

Simulation results show how the online learning of the neural estimators increases the robustness of the controllers,

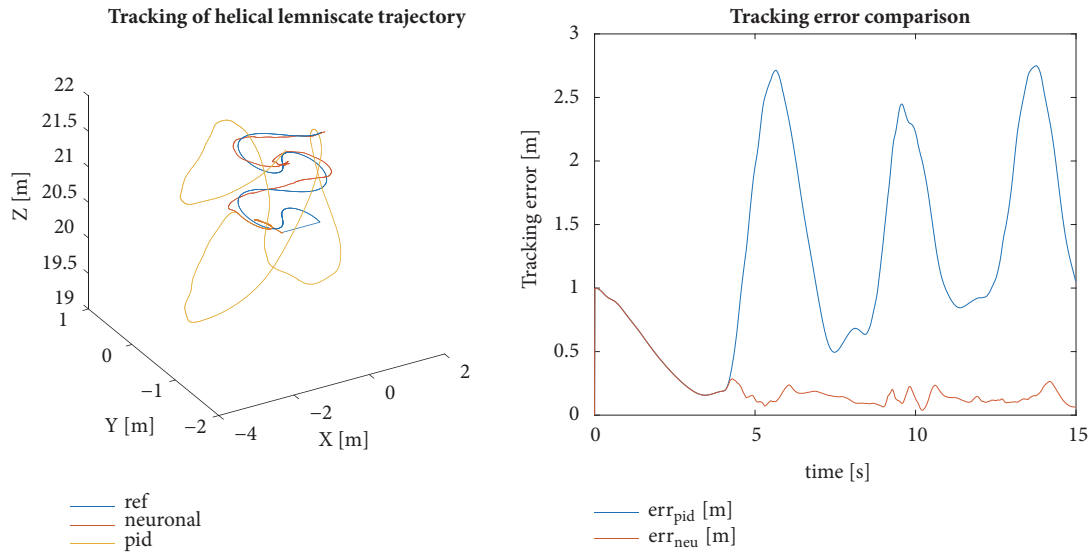

Figure 20: UAV tracking of a helical lemniscate trajectory with sinusoidal wind profile disturbance (left) and its tracking error (right).

reducing the effects of the mass variation and the wind on the UAV.

Among other possible future works, we may highlight the study of the influence of other disturbances such as the ones generated by the engines. In addition, the analysis of the parallelization of this approach for real-time application could be another interesting research line and help to deal with the high computational demand of these systems.

Data Availability

The data used to support the findings of this study are included within the article.

Disclosure

This is an extended version of an earlier version of this work presented in (13th Int. Conf. on Soft Computing Models in Industrial and Environmental Applications, 2018), [32].

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] V. San Juan, M. Santos, J. M. And, and J. M. Andújar, "Intelligent UAV Map Generation and Discrete Path Planning for Search and Rescue Operations," *Complexity*, vol. 2018, Article ID 6879419, 17 pages, 2018.

[2] R. Szabolcsi, "The Quadrotor-Based Night Watchbird UAV System Used In The Force Protection Tasks," *International conference Knowledge-Based Organization*, vol. 21, no. 3, pp. 749–755, 2015.

[3] R. Grassi, P. Rea, E. Ottaviano, and P. Maggiore, "Application of an Inspection Robot Composed by Collaborative Terrestrial and Aerial Modules for an Operation in Agriculture," in *Proceedings of the In International Conference on Robotics in Alpe-Adria Danube Region*, pp. 539–546, Cham, 2017.

[4] A. G. E. Ruiz, H. Alazki, J. E. V. Rubio, and O. G. Salazar, "Embedded super twisting control for the attitude of a Quadrotor," *IEEE Latin America Transactions*, vol. 14, no. 9, pp. 3974–3979, 2016.

[5] M. Santos, V. López, and F. Morata, "Intelligent fuzzy controller of a quadrotor," in *Proceedings of the IEEE International Conference on Intelligent Systems and Knowledge Engineering (ISKE '10)*, pp. 141–146, IEEE, Hangzhou, China, November 2010.

[6] Z. Wu, Z. Guan, C. Yang, and J. Li, "Terminal guidance law for UAV based on receding horizon control strategy," *Complexity*, vol. 2017, Article ID 2750172, 19 pages, 2017.

[7] P. Alfredo Toriz, B. Modesto Raygoza, and N. Daniel Martínez, "UAV technology inclusion model for preventing high-risk jobs in construction industries based on the IVAS methodology," *RIAI - Revista Iberoamericana de Automatica e Informatica Industrial*, vol. 14, no. 1, pp. 94–103, 2017.

[8] A. A. Fahmy and A. M. Abdel Ghany, "Adaptive functional-based neuro-fuzzy PID incremental controller structure," *Neural Computing and Applications*, vol. 26, no. 6, pp. 1423–1438, 2015.

[9] P. Garcia-Aunon, M. Santos Peñas, and J. M. García, "A new UAV ship-tracking algorithm," in *Proceedings of the Preprints of the 20th IFAC World Congress*, pp. 13090–13095.

[10] Z. Zhou, H. Wang, and Z. Hu, "Event-Based Time Varying Formation Control for Multiple Quadrotor UAVs with Markovian Switching Topologies," *Complexity*, vol. 2018, Article ID 8124861, 15 pages, 2018.

[11] X. Zhang, X. Li, K. Wang, and Y. Lu, "A survey of modelling and identification of quadrotor robot," *Abstract and Applied Analysis*, vol. 2014, Article ID 320526, 16 pages, 2014.

[12] M. Santos, "Un Enfoque Aplicado del Control Inteligente," *Revista Iberoamericana de Automática e Informática Industrial RIAI*, vol. 8, no. 4, pp. 283–296, 2011.

[13] M. Santos, R. López, and J. M. De La Cruz, "A neuro-fuzzy approach to fast ferry vertical motion modelling," *Engineering*

Applications of Artificial Intelligence, vol. 19, no. 3, pp. 313–321, 2006.

[14] B.-C. Min, J.-H. Hong, and E. T. Matson, "Adaptive robust control (ARC) for an altitude control of a quadrotor type UAV carrying an unknown payloads," in *Proceedings of the 11th International conference on control, automation and systems Korea*, pp. 26–29, 2011.

[15] C. Wang, M. Nahon, and M. Trentini, "Controller development and validation for a small quadrotor with compensation for model variation," in *Proceedings of the 2014 International Conference on Unmanned Aircraft Systems, ICUAS 2014*, pp. 902–909, May 2014.

[16] C. Wang, B. Song, P. Huang, and C. Tang, "Trajectory Tracking Control for Quadrotor Robot Subject to Payload Variation and Wind Gust Disturbance," *Journal of Intelligent & Robotic Systems*, vol. 83, no. 2, pp. 315–333, 2016.

[17] K. Sreenath, N. Michael, and V. Kumar, "Trajectory generation and control of a quadrotor with a cable-suspended load-a differentially-flat hybrid system," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '13)*, pp. 4888–4895, May 2013.

[18] K. Sreenath and V. Kumar, "Dynamics, control and planning for cooperative manipulation of payloads suspended by cables from multiple quadrotor robots," in *Robotics, Science and Systems (RSS)*, Berlin, Germany, 2013.

[19] S. Lee, D. K. Giri, and H. Son, "Modeling and control of quadrotor UAV subject to variations in center of gravity and mass," in *Proceedings of the 14th International Conference on Ubiquitous Robots and Ambient Intelligence, URAI 2017*, pp. 85–90, Republic of Korea, July 2017.

[20] I. Palunko and R. Fierro, "Adaptive control of a quadrotor with dynamic changes in the center of gravity," in *Proceedings of the In Proceedings 18th IFAC World Congress*, vol. 18, pp. 2626–2631, 2011.

[21] I. Palunko, R. Fierro, and P. Cruz, "Trajectory generation for swing-free maneuvers of a quadrotor with suspended payload: a dynamic programming approach," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '12)*, pp. 2691–2697, 2012.

[22] P. Kotaru, G. Wu, and K. Sreenath, "Dynamics and control of a quadrotor with a payload suspended through an elastic cable," in *Proceedings of the 2017 American Control Conference, ACC 2017*, pp. 3906–3913, USA, May 2017.

[23] B. Yuksel, C. Secchi, H. H. Bülthoff, and A. Franchi, "A nonlinear force observer for quadrotors and application to physical interactive tasks," in *Proceedings of the 2014 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM 2014*, pp. 433–440, July 2014.

[24] J. Escareño, S. Salazar, H. Romero, and R. Lozano, "Trajectory control of a quadrotor subject to 2D wind disturbances: Robust-adaptive approach," *Journal of Intelligent & Robotic Systems*, vol. 70, no. 1-4, pp. 51–63, 2013.

[25] K. Alexis, G. Nikolakopoulos, and A. Tzes, "Switching model predictive attitude control for a quadrotor helicopter subject to atmospheric disturbances," *Control Engineering Practice*, vol. 19, no. 10, pp. 1195–1207, 2011.

[26] D. Cabecinhas, R. Cunha, and C. Silvestre, "A globally stabilizing path following controller for rotorcraft with wind disturbance rejection," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 2, pp. 708–714, 2015.

[27] L. Besnard, Y. B. Shtessel, and B. Landrum, "Quadrotor vehicle control via sliding mode controller driven by sliding mode disturbance observer," *Journal of The Franklin Institute*, vol. 349, no. 2, pp. 658–684, 2012.

[28] J. E. Sierra and M. Santos, "Modelling engineering systems using analytical and neural techniques: Hybridization," *Neurocomputing*, vol. 271, pp. 70–83, 2018.

[29] S. Bansal, A. K. Akametalu, F. J. Jiang, F. Laine, and C. J. Tomlin, "Learning quadrotor dynamics using neural network for flight control," in *Proceedings of the 55th IEEE Conference on Decision and Control, CDC 2016*, pp. 4653–4660, USA, December 2016.

[30] H. Boudjedir, O. Bouhali, and N. Rizoug, "Adaptive neural network control based on neural observer for quadrotor unmanned aerial vehicle," *Advanced Robotics*, vol. 28, no. 17, pp. 1151–1164, 2014.

[31] N. A. Bakshi and R. Ramachandran, "Indirect model reference adaptive control of quadrotor UAVs using neural networks," in *Proceedings of the 10th International Conference on Intelligent Systems and Control, ISCO 2016*, India, January 2016.

[32] J. E. Sierra and M. Santos, "Disturbances Based Adaptive Neuro-Control for UAVs: A First Approach," in *Proceedings of the In The 13th Int. Conf. on Soft Computing Models in Industrial and Environmental Applications, AISC*, pp. 293–302, Springer, 2018.

[33] A. K. Shastry, A. Pattanaik, and M. Kothari, "Neuro-adaptive augmented dynamic inversion controller for quadrotors," *IFAC-PapersOnLine*, vol. 49, no. 1, pp. 302–307, 2016.

[34] H. Yañez-Badillo, R. Tapia-Olvera, O. Aguilar-Mejía, and F. Beltran-Carbajal, "On Line Adaptive Neurocontroller for Regulating Angular Position and Trajectory of Quadrotor System," *RIAI - Revista Iberoamericana de Automatica e Informatica Industrial*, vol. 14, no. 2, pp. 141–151, 2017.

[35] C. Nicol, C. J. B. Macnab, and A. Ramirez-Serrano, "Robust neural network control of a quadrotor helicopter," in *Proceedings of the Canadian Conferenceof Electrical and Computer Engineering, CCECE*, pp. 001233–001238, IEEE, 2008.

[36] NASA Glenn Research Center. https://www.grc.nasa.gov/WWW/K-12/airplane/ldrat.html.

[37] J. Etele, *Overview of wind gust modelling with application to autonomous low-level UAV control*, Mechanical and Aerospace Engineering Department, Carelton University, Ottawa, Canada, 2006.

[38] Royal Meteorological Society, https://www.rmets.org/weather-and-climate/observing/beaufort-scale.

3 Estudio comparativo de modelos de un vehículo aéreo obtenidos mediante técnicas analíticas y basadas en redes neuronales

BIBLIOGRAPHIC DESCRIPTION

Title

Estudio comparativo de modelos de un vehículo aéreo obtenidos mediante técnicas analíticas y basadas en redes neuronales.

Citation

Sierra, J.E., Santos, M. (2013) Estudio comparativo de modelos de un vehículo aéreo obtenidos mediante técnicas analíticas y basadas en redes neuronales. In Actas Multiconferencia CAEPIA'13 (XV Conferencia de la Asociación Española para la Inteligencia Artificial), pages 1270-1279, 2013. ISBN: 978-84-695-8348-7.

Abstract

A partir de datos de entrada/salida reales de un vehículo aéreo no tripulado (UAV) se han generado diferentes modelos mediante distintas técnicas de identificación de sistemas, incluyendo las que proporciona el soft computing. El dispositivo que se ha utilizado es un cuatrirrotor, por lo tanto un sistema complejo, no lineal, multivariable y con acoplamientos. Se han analizado y comparado los resultados de la aplicación de técnicas paramétricas y de estrategias inteligentes para la obtención de modelos orientados al control.

References

Bendat et al(1980); Isaksson,A.J. (1993); Kim et al (2003); Norgard, P.M. et al (2000); Park, J. et al (1991); Rutkowski, L. (2004); San Martin, R. et al (2006); Santos et al (2006)

Estudio comparativo de modelos de un vehículo aéreo obtenidos mediante técnicas analíticas y basadas en redes neuronales

J. Enrique Sierra[1], Matilde Santos[2]

[1] Universidad de Burgos, Burgos, España
jesiegar@gmail.com
[2] Universidad Complutense de Madrid, Madrid, España
msantos@ucm.es

Resumen. A partir de datos de entrada/salida reales de un vehículo aéreo no tripulado (UAV) se han generado diferentes modelos mediante distintas técnicas de identificación de sistemas, incluyendo las que proporciona el soft computing. El dispositivo que se ha utilizado es un cuatrirotor, por lo tanto un sistema complejo, no lineal, multivariable y con acoplamientos. Se han analizado y comparado los resultados de la aplicación de técnicas paramétricas y de estrategias inteligentes para la obtención de modelos orientados al control.

Palabras clave: Soft computing, Redes neuronales, técnicas paramétricas, modelización, identificación, vehículos aéreos no tripulados (UAV).

1 Introducción

La identificación de sistemas es uno de los pasos más importantes para obtener modelos útiles y realizables de sistemas complejos. Fundamentalmente existen dos vías para alcanzar este objetivo: la primera de ellas consiste en aplicar las ecuaciones físicas conocidas que gobiernan el comportamiento dinámico y emplear este conocimiento para construir las ecuaciones diferenciales que lo representan. Esta metodología resulta muy útil cuando las variables son perfectamente conocidas y las relaciones entre ellas no son demasiado complejas. La otra vía consiste en la medida y el tratamiento de las señales en la entrada y la salida del sistema, y la aplicación de diferentes mecanismos de mapeado para describir las relaciones internas subyacentes entre las entradas y las salidas. En esta categoría se encuentran, entre otras, algunas de las técnicas proporcionadas por el Soft Computing, como las redes neuronales y los algoritmos genéticos [5].

La modelización de un vehículo aéreo no tripulado, UAV (Unmanned Aerial Vehicle), no es una tarea sencilla. La complejidad proviene de diferentes fuentes: la aleatoriedad de los flujos de aire y de las fuerzas exógenas, la inhomogeneidad de la estructura mecánica interna, y la fuerte no linealidad de la dinámica del sistema. Todos estos factores convierten a las técnicas basadas en el control inteligente en un camino idóneo para modelar estos sistemas. Otros trabajos han aplicado estas estra-

tegias para modelar estos sistemas, tales como [9] y [10]. También se pueden encontrar ejemplos de la aplicación de las técnicas inteligentes para modelar otros sistemas complejos no lineales como, por ejemplo, vehículos marinos [11]. Mediante este trabajo se realizan dos aportaciones novedosas al estudio de los UAVs, la aplicación de las redes neuronales de regresión generalizada como técnica de modelización, y la utilización del error cuadrático medio de la respuesta al escalón, estimada mediante análisis de correlación, como herramienta de contraste de la eficacia del modelo.

Empleando datos reales de entrada/salida de un UAV se han obtenido diferentes modelos basados en redes neuronales. También se han utilizado métodos tradicionales de identificación y los resultados se han comparado.

El vehículo a modelar es el ARDRONE 2.0 comercializado por Parrot (figura 1). Este aparato se fabrica en fibra de carbono y plástico PA66. Está equipado con cuatro motores tipo brushless de alta eficiencia, alimentados por una batería de litio de 11.1 V y 1000 mAh que le confiere una autonomía de vuelo de aproximadamente 12 minutos. La masa es de 380 g sin carcasa y 420 g con carcasa. Dispone de un acelerómetro de 3 ejes, un giróscopo de 2 ejes (balanceo y cabeceo), y un giróscopo de precisión para el ángulo de guiñada [7].

Fig. 1. Vehículo aéreo no tripulado seleccionado (Fuente: www.parrot.com)

A continuación se detalla la organización del documento. En la sección 2 se desarrolla un análisis preliminar describiendo las señales examinadas y el análisis de correlación. La sección 3 se destina al estudio de los modelos obtenidos mediante técnicas tradicionales. En la sección 4 se aplican las redes neuronales basadas en los datos experimentales. Los resultados son discutidos y comparados en 5. El documento finaliza con las conclusiones.

2 Análisis preliminar

2.1 Señales empleadas

Para realizar el modelado se parte de un conjunto de datos suministrados por el Grupo de Ingeniería de Control de CEA para el concurso de Ingeniería de Control 2013 [2]. El conjunto recoge los siguientes datos: instante de muestreo, pitch, roll, x, vx, y e vy, con un tiempo de muestreo de 60 ms, tomados durante 9 minutos. Nos

centraremos en el modelado de las velocidades vx y vy, puesto que la posición (x,y) viene determinada por la integración de vx y vy. Las señales de entrada y salida se representan en la figura 2. En azul vx y pitch y en verde vy y pitch. Puede adelantarse ya el importante desacoplo entre las variables que se corroborará mediante la correlación.

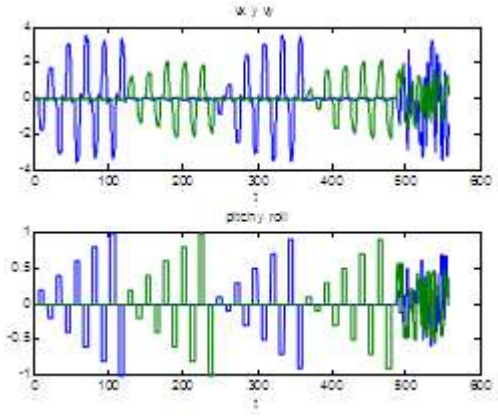

Fig. 2. Señales de entrada y salida

2.2 Análisis de correlación

A partir de la estima de la covarianza entre la entrada y la salida de un sistema y la estima de la varianza de la entrada, puede calcularse la estima de la respuesta al impulso, con las expresiones siguientes:

$$\hat{R}_{yu}(\tau) = \frac{1}{N}\sum y(t)u(t-\tau) \tag{1}$$

$$\hat{\lambda} = \frac{1}{N}\sum u(t)^2 \tag{2}$$

$$g = \frac{\hat{R}_{yu}(\tau)}{\hat{\lambda}} \tag{3}$$

Siendo y la salida del sistema, u la entrada, N el número de datos, R la covarianza, λ la varianza y g la respuesta estimada al impulso. Integrando la respuesta al impulso puede obtenerse la respuesta al escalón, la cual puede ser muy útil para ayudarnos a estimar el grado de acoplamiento entre las variables del sistema. Esta metodología de estudio suele denominarse análisis de correlación [1]. Empleando el método descrito se obtiene la figura 4.

En la respuesta al escalón de la matriz de funciones de transferencia de la figura 4, se aprecia como la ganancia en estado estacionario de los elementos de la diagonal principal, correspondientes al enlace (fila 1, columna 1) y (fila 2, columna 2), son bastante mayores que el resto. Este hecho nos indica que las dinámicas del sistema están parcialmente desacopladas, pudiendo convertir el sistema MIMO (multi input multi output) en dos sistemas SISO (single input single output). Aprovechando este

hecho, los esfuerzos en modelaje en este trabajo se centrarán en la influencia del pitch sobre la velocidad en el eje x.

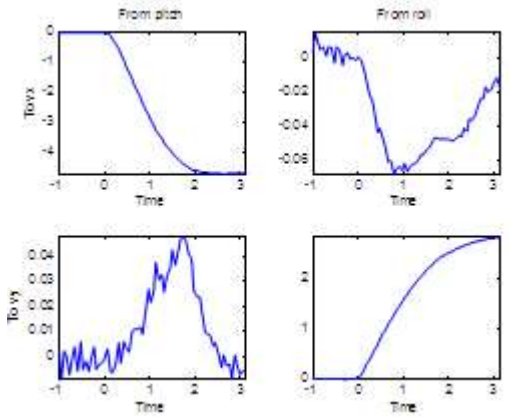

Fig. 3. Respuesta a un escalón estimada del sistema MIMO

3 Modelos paramétricos

3.1 Modelos basados en el espacio de estados

Mediante el modelo en el espacio de estados se describe el comportamiento del sistema a partir de las siguientes expresiones genéricas, [4]:

$$\dot{x}(t) = Ax(t) + Bu(t) + Ke(t) \tag{4}$$

$$y(t) = Cx(t) + Du(t) + e(t) \tag{5}$$

Donde x(t) representa el estado interno del sistema, u(t) las entradas, y(t) las salidas, e(t) una señal de error, y A, B, C, D son polinomios. En las figuras 4 izquierda y derecha se representan respectivamente, la comparación de la respuesta del modelo con los datos reales y la comparación de la estima de la respuesta al escalón respectivamente para orden 4.

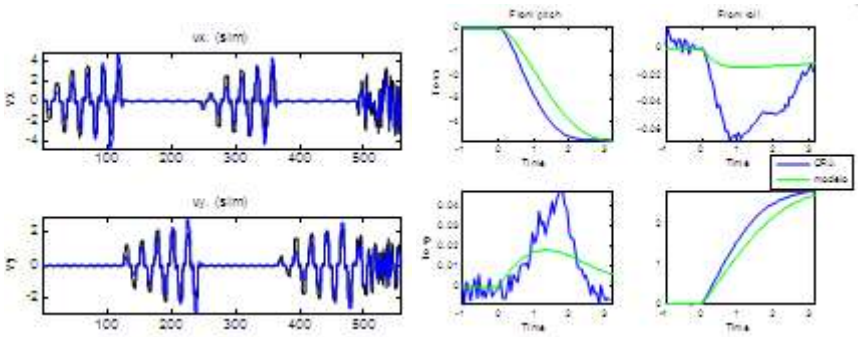

Fig. 4. Simulación del modelo de sistema de estados con orden 4 (izquierda). Comparación respuesta a un escalón del modelo de sistema de estados con orden 4(derecha)

3.2 Modelos ARX

Los modelos ARX (autoregressive exogenous model), descritos en [3], describen el sistema con una ecuación de la siguiente forma:

$$Ay(t) = Bu(t) + e \tag{6}$$

Donde y es la salida, u la entrada y e el error de la estima y A y B son polinomios. Si se restringe el grado máximo de los polinomios a 10, el mejor resultado se consigue con la configuración ARX (10, 10, 1) donde 10 es el grado de A y de B y 1 son las muestras de retardo. En la figura 5 izquierda se representa la comparación de la simulación temporal de los diferentes modelos estudiados con los datos reales y en 5 derecha la comparativa entre la respuesta al escalón estimada mediante análisis de correlación y la proporcionada por los diferentes modelos ARX estudiados

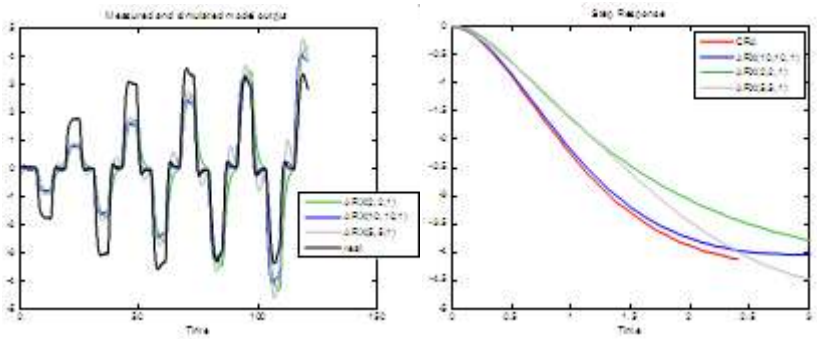

Fig. 5. Comparativa entre los datos reales y simulaciones de los modelos ARX. (izquierda). Comparativa de la respuesta al escalón estimada mediante CRA y los modelos ARX. (derecha)

3.3 Modelos OE

Los modelos OE describen el sistema mediante la expresión:

$$y(t) = \frac{B}{F}u(t)+e \tag{7}$$

Donde B y F son polinomios, e el error de la estima, u la entrada e y la salida del sistema. Si se restringe el grado máximo de los polinomios a 10 el mejor modelo se consigue con el OE (10, 10, 1). Siendo 3 el grado del polinomio B y F y 1 el retardo del sistema

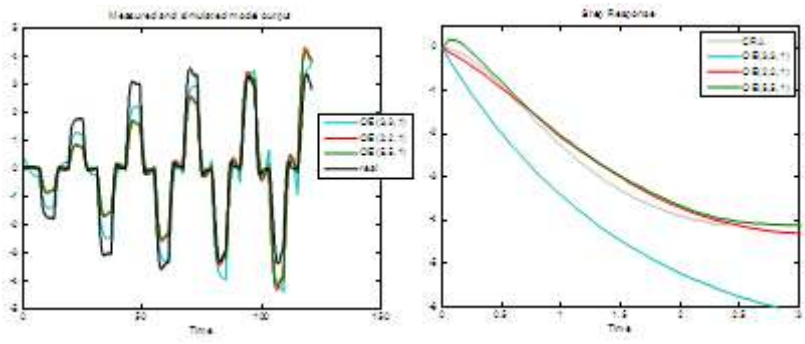

Fig. 6. Comparativa entre los datos reales y simulaciones de los modelos OE. (izquierda). Comparativa de la respuesta al escalón estimada mediante CRA y los modelos OE (derecha)

En la figura 6 izquierda se presenta la comparativa de los datos de salida reales de vx, y el resultado de la simulación temporal para diferentes modelos OE. Mientras que en la figura 6 derecha se presenta la comparación de la estima de la respuesta al escalón obtenida mediante análisis de correlación y la proporcionada por los modelos OE estudiados.

4 Modelos basados en redes neuronales

4.1 Redes neuronales de base radial

Las redes de base radial están formadas por tres capas, una capa de entrada donde se reciben las señales del exterior, una capa oculta conectada con la capa de entrada donde cada elemento tiene asociada una función de base radial y una capa de salida donde se combinan linealmente los resultados de las funciones de activación de la capa oculta [6].

La salida de cada elemento de la capa oculta Zi(n) se calcula como la distancia entre el patrón de entrada al centro del cluster ponderada por di y aplicando este valor a una función de base radial, según la siguiente expresión:

$$z_i(n) = \Phi\left(\frac{\sqrt{\sum(x_j(n)-c_{ij})^2}}{d_i}\right) \tag{8}$$

La combinación lineal de los valores de salida de la capa oculta para generar los valores de salida de la red se realiza mediante la siguiente expresión:

$$y_k(n) = \sum w_{ik} z_i(n) + \mu_k \tag{9}$$

Siendo wik los pesos de la capa de salida y μk los umbrales de activación. Siendo en nuestro caso, la función de base radial Φ:

$$\Phi(n) = e^{-n^2} \tag{10}$$

Para realizar el estudio, el conjunto de datos de entrada de la red se ha formado con la señal de entrada en el instante actual pitch(t) y valores en algunos instantes de muestreo anteriores pitch(t-Ki) donde i = 1…100/K para diferentes valores de K. Y el conjunto de datos objetivo se ha generado con los valores de la señal de salida en el instante actual vx(t).

Se han empleado 5000 muestras de las señales, el 60% se ha empleado para entrenar las redes y el resto para simular.

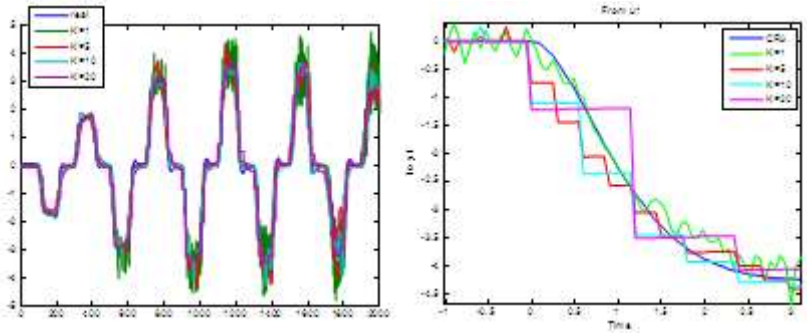

Fig. 6. Comparativa entre los datos reales y simulaciones (izquierda) y de la respuesta al escalón (derecha), para diferentes valores de K de la red de base radial

Como puede apreciarse en las figura 7, donde se comparan los resultados de simulaciones y la respuesta al escalón conforme se aumenta K, la estima de la respuesta al escalón es mejor pero como contrapartida se aumenta el ruido en la simulación temporal.

4.2 Redes neuronales de regresión generalizada

Las redes de regresión generalizada suponen una modificación sobre las redes de base radial. En la capa de salida se realiza una normalización de los valores obtenidos en la capa oculta y una ponderación para obtener los valores de salida de la red. Estas redes resultan especialmente útiles en la aproximación de funciones [8]. Esta característica de especialidad puede observarse en la figura 8 donde la aproximación a los valores reales para diferentes K es prácticamente perfecta. En el caso de la respuesta al impulso de la figura 8, se puede observar como la aproximación es muy buena para todos los valores de K, la diferencia consiste en el tamaño de los escalones empleados, el cual crece conforme K aumenta

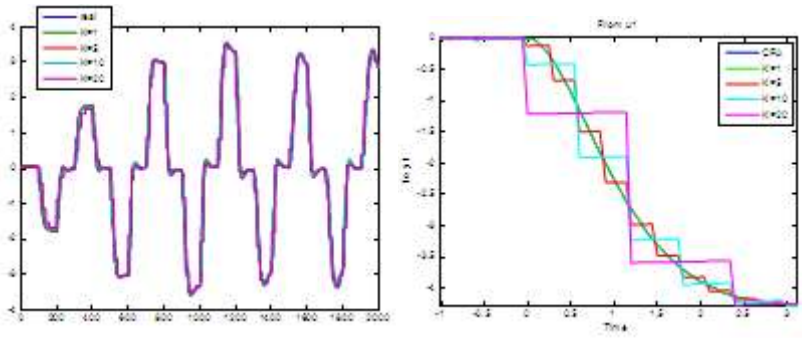

Fig. 7. Comparativa entre los datos reales y simulaciones (izquierda) y de la respuesta al escalón (derecha), para diferentes valores de K de la red de regresión generalizada

4.3 Redes neuronales feed-forward backpropagation

Estas redes neuronales son más simples que las anteriores. Están formadas por varias capas, y cada capa es similar a un perceptrón pero substituyendo la función de activación de tipo escalón por un sigmoide [5]. Para la comparación se ha empleado una red de 5 capas, con un algoritmo de entrenamiento de Levenberg-Marquardt y como función de aprendizaje el gradiente descendiente. En la figura 9 izquierda se presenta la comparativa de las simulaciones obtenidas para diferentes valores de K. Por otra parte, en la figura 9 derecha se presenta la comparativa de la respuesta el escalón obtenida con los modelos ensayados para diferentes valores de K

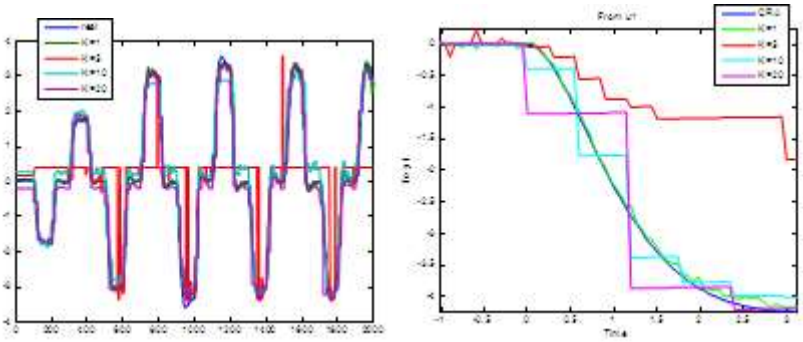

Fig. 8. Comparativa entre los datos reales y simulaciones (izquierda) y de la respuesta al escalón (derecha), para diferentes valores de K de la red feed-forward backpropagation

Se aprecia en las figura 9 y 9 como para K=5 la estima de la respuesta al escalón se aleja bastante de la obtenida mediante el análisis de correlación.

5 Resultados

La tabla 1 muestra los resultados cuantitativos en términos de error cuadrático medio, MSE (Mean Squared Error) para cada técnica estudiada. Tanto los resultados gráficos de las secciones anteriores como los cuantitativos de esta sección, muestran que las redes de regresión generalizada presentan el mejor comportamiento.

	Modelos paramétricos			Modelos basados en redes neuronales		
Technique	Espacio Estados orden 4	ARX 10,10,1	OE 10,10,1	Feedforward Network	Radial Basis Network	Generalized Regression Network
MSE	0.6439	0.5056	0.4675	0.0541	0.1477	2.68E-5
MSE step response	0.1478	0.0122	0.1182	0.0319	0.0602	1.35E-5

Tabla 1. Comparación del MSE para valores en el dominio del tiempo y la respuesta al escalón estimada.

6 Conclusiones

En este trabajo se ha analizado la posibilidad de modelar un vehículo aéreo no tripulado usando diferentes técnicas de identificación de sistemas. Se han comparado técnicas paramétricas y modelos basados en redes neuronales. En el dominio del tiempo, se ha comprobado el error entre las salidas reales y las salidas del modelo, y el error entre la respuesta al escalón estimada obtenida de datos reales y la respuesta al escalón estimada obtenida del modelo.

Los UAVs son sistemas muy complejos de modelar, la complejidad deriva de la aleatoriedad de los flujos de aire y de las fuerzas exógenas y de la dinámica fuertemente no lineal. Estos factores señalan al control inteligente como una estrategia eficiente para modelar estos sistemas. Para a llevar a cabo esta tarea, se han implementado un conjunto de herramientas informáticas que nos permitirán modelar nuevos tipos de UAVs acelerando el proceso de desarrollo, reduciendo por tanto el tiempo y el coste.

Los resultados son altamente satisfactorios: el MSE obtenido para las redes de regresión generalizada indica que el modelo proporciona un comportamiento excelente tanto para la secuencia temporal como para la respuesta al escalón. En última instan-

cia, se ha contrastado la idoneidad de las técnicas basadas en redes neuronales para identificar y modelar estos sistemas.

7 Agradecimientos

A los autores les gustaría agradecer al Grupo de Ingeniería de Control del Comité Español de Automática [2] la aportación del fichero con datos reales empleado para modelar el UAV.

Referencias

1. Bendat, J.S., Piersol, A.G.,: Engineering Applications of Correlation and Spectral Analysis. , Wiley-Interscience, New York (1980)
2. Control Engineering Group of the Spanish Committee of Automatic: www.ceautomatica.es/og/ingenieria-de-control (2013)
3. Isaksson,A.J.: Identification of ARX-models Subject to Missing Data. In: IEEE Transactions on Automatic Control (1993)
4. Kim,C.J., Nelson,C.R.: State-space models with regime switching: classical and Gibbs-sampling approaches with applications. In: MIT Press Books, (2003)
5. Norgard, P.M. et al: Neural Networks for Modelling and Control of Dynamic Systems – A Practitioner's Handbook. Springer,London (2000)
6. Park, J., Sandberg, I.W,: Universal Approximation Using Radial-basis-function Networks. In: Neural Computation (1991)
7. Parrot . www.parrot.com (2013)
8. Rutkowski, L.,: Generalized Regression Neural Networks in Time-varing Environment. In: IEEE Transactions on Neural Networks (2004)
9. San Martin,R. et al: Neural Networks Training Architecture for UAV Modelling. In: World Automation Congress (WAC) , Budapest, Hungary (2006, July 24-26)
10. San Martin, R. et al: Unmanned Aerial Vehicle (UAV) Modelling Based on Supervised Neural Networks. In: Proceedings of the 2006 IEEE International Conference on Robotics and Automation Orlando, Florida (May 2006)
11. Santos, M., López, R., de la Cruz, J.M.:A Neuro-Fuzzy Approach to Fast Ferry Vertical Motion Modelling. In: Engineering Application of Artificial Intelligence, vol. 19, pp.313-321, (2006)

4 Modelado de un vehículo aéreo no tripulado mediante aplicación conjunta de técnicas paramétricas y neuronales

BIBLIOGRAPHIC DESCRIPTION

Title

Modelado de un vehículo aéreo no tripulado mediante aplicación conjunta de técnicas paramétricas y neuronales.

Citation

Sierra, J.E., Santos, M. (2013) Modelado de un vehículo aéreo no tripulado mediante aplicación conjunta de técnicas paramétricas y neuronales. In Actas del Congreso JJAA 2013, XXXIV Jornadas de Automática, pages 189-194, 2013. ISBN: 978-84-616-5063-7.

Abstract

A partir de datos de entrada/salida reales de un vehículo aéreo no tripulado (UAV) se han generado diferentes modelos mediante distintas técnicas de identificación de sistemas, incluyendo las que proporciona el soft computing. El dispositivo que se ha utilizado es un cuatrirrotor, por lo tanto un sistema complejo, no lineal, multivariable y con acoplamientos. Se han analizado y comparado los resultados de la aplicación individual y conjunta de técnicas paramétricas y de estrategias inteligentes para la obtención de modelos.

References

Bendat, J. S. et al (1980); Isaksson, A. J. (1993). Nørgård, P. M. et al (2000). Park, J. et al (1991); San Martin, R. et al (2006); Santos, M. et al (2006)

MODELADO DE UN VEHÍCULO AEREO NO TRIPULADO MEDIANTE APLICACIÓN CONJUNTA DE TÉCNICAS PARAMÉTRICAS Y NEURONALES

J. Enrique Sierra
Universidad de Burgos, jesiegar@gmail.com

Matilde Santos
Universidad Complutense de Madrid, msantos@ucm.es

Resumen

A partir de datos de entrada/salida reales de un vehículo aéreo no tripulado (UAV) se han generado diferentes modelos mediante distintas técnicas de identificación de sistemas, incluyendo las que proporciona el soft computing. El dispositivo que se ha utilizado es un cuatrirrotor, por lo tanto un sistema complejo, no lineal, multivariable y con acoplamientos. Se han analizado y comparado los resultados de la aplicación individual y conjunta de técnicas paramétricas y de estrategias inteligentes para la obtención de modelos.

Palabras Clave: Soft computing, redes neuronales, técnicas paramétricas, modelización, identificación, vehículos aéreos no tripulados (UAV).

1 INTRODUCCIÓN

La identificación de sistemas es una de las aproximaciones más importantes para obtener modelos útiles de sistemas complejos. Fundamentalmente existen dos vías para alcanzar este objetivo: la primera de ellas consiste en aplicar las ecuaciones físicas conocidas que gobiernan el comportamiento dinámico, y emplear este conocimiento para construir las ecuaciones diferenciales que lo representan. Esta metodología resulta muy útil cuando las variables son perfectamente conocidas y las relaciones entre ellas no son demasiado complejas. La otra vía consiste en la medida y el tratamiento de las señales a la entrada y la salida del sistema, y la aplicación de diferentes mecanismos de mapeado para describir las relaciones internas subyacentes entre ellas. En esta categoría se encuentran, entre otras, algunas de las técnicas proporcionadas por la Inteligencia Artificial, como son las redes neuronales, la lógica borrosa y los algoritmos genéticos [4].

La modelización de un vehículo aéreo no tripulado, UAV (*Unmanned Aerial Vehicle*), no es una tarea sencilla. La complejidad proviene de diferentes fuentes: la aleatoriedad de los flujos de aire y de las fuerzas exógenas, la no homogeneidad de su estructura mecánica interna, y la fuerte no linealidad de la dinámica del mismo. Todos estos factores convierten a las técnicas basadas en la inteligencia artificial en un camino idóneo para identificar estos sistemas.

Otros trabajos que han aplicado estrategias inteligentes para modelar estos sistemas, en concreto las redes neuronales, son [7] y [8]. También se pueden encontrar ejemplos de la aplicación de estos métodos heurísticos para identificar otros sistemas complejos no lineales como, por ejemplo, vehículos marinos [9].

En este trabajo se realizan dos aportaciones novedosas al estudio de los UAVs: la aplicación combinada de técnicas paramétricas y neuronales como metodología de identificación, y la utilización del error cuadrático medio de la respuesta al escalón, estimada mediante análisis de correlación, y del error cuadrático medio de la respuesta estimada en frecuencia, como dos herramientas de evaluación y validación de la eficacia del modelo.

Empleando datos reales de entrada/salida de un UAV, proporcionados por el grupo de Ingeniería de Control de CEA (Comité Español de Automática) [2], se han obtenido diferentes modelos basados en redes neuronales. También se han utilizado métodos tradicionales de identificación paramétrica y los resultados se han comparado. Por último, se ha aprovechado la sinergia entre ambos métodos, combinándolos para mejorar los modelos obtenidos individualmente.

El vehículo a modelar es el ARDRONE 2.0 comercializado por Parrot (Figura 1). Este aparato se fabrica en fibra de carbono y plástico PA66. Está equipado con cuatro motores tipo *brushless* de alta eficiencia, alimentados por una batería de litio de

11.1 V y 1000 mAh que le confiere una autonomía de vuelo de aproximadamente 12 minutos. La masa es de 380 g sin carcasa y 420 g con carcasa. Dispone de un acelerómetro de 3 ejes, un giróscopo de 2 ejes (balanceo y cabeceo), y un giróscopo de precisión para el ángulo de guiñada [6].

Fig. 1: Vehículo cuatrirrotor utilizado [6].

A continuación se detalla la organización del artículo. En la sección 2 se desarrolla un análisis preliminar describiendo las señales examinadas, el análisis de correlación y el análisis espectral. La sección 3 se destina al estudio de los diferentes modelos contrastados: los basados en técnicas paramétricas, los que utilizan redes neuronales, y la aplicación conjunta de ambas metodologías. Los resultados son discutidos y comparados en la sección 4. El documento finaliza con las conclusiones.

2 ANÁLISIS PRELIMINAR

2.1 SEÑALES EMPLEADAS

Para realizar el modelado se parte de un conjunto de datos suministrados por el Grupo de Ingeniería de Control de CEA para el concurso de Ingeniería de Control 2013 [2]. El conjunto recoge los siguientes datos: periodo de muestreo, señales *pitch*, *roll*, posición (*x*,*y*), y velocidades en los dos ejes, v_x e v_y, obtenidas durante 9 minutos con un tiempo de muestreo de 60 ms.

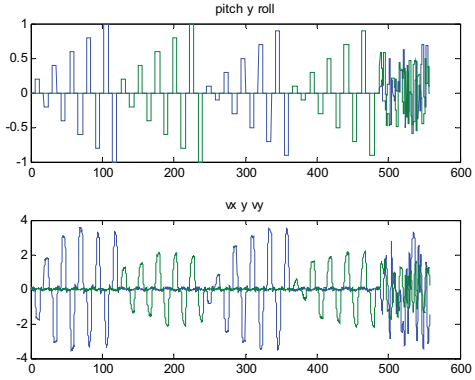

Fig. 2: Señales empleadas.

Las señales de entrada y salida se representan en la Figura 2. En azul v_x y *pitch*, y en verde v_y y *roll*. Puede adelantarse ya el importante desacoplo entre las variables que se corroborará mediante la correlación

Nos centraremos en el modelado de las velocidades v_x y v_y, puesto que la posición (*x*,*y*) viene determinada por la integración de esas velocidades.

2.2 ANÁLISIS DE CORRELACION

A partir de la estima de la covarianza entre la entrada y la salida de un sistema, y la estima de la varianza de la entrada, se puede estimar la respuesta al impulso con las expresiones siguientes:

$$\hat{R}_{yu}(\tau) = \frac{1}{N}\sum y(t)u(t-\tau) \qquad (1)$$

$$\hat{\lambda} = \frac{1}{N}\sum u(t)^2 \qquad (2)$$

$$g = \frac{\hat{R}_{yu}(\tau)}{\hat{\lambda}} \qquad (3)$$

Siendo *y* la salida del sistema, *u* la entrada, *N* el número de datos, *R* la covarianza, λ la varianza, y *g* la respuesta estimada al impulso. Integrando la respuesta al impulso puede obtenerse la respuesta al escalón, la cual puede ser muy útil para ayudarnos a estimar el grado de acoplamiento entre las variables del sistema. Esta metodología de estudio suele denominarse análisis de correlación [1].

En la respuesta al escalón de la matriz de funciones de transferencia de la Figura 3 se aprecia como la ganancia en estado estacionario de los elementos de la diagonal principal, (1,1) y (2,2), son bastante mayores que el resto. Esto indica que las dinámicas del sistema están parcialmente desacopladas, pudiendo convertir el sistema MIMO (Multi-Input Multi-Output) en dos sistemas SISO (Single-Input Single-Output).

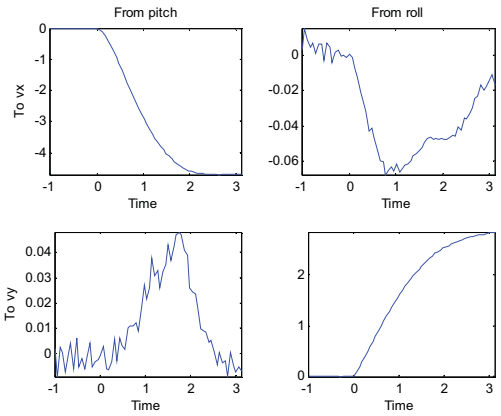

Fig. 3. Respuesta estimada a un escalón del sistema MIMO

Aprovechando esta simplificación, los esfuerzos en modelado en este trabajo se centrarán en la influencia del *pitch* sobre la velocidad en el eje *x*, y la del *roll* sobre la velocidad en el eje *y*.

2.3 ANÁLISIS ESPECTRAL

Una vez calculada la estima de la covarianza, ésta puede emplearse para calcular la respuesta estimada en frecuencia aplicando para ello la transformada de Fourier ponderada por una función de enventanado [1], como se describe en las expresiones siguientes:

$$\widehat{\Phi}_{yu}(\omega) = \sum w_M \, \hat{R}_{yu}(\tau) e^{-i\omega\tau} \qquad (4)$$

$$\widehat{\Phi}_{u}(\omega) = \sum w_M \, \hat{R}_{u}(\tau) e^{-i\omega\tau} \qquad (5)$$

$$G(e^{iwT}) = \frac{\widehat{\Phi}_{yu}(\omega)}{\widehat{\Phi}_{u}(\omega)} \qquad (6)$$

3 MODELOS DEL UAV OBTENIDOS

3.1 BASADOS EN TÉCNICAS PARAMÉTRICAS

Se han probado diferentes modelos paramétricos y se ha escogido, por su sencillez y efectividad, el modelo paramétrico ARX. Los modelos ARX (A*utoregressive Exogenous Model*), descritos en [3], representan el sistema con una ecuación de la siguiente forma:

$$Ay(t) = Bu(t) + e \qquad (7)$$

Donde *y* es la salida, *u* la entrada, *e* el error de la estima, y *A* y *B* son polinomios de ajuste. Si se restringe el grado máximo de los polinomios a 30, el mejor resultado conseguido es con la configuración ARX (10, 10, 1), donde 10 es el grado de *A* y de *B*, y 1 son las muestras de retardo.

3.2 BASADOS EN REDES NEURONALES

Se han probado distintos tipos de redes neuronales y se han escogido las redes de base radial por su aptitud para la aproximación de funciones [4, 5]. Las redes de base radial están formadas por tres capas: una de entrada donde se reciben las señales del exterior; una capa oculta, conectada con la capa de entrada donde cada elemento tiene asociada una función de base radial; y una capa de salida donde se combinan linealmente los resultados de las funciones de activación de la capa oculta [5].

La salida de cada elemento de la capa oculta, $z_i(n)$, se calcula como la distancia entre el patrón de entrada al centro del *cluster* ponderada por d_i, y aplicando este valor a una función de base radial, según la siguiente expresión:

$$z_i(n) = \Phi\left(\frac{\sqrt{\sum(x_j(n)-c_{ij})^2}}{d_i}\right) \qquad (8)$$

$$y_k(n) = \sum w_{ik} z_i(n) + \mu_k \qquad (9)$$

Siendo w_{ik} los pesos de la capa de salida y μ_k los umbrales de activación. La función de base radial Φ utilizada es:

$$\Phi(n) = e^{-n^2} \qquad (10)$$

Para realizar la identificación del modelo el conjunto de datos de entrada de la red se ha formado con la señal de entrada en el instante actual, *pitch(t)*, y valores en los 10 instantes de muestreo anteriores, *pitch(t-iT$_s$)* donde i = 1…10, y *Ts* es tiempo de muestreo. El conjunto de datos objetivo se ha generado con los valores de la señal de salida en el instante actual $v_x(t)$. Un proceso similar se ha llevado a cabo para la relación *roll* → v_y. Para simplificar el modelo y establecer las comparaciones en términos justos se ha limitado el número de neuronas de la capa oculta a 50.

Se han empleado 5000 muestras de las señales, el 60% para entrenar las redes y el resto para su simulación y validación.

3.3 APLICACIÓN COMBINADA DE TÉCNICAS PARAMÉTRICAS Y NEURONALES

La combinación de las técnicas anteriores empleando la topología y la metodología adecuadas proporciona mejores resultados en términos de complejidad computacional y del error cometido en el modelo. En este estudio se han empleado dos topologías de combinación diferentes: la conexión en cascada y la conexión en paralelo. En las figuras 4 y 5 se representan ambas configuraciones.

En el primer caso, configuración en cascada, el entrenamiento de la red se realiza de la forma siguiente:

- Se realiza la identificación del modelo paramétrico que mejor aproxime a las señales.
- Se simula la salida esperada del sistema a partir de un conjunto de señales de entrada conocido.
- Se entrena la red neuronal tomando como conjunto de entrada el resultado de la simulación anterior, y como conjunto de salida los datos de salida reales conocidos.

Para la conexión en cascada se substituye el tercer paso por:

- Se entrena la red neuronal tomando como conjunto de entrada el conjunto de señales de entrada conocido del paso 2, y como conjunto de salida la señal de error entre la salida esperada del paso 2 y los datos de salida reales conocidos del paso 3.

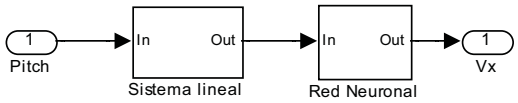

Fig. 4. Conexión en cascada de las dos técnicas de modelado.

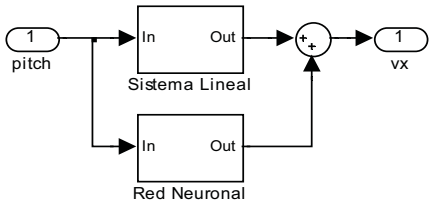

Fig. 5. Conexión en paralelo de las dos técnicas de modelado.

Para que las comparaciones entre los modelos de las secciones 3.1 y 3.2 y la técnica combinada sean equiparables, se escoge como modelo lineal el ARX-10,10,1 y como red neuronal la red de base radial con 30 neuronas en la capa oculta.

4 RESULTADOS

En esta sección se presenta una comparación de los modelos obtenidos por las diferentes técnicas expuestas. Todas las estrategias se han implementado en Matlab/Simulink. Los modelos se evalúan tanto en el dominio del tiempo como en el de la frecuencia. Se define el error como la diferencia entre la salida real del sistema (datos experimentales proporcionados por los ensayos de la organización) y la salida del modelo correspondiente. Esta diferencia también se calcula para la respuesta al escalón y para la respuesta estimada en frecuencia.

Las figuras 6 y 7 muestran la respuesta del sistema (velocidad en el eje *x*, figura 6, y velocidad en el eje *y*, figura 7) cuando se aplican diferentes escalones como señal de referencia. Si ordenamos los resultados en base a la similitud con la salida real, en ambos casos la conexión en cascada es la que mejor comportamiento proporciona, seguida por la conexión en paralelo, base radial, y por último la identificación mediante ARX.

Por lo tanto la utilización combinada de técnicas paramétricas y redes neuronales da mejores resultados que la utilización individual de las mismas.

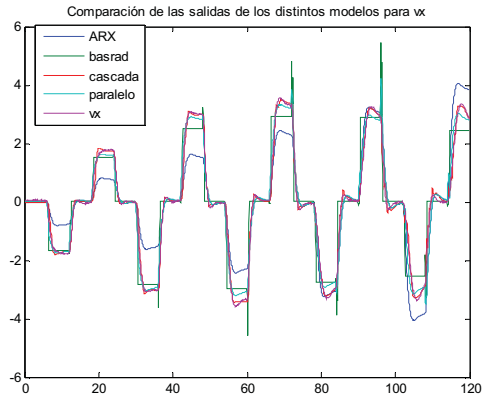

Fig. 6. Comparación de las salidas de los modelos para v_x.

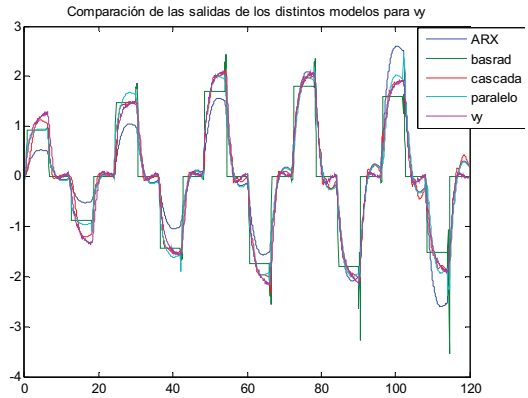

Fig. 7. Comparación de las salidas de los modelos para v_y.

Las respuestas al escalón se representan en las figuras 8 (velocidad en el eje x) y 9 (velocidad en el eje y). El mejor resultado para v_x es el proporcionado por la conexión en paralelo seguida por la conexión en cascada, ARX, y base radial. Para v_y es la conexión en cascada la que mejor se aproxima a la respuesta al escalón estimada, seguida por ARX, conexión en paralelo y base radial.

En este caso queda patente la asimetría del sistema, ya que se obtienen diferentes aproximaciones en una dirección que en otra, y unas técnicas identifican mejor según la variable, y por lo tanto el eje, del que se trate.

Las figuras 10 (velocidad en el eje x) y 11 (velocidad en el eje y) muestran la comparación de la respuesta en frecuencia estimada para los diferentes modelos estudiados. En ambos casos el modelo que más se acerca a la respuesta en frecuencia estimada mediante

los datos reales de entrada y salida es el ARX seguido por la conexión en cascada, la conexión en paralelo y base radial.

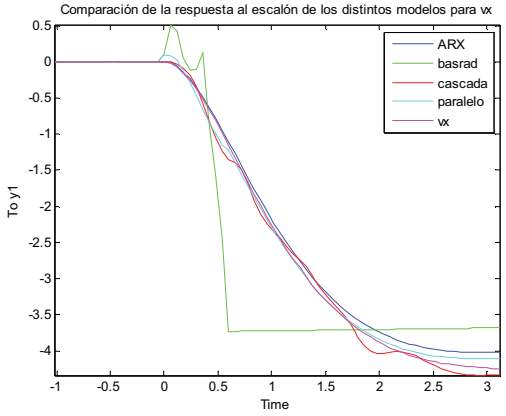

Fig. 8. Comparación de la respuesta al escalón de los modelos para v_x.

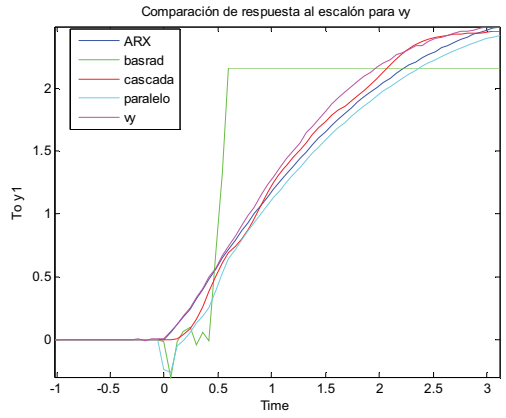

Fig. 9. Comparación de la respuesta al escalón de los modelos para v_y.

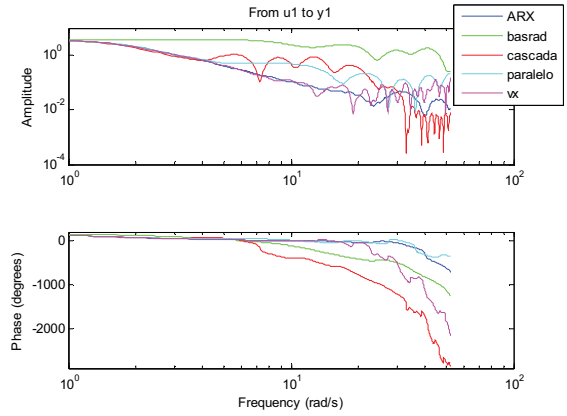

Fig. 10. Comparación de la respuesta en frecuencia de los modelos para v_x.

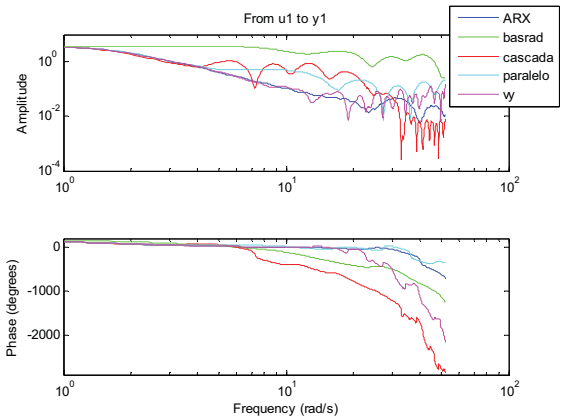

Fig. 11. Comparación de la respuesta en frecuencia de los modelos para v_y.

Por lo tanto en el dominio de la frecuencia se puede concluir que se obtienen mejores resultados con técnicas clásicas para esta aplicación con los datos disponibles.

La tabla 1 presenta los resultados cuantitativos en términos de error cuadrático medio, MSE (*Mean Squared Error*) para cada técnica estudiada.

Tabla 1: Comparación del MSE para valores en el dominio del tiempo, la respuesta al escalón estimada y la respuesta en frecuencia estimada.

Técnica	**Salida**	**MSE**	**MSE respuesta al escalón**	**MSE respuesta en frecuencia**
ARX10,10,1	Vx	0.5056	0.0122	0.0014
ARX10,10,1	Vy	0.1440	0.0085	0.0032
Base radial	Vx	0.3074	0.6680	3.2289
Base radial	Vy	0.1432	0.2271	1.4363
Cascada	Vx	0.0155	0.0080	0.0767
Cascada	Vy	0.0207	0.0048	0.0100
Paralelo	Vx	0.0815	0.0050	0.0252
Paralelo	Vy	0.0337	0.0269	0.0800

Los resultados muestran que las topologías en cascada y en paralelo proporcionan mejores resultados que la utilización independiente de la red de base radial, que es la que proporciona peores resultados tanto en el dominio del tiempo como para el dominio de la frecuencia.

La comparativa indica que en el caso del ARX, cuando se emplea la conexión en cascada, se consiguen mejores resultados en lo que respecta al MSE y a la respuesta al escalón. En el caso de la respuesta en frecuencia se ha observado que el modelo ARX presenta un mejor comportamiento que

el ofrecido por el resto de los modelos. Esta diferencia se habría visto reducida si se hubiera aumentado el número de neuronas de la capa oculta de la red de base radial.

5 CONCLUSIONES

Los UAVs son sistemas difíciles de modelar. Su complejidad deriva de la aleatoriedad de los flujos de aire y de las fuerzas exógenas, y de su dinámica fuertemente no lineal. Estos factores señalan a las técnicas inteligentes como estrategias eficientes para modelar estos sistemas. Para probarlo, en este trabajo se han implementado un conjunto de técnicas tanto analíticas como inteligentes que permiten identificar cualquier tipo de UAV en base a datos experimentales.

En concreto, para modelar el UAV objeto de estudio del que se cuenta con datos reales de ensayos, se han implementado y comparado técnicas paramétricas y modelos basados en redes neuronales. Se han aplicado tanto en el dominio del tiempo como en el dominio de la frecuencia.

Por otro lado, se han combinado las técnicas paramétricas con las inteligentes, en dos configuraciones sencillas, en serie o cascada y en paralelo.

Los resultados son altamente satisfactorios: el error MSE –que se ha usado como criterio para evaluar los modelos- obtenido mediante la aplicación conjunta de las técnicas paramétricas y neuronales indica que esto modelos proporcionan un comportamiento satisfactorio tanto para la secuencia temporal como para la respuesta en frecuencia.

En última instancia se ha contrastado la idoneidad de las técnicas basadas en redes neuronales para identificar y modelar estos sistemas, y se ha probado como la sinergia de técnicas clásicas e inteligentes puede ser una alternativa muy eficiente en problemas complejos.

AGRADECIMIENTOS

A los autores les gustaría agradecer al Grupo de Ingeniería de Control del Comité Español de Automática [2] la aportación del fichero con datos reales empleado para modelar el UAV.

REFERENCIAS

[1] Bendat, J. S., & Piersol, A. G. (1980). Engineering applications of correlation and spectral analysis. New York, Wiley-Interscience, 1980. 315 p.

[2] Grupo Ingeniería de Control (2013), Comité Español de Automática (CEA), www.ceautomatica.es/og/ingenieria-de-control.

[3] Isaksson, A. J. (1993). Identification of ARX-models subject to missing data. Automatic Control, IEEE Transactions on, 38(5), 813-819.

[4] Nørgård, P. M., Ravn, O., Poulsen, N. K., Hansen, L. K. (2000). Neural Networks for Modelling and Control of Dynamic Systems-A Practitioner's Handbook. Springer-London.

[5] Park, J., Sandberg, I. W. (1991). Universal approximation using radial-basis-function networks. Neural computation, 3(2), 246-257.

[6] Parrot. www.parrot.com (2013)

[7] San Martin, R., Barrientos, A., Gutierrez, P., Del Cerro, J. (2006). Neural networks training architecture for UAV modelling. In: IEEE World Automation Congress WAC'06, 1-6.

[8] San Martin, R., Barrientos, A., Gutierrez, P., Del Cerro, J. (2006). Unmanned aerial vehicle (UAV) modelling based on supervised neural networks. In: Proc. IEEE Int. Conference on Robotics and Automation ICRA 2006, 2497-2502.

[9] Santos, M., López, R., & De La Cruz, J. M. (2006). A neuro-fuzzy approach to fast ferry vertical motion modelling. Engineering Applications of Artificial Intelligence, 19(3), 313-321.

5 Modelado de un vehículo aéreo no tripulado mediante combinación de técnicas paramétricas y neurodifusas

BIBLIOGRAPHIC DESCRIPTION

Title

Modelado de un vehículo aéreo no tripulado mediante combinación de técnicas paramétricas y neurodifusas.

Citation

Sierra, J.E., Santos, M. (2014) Modelado de un vehículo aéreo no tripulado mediante combinación de técnicas paramétricas y neurodifusas. In Actas del XVII Español sobre Tecnologías y Lógica Fuzzy, pages 339-344, 2014. ISBN: 978-84-15688-76-1.

Abstract

A partir de datos de entrada/salida reales de un vehículo aéreo no tripulado (UAV) se han generado diferentes modelos mediante distintas técnicas de identificación de sistemas, incluyendo las que proporciona el soft computing. El dispositivo que se ha utilizado es un cuatrirrotor, por lo tanto un sistema complejo, no lineal, multivariable y con acoplamientos. Se han analizado y comparado los resultados de la aplicación conjunta de técnicas paramétricas y basadas en redes neurodifusas para la obtención de modelos orientados al control.

References

Bendat et al(1980); Isaksson,A.J. (1993); Jang, J-S. R. et al (1993); Norgard, P.M. et al (2000); Park, J. et al (1991); Rutkowski, L. (2004); San Martin, R. et al (2006); Santos et al (2006)

MODELADO DE UN VEHÍCULO AÉREO NO TRIPULADO MEDIANTE COMBINACIÓN DE TÉCNICAS PARAMÉTRICAS Y NEURODIFUSAS

Jesús Enrique Sierra[1], **Matilde Santos**[2]

[1]*Universidad de Burgos, Burgos*, jesierra@ubu.es
[2]*Universidad Complutense de Madrid, Madrid*, msantos@ucm.es

Resumen

A partir de datos de entrada/salida reales de un vehículo aéreo no tripulado (UAV) se han generado modelos orientados al control mediante técnicas paramétricas, neuronales y neurodifusas. El dispositivo empleado es un cuatrirrotor, por lo tanto un sistema complejo, no lineal, multivariable y con acoplamientos. Los resultados de la aplicación de los diferentes modelos se han analizado y comparado en los dominios temporal y frecuencial. Esta comparación señala que la combinación de los métodos ARX y ANFIS proporciona una excelente aproximación en ambos dominios.

Palabras Clave: Soft computing, ANFIS, redes neuronales, redes neurodifusas, técnicas paramétricas, modelización, identificación, vehículos aéreos no tripulados (UAV).

1 INTRODUCCIÓN

La identificación de sistemas es uno de los pasos más importantes para obtener modelos útiles de sistemas complejos. Fundamentalmente existen dos vías para alcanzar este objetivo: la primera de ellas consiste en aplicar las ecuaciones físicas conocidas que gobiernan el comportamiento dinámico y emplear este conocimiento para construir las ecuaciones diferenciales que lo representan. Esta metodología resulta muy útil cuando las variables son perfectamente conocidas y las relaciones entre ellas no son demasiado complejas. La otra vía consiste en la medida y el tratamiento de las señales en la entrada y la salida del sistema, y la aplicación de diferentes mecanismos de mapeado para describir las relaciones internas subyacentes entre las entradas y las salidas. En esta categoría se encuentran, entre otras, algunas de las técnicas proporcionadas por la Inteligencia Artificial, como son las redes neuronales, redes neurodifusas y los algoritmos genéticos [5].

La modelización de un vehículo aéreo no tripulado, UAV (Unmanned Aerial Vehicle), no es una tarea sencilla. La complejidad proviene de diferentes fuentes: la aleatoriedad de los flujos de aire y de las fuerzas exógenas, la no homogeneidad de su estructura mecánica interna, y la fuerte no linealidad de la dinámica del mismo. Todos estos factores convierten a las técnicas basadas en la inteligencia artificial en un camino idóneo para modelar estos sistemas.

Otros estudios anteriores se enfrentaron a este problema de modelización empleando redes neuronales como: [7] y [9]. También pueden encontrarse otros ejemplos de aplicación de técnicas inteligentes para modelar sistemas complejos no lineales como, por ejemplo, vehículos marinos [8]. Mediante este trabajo se realizan dos aportaciones novedosas en el estudio de los UAVs: la aplicación combinada de técnicas paramétricas y neurodifusas como metodología de modelización, y la utilización del error cuadrático medio de la respuesta al escalón, estimada mediante análisis de correlación, y del error cuadrático medio de la respuesta en frecuencia estimada, como dos herramientas de evaluación y validación de la eficacia del modelo.

En este trabajo, empleando datos reales de entrada/salida de un UAV, se han obtenido diferentes modelos basados en técnicas paramétricas, redes neuronales y redes neurodifusas, orientados todos ellos al control, y los resultados se han comparado. Por último se ha aplicado la sinergia de ambas aproximaciones, combinando los métodos, para mejorar los modelos obtenidos por cada una de ellas de forma individual.

El vehículo a modelar es el ARDRONE 2.0 comercializado por Parrot (Figura 1). Este aparato se fabrica en fibra de carbono y plástico PA66. Está equipado con cuatro motores tipo brushless de alta eficiencia, alimentados por una batería de litio de 11.1 V y 1000 mAh que le confiere una autonomía de vuelo de aproximadamente 12 minutos. La masa es de 380 g sin carcasa y 420 g con carcasa. Dispone de un acelerómetro de 3 ejes, un giróscopo de 2 ejes (balan-

ceo y cabeceo), y un giróscopo de precisión para el ángulo de guiñada.

Figura 1: Vehículo cuatrirrotor utilizado (`http://www.parrot.com`).

A continuación se detalla la organización del documento. En la sección 2 se desarrolla un análisis preliminar describiendo las señales examinadas, el análisis de correlación y el análisis espectral. La sección 3 se destina al estudio de los diferentes modelos contrastados: basados en técnicas paramétricas, basados en redes neuronales y neurodifusas y la aplicación conjunta de ambas metodologías. Los resultados son discutidos y comparados en la sección 4. El documento finaliza con las conclusiones.

2 ANÁLISIS PRELIMINAR

2.1 SEÑALES EMPLEADAS

Para realizar el modelado se parte de un conjunto de datos suministrados por el Grupo de Ingeniería de Control de CEA (Comité Español de Automática) para el concurso de Ingeniería de Control 2013 [2]. El conjunto recoge los siguientes datos: periodo de muestreo, señales pitch, roll, x, v_x, y e v_y, obtenidas durante 9 minutos con un tiempo de muestreo de 60 ms. Las señales de entrada y salida se representan en la Figura 2 (en azul: pitch y v_x, en verde: roll y v_y).

Puede adelantarse ya el importante desacoplo entre las variables que se corroborará mediante la correlación. Nos centraremos en el modelado de las velocidades v_x y v_y, puesto que la posición (x,y) viene determinada por la integración de esas velocidades.

2.2 ANÁLISIS DE CORRELACIÓN

A partir de la estima de la covarianza entre la entrada y la salida de un sistema, y la estima de la varianza de la entrada, se puede estimar la respuesta al impulso, con las

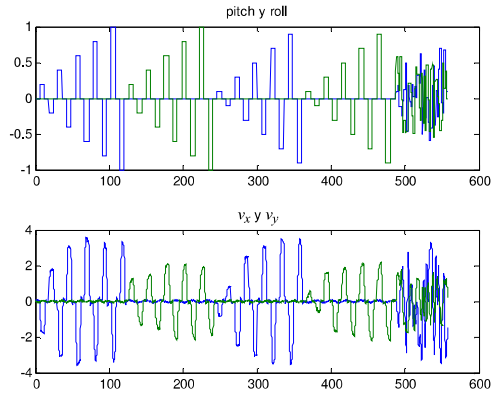

Figura 2: Señales empleadas.

expresiones siguientes:

$$\widehat{R}^N_{yu}(\tau) = \frac{1}{N}\sum_{t=1}^{N} y(t)(t-\tau) \tag{1}$$

$$\widehat{\lambda}^N = \frac{1}{N}\sum_{t=1}^{N} u(t)^2 \tag{2}$$

$$\widehat{g}^N_\tau = \frac{\widehat{R}^N_{yu}(\tau)}{\widehat{\lambda}^N} \tag{3}$$

Siendo y la salida del sistema, u la entrada, N el número de datos, $\widehat{R}$ la covarianza, $\widehat{\lambda}$ la varianza y $\widehat{g}$ la respuesta estimada al impulso. Integrando la respuesta al impulso puede obtenerse la respuesta al escalón, la cual puede ser muy útil para ayudarnos a estimar el grado de acoplamiento entre las variables del sistema. Esta metodología de estudio suele denominarse análisis de correlación [1]. Empleando el método descrito se obtiene la Figura 3.

En la respuesta al escalón de la matriz de funciones de transferencia de la Figura 3 se aprecia como la ganancia en estado estacionario de los elementos de la diagonal principal, correspondientes al enlace (fila 1, columna 1) y (fila 2, columna 2), son bastante mayores que el resto. Este hecho indica que las dinámicas del sistema están parcialmente desacopladas, pudiendo convertir el sistema MIMO (multi-input multi-output) en dos sistemas SISO (singleinput single-output). Aprovechando esta simplificación, los esfuerzos en modelado en este trabajo se centrarán en la influencia del pitch sobre la velocidad en el eje x y del roll sobre la velocidad en el eje y.

2.3 ANÁLISIS ESPECTRAL

Una vez calculada la estima de la covarianza, ésta puede emplearse para calcular la respuesta en frecuencia estimada [1] aplicando para ello la transformada de Fourier ponderada por una función de *enventanado*, como se describe en las expresiones siguientes:

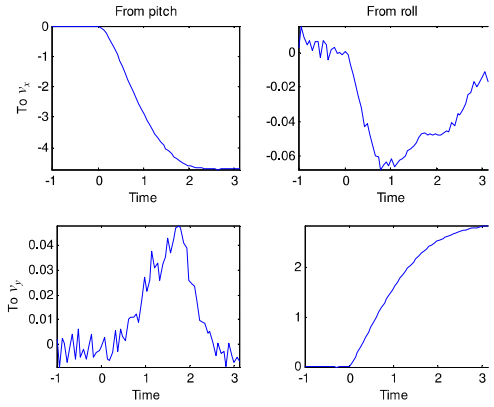

Figura 3: Respuesta a un escalón estimada del sistema MIMO

$$\widehat{\Phi}_{yu}^{N}(\omega) = \sum_{\tau=-M}^{M} w_M \widehat{R}_{yu}^{N}(\tau) e^{-i\omega\tau} \tag{4}$$

$$\widehat{\Phi}_{u}^{N}(\omega) = \sum_{\tau=-M}^{M} w_M \widehat{R}_{u}^{N}(\tau) e^{-i\omega\tau} \tag{5}$$

$$G^{N}(e^{iwT}) = \frac{\widehat{\Phi}_{yu}^{N}(\omega)}{\widehat{\Phi}_{u}^{N}(\omega)} \tag{6}$$

3 MODELOS

3.1 BASADOS EN TÉCNICAS PARAMÉTRICAS

Se han probado diferentes modelos paramétricos y se ha escogido, por su sencillez y efectividad, el modelo paramétrico ARX. Los modelos ARX (Autoregressive Exogenous Model), descritos en [3], describen el sistema con una ecuación de la siguiente forma:

$$Ay(y) = Bu(t) + \varepsilon \tag{7}$$

Donde y es la salida, u la entrada, ε el error de la estima, y A y B son polinomios. Si se restringe el grado máximo de los polinomios a 30, el mejor resultado se consigue con la configuración ARX $(10, 10, 1)$ donde 10 es el grado de A y de B, y 1 son las muestras de retardo.

3.2 BASADOS EN REDES NEURONALES

Se han probado distintos tipos de redes neuronales y se han escogido las redes de base radial por su aptitud para la aproximación de funciones, [4, 5]. Las redes de base radial están formadas por tres capas, una capa de entrada donde se reciben las señales del exterior; una capa oculta, conectada con la capa de entrada donde cada elemento tiene asociada una función de base radial; y una capa de salida donde se combinan linealmente los resultados de las funciones de activación de la capa oculta [6].

La salida de cada elemento de la capa oculta zi(n) se calcula como la distancia entre el patrón de entrada al centro del cluster ponderada por di, y aplicando este valor a una función de base radial, según la siguiente expresión:

$$z_i(n) = \Phi\left(\frac{\sqrt{\sum\left(x_j(n) - c_{ij}\right)^2}}{d_i}\right) \tag{8}$$

$$y_k(n) = \sum w_{ik} z_i(n) + \mu_k \tag{9}$$

Siendo w_{ik} los pesos de la capa de salida y μ_k los umbrales de activación. La función de base radial Φ utilizada es:

$$\Phi(n) = e^{-n^2} \tag{10}$$

Para realizar el estudio, el conjunto de datos de entrada de la red se ha formado con la señal de entrada en el instante actual, $pitch(t)$, y valores en los 5 instantes de muestreo anteriores, $pitch(t - iTs)$ donde $i = 1 \ldots 5$, y Ts es tiempo de muestreo. El conjunto de datos objetivo se ha generado con los valores de la señal de salida en el instante actual $v_x(t)$. Un proceso similar se ha llevado a cabo para la relación $roll \rightarrow v_y$. Para simplificar el modelo y establecer las comparaciones en términos justos se ha limitado el número de neuronas de la capa oculta a 30.

Se han empleado 5000 muestras de las señales, el 60 % se ha empleado para entrenar las redes y el resto para simular y validarlas.

3.3 BASADOS EN TÉCNICAS NEURODIFUSAS

Se ha empleado la técnica ANFIS (Adaptive Neural Fuzzy Inference System) [4]. A grandes rasgos se basa en la creación de un árbol de decisión para clasificar los datos de entre p^n modelos de regresión. Donde n es el número de entradas del modelo y p es el número de particiones de cada variable.

Estas redes están formadas por cinco capas. En la capa de entrada (capa de valores) se *fuzzifican* las entradas. Esta capa se compone de p^n nodos, donde la salida de cada uno de ellos viene dada por:

$$O_{1,i} = \mu_{Aj}(x_k),\ i = 1, \ldots p^n, j = 1, \ldots p, k = 1, \ldots n \tag{11}$$

Donde A es una etiqueta lingüística (pequeño, grande, ...), y la función μ viene dada por:

$$\mu_A(x_k) = \frac{1}{1 + \left| \frac{x_k - c_i}{a_i} \right|^{2b}} \quad (12)$$

Con las salidas de los nodos de la primera capa se construyen las reglas difusas en la segunda capa (capa de reglas) a través de alguna T-norma (mínimo, producto, ...):

$$O_{2,i} = w_i = T(\mu_{Aj}, \mu_{Bj}, \ldots),\ i = 1, \ldots p^n, j = 1, \ldots p \quad (13)$$

En la capa 3 (capa de normalización) se normalizan las salidas de la capa 2:

$$O_{3,i} = \overline{w}_i = \frac{w_i}{\sum w_i},\ i = 1, \ldots p^n \quad (14)$$

En la capa 4 (capa de funciones), las salidas de la capa 3 ($\overline{w}_i$), ponderan los resultados de la función de regresión lineal de su nodo generando las reglas de salida:

$$O_{4,i} = \overline{w}_i f_i = \overline{w}_i \left(p_{1_i} x_i + \cdots p_{n_i} x_n \right),\ i = 1, \ldots p^n \quad (15)$$

Donde $p_k, k = 1, \ldots n$ son los parámetros consecuencia. Finalmente, en la capa de salida se realiza una suma ponderada de las salidas de la capa 4:

$$O_5 = \sum_i \overline{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \quad (16)$$

En nuestro caso el conjunto de entrenamiento es el mismo que para la red de funciones de base radial. Con lo que asignando un número de particiones $p = 2$, se obtienen un conjunto de $2^6 = 64$ reglas. Se fija el número de iteraciones a 1 para reducir la carga computacional.

3.4 APLICACIÓN COMBINADA DE TÉCNICAS

La combinación de las técnicas anteriores empleando la topología y la metodología adecuadas proporciona mejores resultados en términos de complejidad computacional y del error cometido en el modelo. En este estudio se ha empleado la conexión en cascada de la Figura 4.

En el primer caso, configuración en cascada, el entrenamiento de la red se realiza de la forma siguiente: *i*) se realiza la identificación del modelo paramétrico que mejor se aproxime, *ii*) a continuación se simula la salida esperada del sistema a partir de un conjunto de señales de entrada conocido, *iii*) finalmente, se entrena la red tomando como conjunto de entrada el resultado de la simulación anterior, y como conjunto de salida los datos de salida reales conocidos.

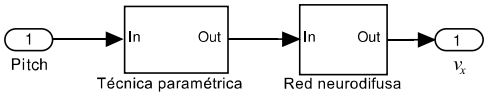

Figura 4: Conexión en cascada de las dos técnicas de modelado

Para que las comparaciones entre los modelos de las secciones 3.1 y 3.2 y la técnica combinada sean equiparables, se escoge como modelo lineal el ARX-10,10,1 y como red neuronal la red de base radial con 30 neuronas en la capa oculta.

4 RESULTADOS

En esta sección se presenta una comparación de los modelos obtenidos por las diferentes técnicas expuestas. Todas las estrategias se han implementado en Matlab/Simulink. Los modelos se evalúan tanto en el dominio del tiempo como en el de la frecuencia. Se define el error como la diferencia entre la salida real del sistema (datos experimentales proporcionados por los ensayos de la organización) y la salida del modelo correspondiente. Esta diferencia también se calcula para la respuesta al escalón estimada y para la respuesta en frecuencia estimada.

Las figuras 5 y 6 muestran la respuesta del sistema (velocidad en el eje x, Figura 5 y velocidad en el eje y, Figura 6) cuando se aplican diferentes escalones como señal de referencia. Si ordenamos los resultados en base a la similitud con la salida real, en ambos casos la conexión ARX+ANFIS es la que mejor comportamiento proporciona, seguida por la conexión ARX + base radial, y el resto de técnicas.

Las respuestas al escalón se representan en las figuras 7 (velocidad en el eje x) y 8 (velocidad en el eje y). El mejor resultado para vx es el proporcionado por la conexión ARX+ANFIS, seguida por ARX, ARX + base radial, ANFIS y base radial. Para vy es ARX+base radial la conexión que mejor se aproxima a la respuesta al escalón estimada, seguida por ANFIS, ARX+ANFIS, ARX y base radial.

En este caso queda patente la asimetría del sistema, ya que se obtienen diferentes aproximaciones en una dirección que en otra, y unas técnicas identifican mejor según la variable, y por lo tanto el eje, del que se trate.

Las figuras 9 (velocidad en el eje x) y 10 (velocidad en el eje y) muestran la comparación de la respuesta en frecuencia estimada para los diferentes modelos estudiados. En ambos casos el modelo que más se acerca a la respuesta en frecuencia estimada mediante los datos reales de entrada

y salida es el ARX seguido por la conexión ARX + base radial, ARX+ANFIS, base radial y ANFIS. Por lo tanto en el dominio de la frecuencia se puede concluir que se obtienen mejores resultados con técnicas clásicas.

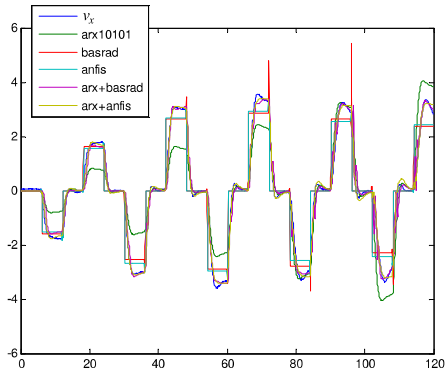

Figura 5: Comparación de las salidas de modelos para v_x.

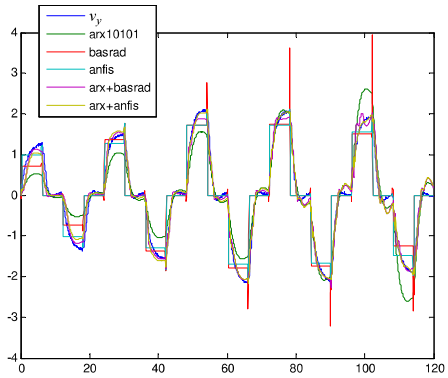

Figura 6: Comparación de las salidas de modelos para v_y.

Los resultados muestran que de forma general, la conexión en cascada proporciona mejores resultados que la utilización independiente de la red de base radial, ANFIS o ARX.

La tabla 1 presenta los resultados cuantitativos en términos de error cuadrático medio, MSE (*Mean Squared Error*) para cada técnica estudiada. Se ha escogido este estadístico por su denotada utilidad en la comparación de modelos cuando sólo se dispone de un conjunto de datos. ([5] y [6])

La comparativa general arroja que el modelo que mejor se comporta es el ARX+ANFIS. En el caso de la respuesta en frecuencia se ha observado que el modelo ARX presenta un mejor comportamiento que el ofrecido por el resto de los modelos. Esta diferencia se habría visto reducida si se hubiera aumentado el número de neuronas de la capa oculta de la red de base radial, o el número de particiones o iteraciones en ANFIS.

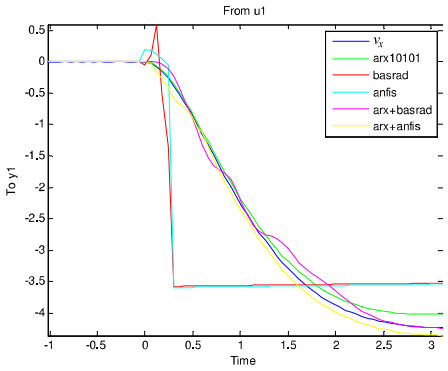

Figura 7: Comparación de la respuesta al escalón de modelos para v_x.

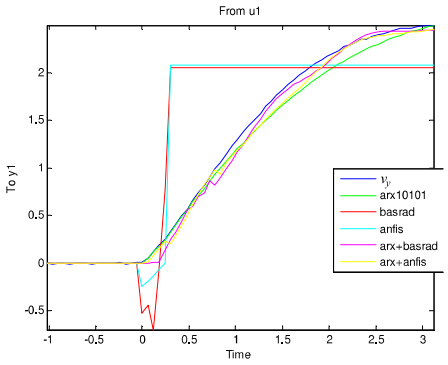

Figura 8: Comparación de la respuesta al escalón de modelos para v_y.

5 CONCLUSIONES

Los UAVs son sistemas muy complejos de modelar. Su complejidad deriva de la aleatoriedad de los flujos de aire y de las fuerzas exógenas, y de su dinámica fuertemente no lineal. Estos factores señalan a las técnicas inteligentes como estrategias eficientes para modelar estos sistemas. Para probarlo, en este trabajo se han implementado un conjunto de herramientas informáticas que permiten modelar cualquier tipo de UAV en base a datos experimentales con diferentes técnicas, analíticas e inteligentes.

En concreto, para modelar el UAV objeto de estudio, del que se cuentan con datos reales de ensayos, se han usado diferentes técnicas de identificación de sistemas. Por un lado se han implementado y comparado técnicas paramétricas y modelos basados en redes neurodifusas. Se han aplicado tanto en el dominio del tiempo como en el dominio de la frecuencia.

Por otro lado, se han combinado las técnicas paramétricas con las inteligentes consiguiendo una mejora en la aproximación importante.

Los resultados son altamente satisfactorios: el error MSE

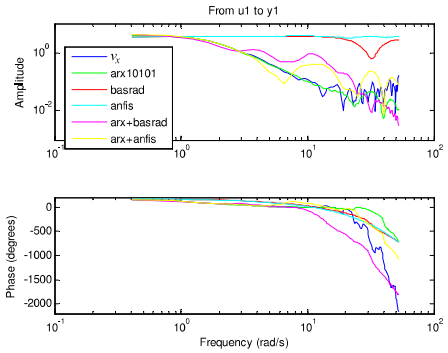

Figura 9: Comparación de la respuesta en frecuencia de los modelos para v_x.

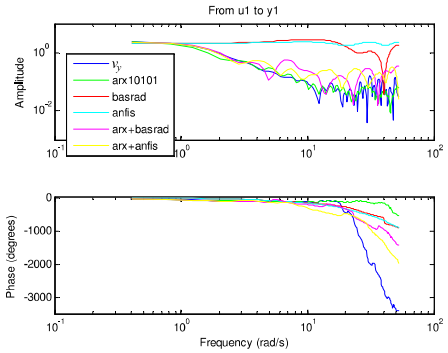

Figura 10: Comparación de la respuesta en frecuencia de los modelos para v_y.

—empleado como criterio evaluador de los modelos— obtenido mediante la aplicación combinada de técnicas indica que estos modelos proporcionan un comportamiento satisfactorio tanto para la secuencia temporal como para la respuesta en frecuencia, especialmente la combinación de ARX + ANFIS.

En última instancia se ha probado como la sinergia de técnicas clásicas e inteligentes puede ser una alternativa muy eficiente en el modelado de sistemas complejos

Agradecimientos

A los autores les gustaría agradecer la aportación del fichero con datos reales al Grupo de Ingeniería de Control del Comité Español de Automática [2].

Referencias

[1] Julius S. Bendat and Allan G. Piersol. *Engineering applications of correlation and spectral analysis*. Wiley, New York, NY [u.a.], 1980.

[2] Comité español de automática. Bases del concurso en ingeniería de control, 2013. `http://www.ceautomatica.es/og/ingenieria-de-control`

Tabla 1: Comparación del MSE para valores en el dominio del tiempo, la respuesta al escalón estimada y la respuesta en frecuencia estimada.

Técnica	Salida	MSE	MSE respuesta al escalón	MSE respuesta en frecuencia	MSE medio
ARX10,10,1	v_x	0.5056	0.0122	**0.0014**	0.1731
ARX10,10,1	v_y	0.1440	0.0085	**0.0032**	0.0519
Base radial	v_x	0.5097	1.1919	6.1121	2.6046
Base radial	v_y	0.2345	0.3796	2.6912	1.1018
ANFIS	v_x	0.4816	1.1807	11.5394	4.4006
ANFIS	v_y	0.3765	**0.0048**	3.9787	1.5219
ARX+Basrad	v_x	0.0572	0.0172	0.0192	0.0493
ARX+Basrad	v_y	0.0248	0.0045	0.0204	0.0182
ARX+ANFIS	v_x	**0.0145**	**0.0087**	0.0735	**0.0141**
ARX+ANFIS	v_y	**0.0227**	0.0052	0.0254	**0.0161**

[3] A.J. Isaksson. Identification of arx-models subject to missing data. *Automatic Control, IEEE Transactions on*, 38(5):813–819, 1993.

[4] J.-S.R. Jang. Anfis: adaptive-network-based fuzzy inference system. *Systems, Man and Cybernetics, IEEE Transactions on*, 23(3):665–685, 1993.

[5] Magnus Nørgård, O. E. Ravn, N. K. Poulsen, and L. K. Hansen. *Neural Networks for Modelling and Control of Dynamic Systems: A Practitioner's Handbook*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st edition, 2000.

[6] J. Park and I. W. Sandberg. Universal approximation using radial-basis-function networks. *Neural Comput.*, 3(2):246–257, June 1991.

[7] R. San Martin, A. Barrientos, P. Gutierrez, and J. Del Cerro. Unmanned aerial vehicle (UAV) modelling based on supervised neural networks. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 2497–2502, 2006.

[8] M. Santos, R. López, and J.M de la Cruz. A neuro-fuzzy approach to fast ferry vertical motion modelling. *Eng. Appl. Artif. Intell.*, 19(3):313–321, April 2006.

[9] J.E. Sierra, M. Santos. Estudio comparativo de modelos de un vehículo aéreo obtenidos mediante técnicas analíticas y basadas en redes neuronales. In *Actas Multiconferencia CAEPIA'13 (XV Conferencia de la Asociación Española para la Inteligencia Artificial), ISBN: 978-84-695-8348-7*, pages 1270-1279, 2013.

6 Adaptive Neural Control-Oriented Models of Unmanned Aerial Vehicles

BIBLIOGRAPHIC DESCRIPTION

Title

Adaptive Neural Control-Oriented Models of Unmanned Aerial Vehicles.

Citation

Sierra, J. E., & Santos, M. (2015). Adaptive Neural Control-Oriented Models of Unmanned Aerial Vehicles. In 10th International Conference on Soft Computing Models in Industrial and Environmental Applications (pp. 329-337). Springer International Publishing.

Abstract

From real input/output data, different models of an unmanned aerial vehicle are obtained by applying adaptive neural networks. These models are control-oriented; their main objective is to help us to design, implement and simulate different intelligent controllers and to test them on real systems. They have been compared to off-line learning neural models with satisfactory results in terms of accuracy and computational cost.

References

Billings, S. A. (2013); Deng, J. (2013); Han, H.G. et al (2012); Hoffer, N. V. et al (2014); Nemes, A. (2015); San Martin R. et al (2006); Santos M. et al (2006); Santos, M. (2011); Sierra J.E. et al (2013); Sayama H. et al (2013); Ugalde, H. M. R. et al (2013).

Adaptive Neural Control-Oriented Models of Unmanned Aerial Vehicles

J. Enrique Sierra and Matilde Santos

Abstract From real input/output data, different models of an unmanned aerial vehicle are obtained by applying adaptive neural networks. These models are control-oriented; their main objective is to help us to design, implement and simulate different intelligent controllers and to test them on real systems. The influence of the selected training data on the final model is also discussed. They have been compared to off-line learning neural models with satisfactory results in terms of accuracy and computational cost.

Keywords Adaptive neural networks · Soft computing · Modeling · Identification · Unmanned aerial vehicles (UAV)

1 Introduction

One of the main steps in the design of a controller is to obtain a good model of the system dynamic. In control engineering, system identification is one of the most important approaches in order to obtain reliable and useful mathematical models of complex systems. Basically there are two ways to approach this target: the first one is to apply the well-known physical equations that govern the dynamic behavior of the real system, and use this knowledge to generate the differential equations that

J. Enrique Sierra (✉)
Languages and Computer Systems, University of Burgos, Burgos, Spain
e-mail: jesierra@ubu.es

M. Santos
Computer Science Faculty, Complutense University of Madrid, Madrid, Spain
e-mail: msantos@ucm.es

Á. Herrero et al. (eds.), *10th International Conference on Soft Computing Models in Industrial and Environmental Applications*, Advances in Intelligent Systems and Computing 368, DOI 10.1007/978-3-319-19719-7_29

7 Control de un vehículo cuatrirrotor basado en redes neuronales

BIBLIOGRAPHIC DESCRIPTION

Title

Control de un vehículo cuatrirrotor basado en redes neuronales.

Citation

Sierra, J. E., & Santos, M. (2017). Control de un vehículo cuatrirrotor basado en redes neuronales. Actas de las XXXVIII Jornadas de Automática

Abstract

En este trabajo se propone una estrategia de control neuronal adaptativa para estabilizar un vehículo aéreo no tripulado (UAV). Se estudia la influencia de la masa del cuatrirrotor y de las perturbaciones. Los resultados muestran cómo el aprendizaje online hace más robusto el control, minimizando los efectos de las variaciones en la masa y de las perturbaciones externas en la altura.

References

Zulu, A. et al (2016); Sierra, J.E. et al (2017); Bansal, S. et al (2016); Boudjedir, H. et al (2014); Bakshi, N. A. et al (2016); Santos, M. (2011); Santos, M. et al (2006); Bouabdallah, S. et al (2006); Wise, K. A. et al (2006)

CONTROL DE UN VEHÍCULO CUATRIRROTOR BASADO EN REDES NEURONALES

Jesús Enrique Sierra
Universidad Complutense de Madrid, jesier01@ucm.es

Matilde Santos
Universidad Complutense de Madrid, msantos@ucm.es

Resumen

En este trabajo se propone una estrategia de control neuronal adaptativa para estabilizar un vehículo aéreo no tripulado (UAV). Se estudia la influencia de la masa del cuatrirrotor y de las perturbaciones. Los resultados muestran cómo el aprendizaje online hace más robusto el control, minimizando los efectos de las variaciones en la masa y de las perturbaciones externas en la altura.

Palabras Clave: Control neuronal, aprendizaje online, redes neuronales, vehículos aéreos no tripulados (UAV), cuatrirrotor.

1 INTRODUCCIÓN

En los últimos años han surgido nuevas y valiosas aplicaciones de los vehículos aéreos no tripulados en diferentes sectores como: defensa, seguridad, construcción, agricultura, lucha contra incendios, entretenimiento, mensajería, etc, que demandan el diseño de controladores eficientes y robustos que soporten estas u otras aplicaciones. Por ello el modelado y el control de estos sistemas inestables y complejos siguen motivando la investigación y el interés de la comunidad científica [1].

La modelización y el control de un vehículo aéreo no tripulado, UAV (*Unmanned Aerial Vehicle*), no es una tarea sencilla. Su complejidad proviene de diferentes fuentes: la aleatoriedad de los flujos de aire y de las fuerzas exógenas, la fuerte no linealidad de la dinámica del mismo, los acoplamientos en su dinámica, la incertidumbre de las medidas,.... Estos factores convierten a las técnicas basadas en la inteligencia artificial en un camino prometedor para la identificación y el control de estos sistemas.

Estas técnicas resultan especialmente útiles cuando se producen modificaciones en parámetros del modelo mientras el sistema está en funcionamiento. Por ejemplo, variaciones en la masa total del sistema en los vehículos que se emplean en tareas logísticas, donde la masa depende en cada momento de la paquetería que están transportando.

Existen otros trabajos que han aplicado redes neuronales al modelado de estos sistemas [2, 3] y también para el control [4, 5]. También se pueden encontrar ejemplos de la aplicación de estas técnicas inteligentes para modelar otros sistemas complejos no lineales [6], por ejemplo, vehículos marinos [7].

En este trabajo se propone una estrategia de control neuronal adaptativa para estabilizar un vehículo aéreo no tripulado (UAV). Se estudia la influencia de la masa del cuatrirrotor y de las perturbaciones. Los resultados muestran cómo el aprendizaje online hace más robusto el control, minimizando los efectos de las variaciones en la masa y de las perturbaciones externas en altura.

Figura 1: Vehículo cuatrirrotor.

La organización del documento es la siguiente. En la sección 2 se exponen las ecuaciones que describen el comportamiento dinámico del UAV. La sección 3 se destina al estudio de la estrategia de control neuronal adaptativa implementada. Los resultados son discutidos y comparados en la sección 4. El documento finaliza con las conclusiones y trabajos futuros.

2 DESCRIPCIÓN DEL SISTEMA

Un vehículo cuatrirrotor consta de cuatro brazos perpendiculares, cada uno de ellos con un motor y un rotor (figura 1). Los cuatro motores proporcionan

propulsión para el control de la sustentación así como el control de la dirección. El sistema consiste en dos pares de rotores opuestos entre sí: (1,3) y (2,4) (Figura 2). Para que esté balanceado un par gira hacia la derecha mientras que el otro gira hacia la izquierda. Un incremento de la velocidad del rotor 3 frente al rotor 1 produce un pitch positivo ($\theta > 0$), mientras que un incremento de velocidad del rotor 4 frente al 2, produce un roll positivo ($\phi > 0$) (Figura 2).

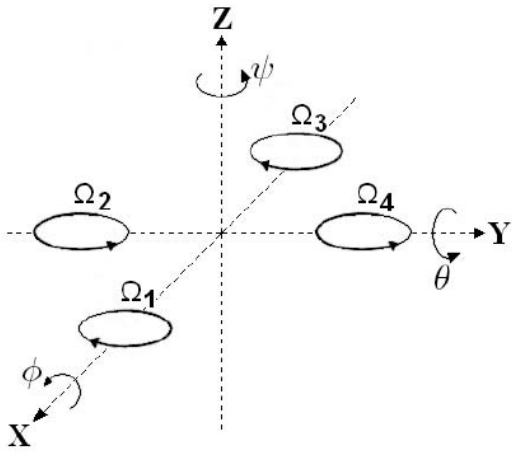

Figura 2: Sistema de coordenadas del UAV.

La posición absoluta es descrita por tres coordenadas (x, y, z) y su orientación por los tres ángulos de Euler (ϕ, θ, ψ), bajo las condiciones $(-\pi \leq \psi < \pi)$ para el yaw, $\left(\frac{\pi}{2} \leq \phi < \frac{\pi}{2}\right)$ para el roll y $(\frac{\pi}{2} \leq \theta < \frac{\pi}{2})$ para el pitch.

Usando el método de Newton-Euler la dinámica angular del sistema puede expresarse como:

$$\tau = J\dot{\omega} + \omega \times J\omega \quad (1)$$

$$J = \begin{pmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{pmatrix} \quad (2)$$

Donde τ es un vector de pares a lo largo de los tres ejes, J es el tensor de inercia, ω es un vector de velocidades angulares y × denota el producto vectorial.

La dinámica de traslación se rige por la ecuación:

$$m\dot{v} = RT - mge_3 \quad (3)$$

Donde m es la masa del cuatrirotor, R es la matriz de rotación, g es la aceleración de la gravedad, T es el vector de fuerzas y $e_3 = [0,0,1]^T$ es un vector unitario que describe la orientación del rotor.

Los vectores τ y T, vienen dados por las velocidades de giro de los rotores:

$$\tau = \begin{pmatrix} bl(\Omega_4^2 - \Omega_2^2) \\ bl(\Omega_3^2 - \Omega_1^2) \\ d(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 - \Omega_3^2) \end{pmatrix} \quad (4)$$

$$T = \begin{pmatrix} 0 \\ 0 \\ b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \end{pmatrix} \quad (5)$$

En las ecuaciones 4 y 5, b es el coeficiente de empuje, d es el coeficiente de arrastre y l es la longitud de cada brazo; Ω_1, ...,Ω_4 son las velocidades de los rotores 1 a 4, respectivamente.

Para simplificar, en vez de utilizar las velocidades de los rotores se pueden utilizar las señales de control u_1, u_2, u_3 y u_4, que se obtienen de la transformación:

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -1 & 0 & 1 \\ -1 & 0 & 1 & 0 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} \quad (6)$$

Esta matriz es invertible por lo que a partir de señales de control pueden generarse consignas de velocidad de los rotores.

Usando las ecuaciones 1 a 6, se llega al siguiente sistema de ecuaciones:

$$\ddot{\phi} = \dot{\theta}\dot{\psi}\,(I_y - I_z)/I_x + (lb/I_x)u_2 \quad (7)$$

$$\ddot{\theta} = \dot{\phi}\dot{\psi}\,(I_z - I_x)/I_y + (lb/I_y)u_3 \quad (8)$$

$$\ddot{\psi} = \dot{\phi}\dot{\theta}\,(I_x - I_y)/I_z + (d/I_z)u_4 \quad (9)$$

$$\ddot{X} = -(sin\theta cos\phi)\,(b/m)u_1 \quad (10)$$

$$\ddot{Y} = (sin\phi)\,(b/m)u_1 \quad (11)$$

$$\ddot{Z} = -g + (cos\theta cos\phi)\,(b/m)u_1 \quad (12)$$

Las constantes empleadas para las simulaciones (Tabla 1) se han extraído de [8].

Tabla 1: Constantes empleadas en el modelo.

Parámetro	Descripción	Valor
l	longitud	0.232 m
m	Masa del cuatrirotor	0.52 Kg
d	Coeficiente de arrastre	7.5e-7 N m s^2
b	Coeficiente de empuje	3.13e-5 N s^2
I_x	Inercia en X	6.228e-3 Kg m^2
I_y	Inercia en Y	6.225e-3 Kg m^2
I_z	Inercia en Z	1.121e-2 Kg m^2

3 DESCRIPCIÓN DEL CONTROLADOR NEURONAL

3.1 ESTRATEGIA DE CONTROL

Existen diferentes estrategias a la hora de emplear las redes neuronales para controlar sistemas. En este caso hemos empleado una variante de la estrategia de aprendizaje generalizado (GLA), refinando la red en tiempo real mediante un aprendizaje adaptativo [9].

En una primera fase se aplica la estrategia de aprendizaje generalizado, entrenando la red offline para identificar la inversa de la planta (Figura 3).

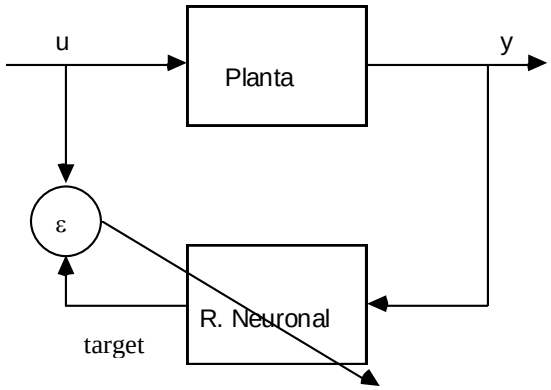

Figura 3: Fase de entrenamiento offline para identificación de la planta.

La red neuronal que sirve de controlador debe actuar como la inversa de la planta, es decir, que a partir de la respuesta deseada (la señal de referencia *r*) calcule la señal de control *u* que conduzca a la salida real del sistema *y* a la señal de referencia.

Una vez que la red se ha entrenado offline, se coloca la red en cascada con la planta. Después se refina la configuración de la red online. Para ello, en cada intervalo de control se ejecutan secuencialmente dos procesos en la red:

1. **Simulación**: A partir de la referencia de entrada se obtiene la salida *u* que se lleva a la planta a la referencia (Figura 4, conmutador en la posición superior).
2. **Aprendizaje**: A partir de las salidas de la planta *y* anteriores se entrena la red neuronal para generar la salida de control *u*. (Figura 4, conmutador en la posición inferior).

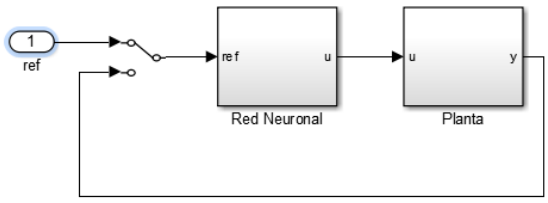

Figura 4: Fase de control + aprendizaje online.

3.2 CONTROL DE ALTITUD

Para comprobar la validez de la técnica nos hemos centrado en la regulación de la altitud. Los cuatrirotores normalmente vienen equipados con acelerómetros, por lo que se supone medible la aceleración en el eje Z ($\ddot{Z}$).

En este caso, la red debe ser capaz de simular la señal de control u_1 a partir de las medidas de aceleración. El conjunto de datos de entrada de la red se forma, en cada instante de tiempo t, con el valor de la señal de control u_1 en los 10 instantes de muestreo anteriores, $u_1(t - iTs)$ donde i = 1…10, y Ts es tiempo de muestreo, y el valor de la aceleración del eje Z en el instante actual $\ddot{Z}$(t). El conjunto de datos objetivo se ha generado con el valor de la señal de control u_1 en el instante actual u_1(t). El Ts empleado en los experimentos ha sido 10ms.

Para controlar la altura Z se añade un controlador PID que genera las consignas necesarias de aceleración en el eje Z. La figura 5 refleja el sistema de control completo propuesto.

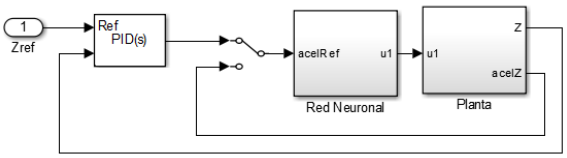

Figura 5: Controlador de altitud propuesto.

Para entrenar la red durante la fase offline se generan trenes de pulsos de amplitud variable durante 4 segundos. Previamente al entrenamiento los conjuntos de entrada y salida son normalizados para ajustar el rango a 0-1. La aplicación de esta técnica en un sistema real, requeriría la supervisión de los pulsos por un piloto, quién se encargaría de ajustar las amplitudes para no deteriorar el sistema.

En el estudio se ha empleado como red neuronal un perceptrón multicapa (MLP) con 5 capas ocultas (Figura 6). El algoritmo de optimización empleado ha sido el Levenberg-Marquardt con $\mu = 0.001$.

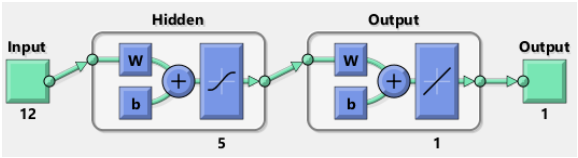

Fig. 6. Esquema de la red neuronal empleada.

En la figura 7 puede apreciarse cómo la salida de la red se aproxima de una manera bastante fiel a la dinámica inversa de la planta tras el entrenamiento offline.

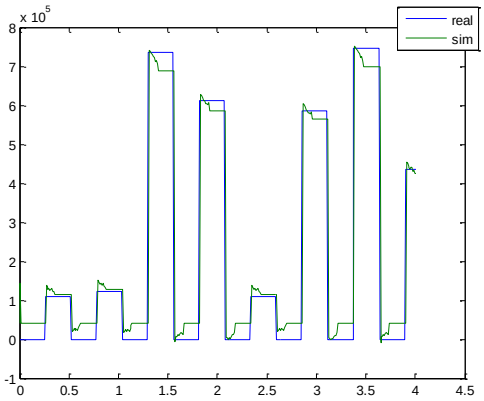

Figura 7: Entrenamiento offline de la red

4 RESULTADOS

Los resultados que se presentan se han obtenido mediante simulación empleando el software Matlab/Simulink. Las simulaciones duran 15 segundos; durante los 4 primeros se entrena el controlador offline y en los restantes 11 segundos se aplica el aprendizaje, introduciendo al controlador como entrada la consigna Zref.

En la figura 8 puede observarse el comportamiento de las distintas fases del controlador. En el eje x se representa el tiempo y en el eje y se representan: la aceleración en Z del sistema (en azul), la consigna de aceleración generada por el controlador una vez entrenado (en verde), y la altitud (en rojo).

Hasta el segundo 4 el controlador está siendo entrenado. Por este motivo, la gráfica en verde vale 0 ese instante. Las señales de control que se emplean para entrenar al controlador provocan que la altitud del sistema varíe. En el instante t = 4, el controlador deja de ser entrenado y pasa a la fase de control. En ese momento se fija el valor 2 como referencia para la altura Z. Se empiezan a generar consignas de aceleración para estabilizar la altitud en torno a ese valor. En esta figura 8 también es interesante ver que las señales acelRef (entrada de la red neuronal) y acelZ (salida del sistema) se parecen cada vez más conforme pasa el tiempo gracias al aprendizaje online.

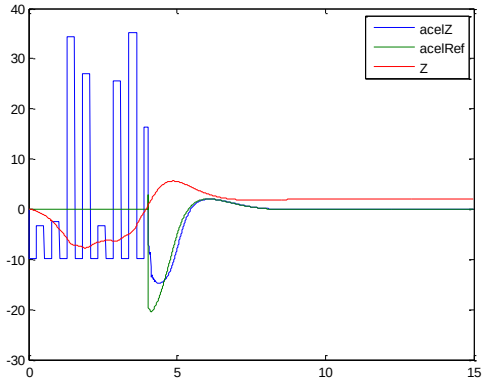

Figura 8: Fases del controlador de altitud propuesto.

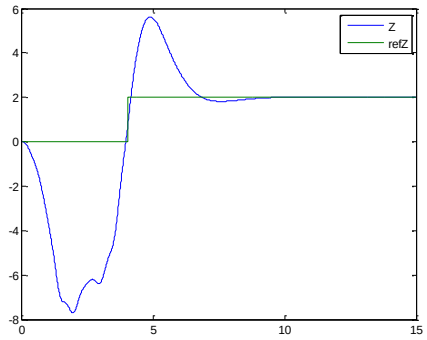

Figura 9: Altitud con aprendizaje online.

En la figura 9 se muestra en azul la altura real del UAV y en verde la referencia. Puede apreciarse como el controlador consigue estabilizar la señal de altitud entorno al valor deseado.

Estos resultado podrían extenderse al control de los tres ángulos de Euler del UAV, para el seguimiento de una trayectoria.

4.1 VARIACIÓN DE LA MASA DEL SISTEMA

Se simula ahora el sistema introduciendo un cambio en la masa (se duplica) del cuatrirotor en el instante 6. Puede observarse en las figuras 10 y 11 cómo el sistema con aprendizaje online consigue aproximar la referencia (verde) mientras que en el sistema sin aprendizaje el error estacionario es notable.

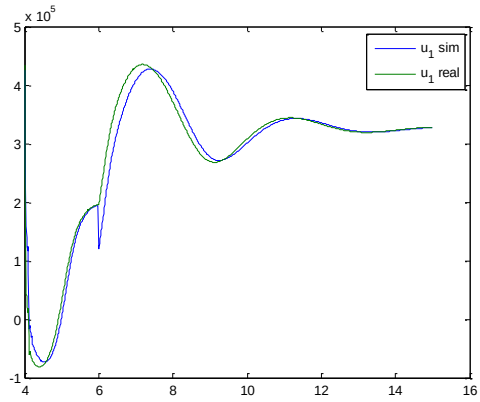

Fig.10 . Simulación de la señal de control u1 con aprendizaje online y variación de la masa.

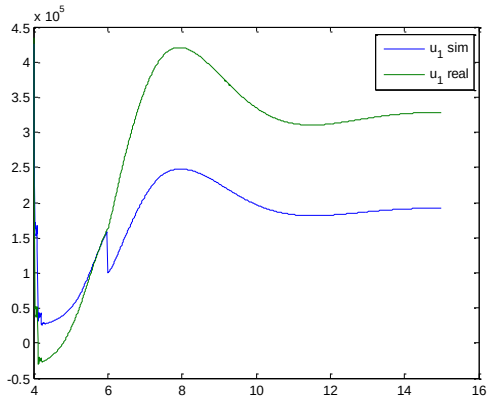

Fig. 11. Simulación de la señal de control u1 sin aprendizaje y variación en masa.

La figura 12 representa la referencia de altitud (en verde) y las señales de altitud con y sin aprendizaje online (azul y rojo respectivamente). El error en la salida de la red neuronal mostrado en la figura 11 se traduce en una peor respuesta a un escalón en la altura. En este caso aumenta el sobreimpulso y el error estacionario es grande.

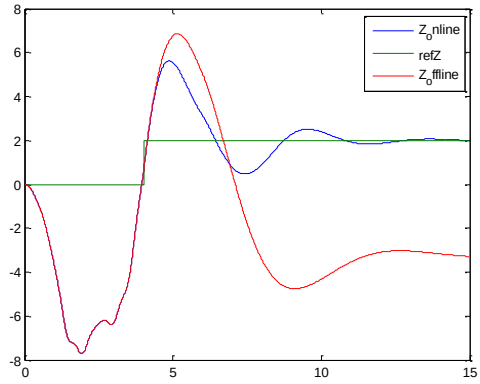

Fig. 12. Comparación del control de altitud con y sin aprendizaje online ante una perturbación.

4.2 INFLUENCIA DE PERTURBACIONES EXTERNAS

El efecto de una perturbación en la ecuación 12 se ha introducido como:

$$\ddot{Z} = -pert - g + (cos\theta cos\phi)\,(b/m)u_1 \qquad (13)$$

En esta simulación se aplica una perturbación de 9.8 m/s^2 en el instante 6. Puede observarse en las figuras 13 y 14 cómo el sistema con aprendizaje online consigue aproximar la referencia (en verde) sin error apreciable mientras que en el sistema sin aprendizaje adaptativo produce error estacionario.

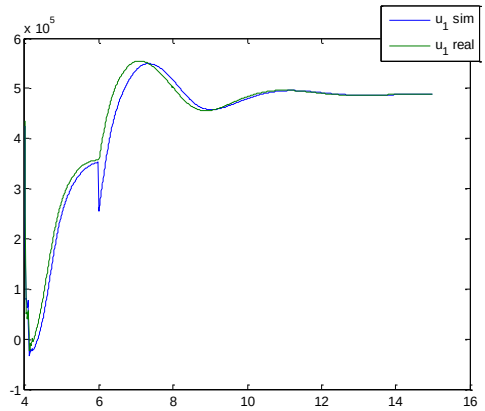

Fig. 13. Simulación de la señal de control u1 con aprendizaje online y perturbación externa.

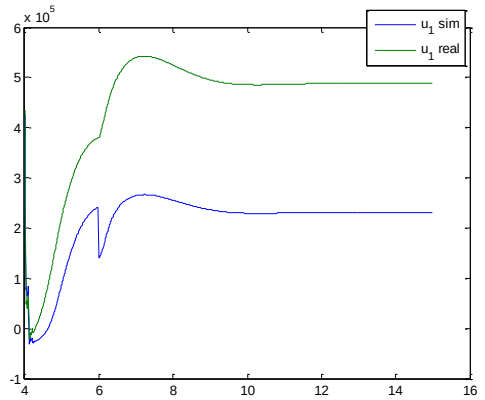

Fig. 14. Simulación de la señal de control u1 sin aprendizaje online y perturbación externa.

La figura 15 representa la referencia de altitud (en verde) y la altura del UAV con y sin aprendizaje online (azul y rojo, respectivamente). El error en la salida de la red neuronal observable en la figura 14 se traduce en una peor respuesta a un escalón. En este caso aumenta el sobreimpulso y el error estacionario es mayor que en el caso de variación de la masa (Figura 12).

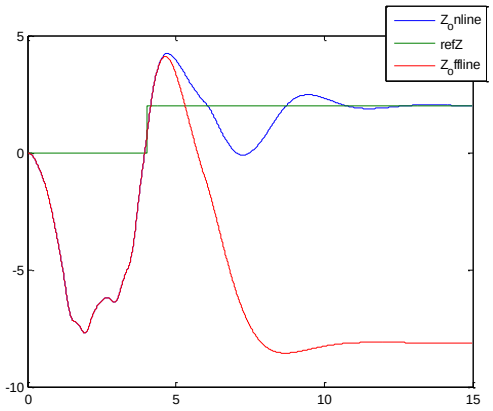

Fig. 15. Comparación del control de altitud con y sin aprendizaje online ante una perturbación.

5 CONCLUSIONES

Los UAV son sistemas complejos de modelar y controlar. La dificultad estriba en la aleatoriedad de las perturbaciones y en su dinámica fuertemente no lineal.

En este trabajo se ha mostrado cómo la aplicación de una técnica de la Inteligencia Artificial, en concreto las redes de neuronas artificiales, se pueden aplicar para diseñar un sistema de control neuronal que sea eficiente.

Se ha validado la propuesta en simulación para el control de la altura de un UAV. Se ha mostrado cómo el aprendizaje adaptativo de la red (online) permite un control capaz de hacer frente a variaciones en la masa del sistema o a perturbaciones externas constantes en la aceleración.

Como trabajos futuros se plantean, por un lado, controlar el sistema completo para considerar los acoplamientos entre sus variables y, por otro, añadir modelos más realistas de perturbaciones.

Agradecimientos

El autor J.E. Sierra agradece a la empresa ASTI la financiación de la presentación de este trabajo en las Jornadas de Automática 2017.

Referencias

[1] Zulu, A., & John, S. (2016). A review of control algorithms for autonomous quadrotors. arXiv preprint arXiv:1602.02622

[2] Sierra, J.E. & Santos, M. (2017). Modeling engineering systems using analytical and neural techniques: hybridization. Neurocomputing,.

[3] Bansal, S., Akametalu, A. K., Jiang, F. J., Laine, F., & Tomlin, C. J. (2016). Learning quadrotor dynamics using neural network for flight control. In: Decision and Control (CDC), 2016 IEEE 55th Conf. on (pp. 4653-4660).

[4] Boudjedir, H., Bouhali, O., & Rizoug, N. (2014). Adaptive neural network control based on neural observer for quadrotor unmanned aerial vehicle. Advanced Robotics, 28(17), 1151-1164.

[5] Bakshi, N. A., & Ramachandran, R. (2016). Indirect model reference adaptive control of quadrotor UAVs using neural networks. In: Intelligent Systems and Control (ISCO), 2016 10th Int. Conf. on (pp. 1-6).

[6] Santos, M. (2011). Aplicaciones Exitosas de Control Inteligente a Casos Reales. Science Direct, Revista Iberoamericana de Automática e Informática Industrial.

[7] Santos, M., López, R., & De La Cruz, J. M. (2006). A neuro-fuzzy approach to fast ferry vertical motion modelling. Engineering Applications of Artificial Intelligence, 19(3), 313-321.

[8] S. Bouabdallah, S. and Siegwart, R. (2006) Advances in Telerobotics, chapter Towards Intelligent Miniature Flying Robots, pp. 429–440.

[9] Wise, K. A., Lavretsky, E., & Hovakimyan, N. (2006, June). Adaptive control of flight: theory, applications, and open problems. In American Control Conference, 2006 (pp. 6-pp). IEEE.

8 Disturbances Based Adaptive Neuro-Control for UAVs: A First Approach

BIBLIOGRAPHIC DESCRIPTION

Title

Disturbances Based Adaptive Neuro-Control for UAVs: A First Approach.

Citation

Sierra, J. E., & Santos, M. (2018, June). Disturbances Based Adaptive Neuro-Control for UAVs: A First Approach. In The 13th International Conference on Soft Computing Models in Industrial and Environmental Applications (pp. 293-302). Springer, Cham.

Abstract

Abstract. In this work an adaptive neuro-control is proposed to cope with some external disturbances that can affect unmanned aerial vehicles (UAV) dynamics, specifically: the variation of the system mass during logistic tasks and the influence of the wind. An intelligent control strategy based on a feedforward neural networks is applied. In particular, a variant of the generalized learning algorithm has been used. Simulation results show how the on-line learning increases the robustness of the controller, reducing the effects of the changes in mass and the effects of wind on the UAV stabilization, thus improving the system response. It has been compared with a PID controller obtaining better results.

References

Ortega, J.J.et al (2016); Fahmy, A. A. et al (2015); Garcia-Auñón, P. et al (2017); Sierra. J.E. et al (2018); Bansal, S. et al (2016);. Boudjedir, H. et al (2014);. Bakshi, N.A. et al (2016);. Sierra, J.E. et al (2017); Yañez-Badillo H. et al;.(2017); Wise, K. A. et al (2006)

Disturbances Based Adaptive Neuro-Control for UAVs: A First Approach

J. Enrique Sierra and Matilde Santos(✉)

Computer Science Faculty, Complutense University of Madrid, Madrid, Spain
{jesier01,msantos}@ucm.es

Abstract. In this work an adaptive neuro-control is proposed to cope with some external disturbances that can affect unmanned aerial vehicles (UAV) dynamics, specifically: the variation of the system mass during logistic tasks and the influence of the wind. An intelligent control strategy based on a feedforward neural networks is applied. In particular, a variant of the generalized learning algorithm has been used. Simulation results show how the on-line learning increases the robustness of the controller, reducing the effects of the changes in mass and the effects of wind on the UAV stabilization, thus improving the system response. It has been compared with a PID controller obtaining better results.

Keywords: Neuro-control · Adaptive control · Disturbances rejection
Online learning · Neural networks · Unmanned Aerial Vehicle (UAV)
Quadrotor

1 Introduction

In recent years, new and valuable applications of unmanned aerial vehicles (UAV) have appeared in different sectors such as defense, security, construction, agriculture, entertainment, shipping, etc. These and other applications demand the design of efficient and robust controllers for those autonomous vehicles. Thus, the modeling and control of these complex and unstable systems still motivate the research and the interest of the scientific community [1–3].

Nevertheless, the modeling and control of unmanned aerial vehicles are not easy tasks. Their complexity comes from the randomness of the airstreams and of the exogenous forces, the high non-linearity of the dynamics, the coupling between the internal variables, the uncertainty of the measurements… These factors make the techniques based on artificial intelligence a promising approach for the identification and control of these systems.

These intelligent strategies are especially interesting when the model parameters vary while the system is working. For example, the total mass will undergo variations when the vehicle is performing logistic tasks, since the mass depends on the packages that are shipped.

There are several studies where neural networks are applied to model these systems [4, 5] and to control them [6–9]. But these intelligent strategies are usually applied without considering external disturbances, an issue that deserves further research.

M. Graña et al. (Eds.): SOCO'18-CISIS'18-ICEUTE'18, AISC 771, pp. 293–302, 2019.
https://doi.org/10.1007/978-3-319-94120-2_28