



UNIVERSIDAD POLITÉCNICA DE MADRID
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
AERONÁUTICA Y DEL ESPACIO
GRADO EN INGENIERÍA AEROESPACIAL

TRABAJO FIN DE GRADO
Control adaptativo para aeronaves no tripuladas

AUTOR: Carlos F. Domínguez Alegre

ESPECIALIDAD: Ciencia y Tecnologías Aeroespaciales (CTA)

TUTOR DEL TRABAJO: Prof. Ignacio Gómez Pérez

Julio de 2019

« *Concevoir une machine volante n'est rien;*
» *Fabriquer est peu;*
» *L'essayer est tout* »

— L. Ferdinand Ferber

Índice general

Índice de figuras	5
Índice abreviaturas	6
1. Motivación y objetivos	8
2. Introducción	9
2.1. Estructura del trabajo	10
3. Fundamento teórico	11
3.1. Estimador mínimos cuadrados	11
3.2. Estimador mínimos cuadrados recursivo	12
3.3. SPSA	13
4. Procedimiento	15
4.1. Síntesis previa del controlador	15
4.2. Trimado de la planta	15
4.2.1. Linealización de la planta	17
4.2.2. Síntesis	18
4.3. Modelo de la planta real frente a estimación	20
4.4. Controladores adaptativos	20
4.4.1. Adaptación mediante RLSE	21
4.4.2. Adaptación mediante SPSA	23
5. Resultados, análisis y comparaciones	24
5.1. Adaptación mediante RLSE	24
5.1.1. Adaptación off-line	24
5.1.2. Adaptación on-line	25
5.2. Adaptación mediante SPSA	27
5.3. Comparación de ambos métodos	27
5.4. Trabajo futuro	28
6. Conclusiones	29
A. Modelo físico y másico del UAV	30
A.1. Modelo másico, cálculo de propiedades	30
A.2. Modelo físico	32
B. Modelo DATCOM del UAV	35
B.1. Archivo de salida	36
B.2. Derivadas de estabilidad y control	40
C. Modelo Simulink UAV	41
C.1. Subsitema 'UAV'	41
C.2. Subsitema 'Enviroment'	44
C.3. Subsitema 'Random input'	44
C.4. Subsitema 'workspace'	44

C.5. Visores o scopes	44
C.6. Subsitema 'Controller'	45
D. Archivos Matlab	47
D.1. Linearize.m	47
D.2. LSE.m	48
D.3. TrimUAV.m	48
D.4. uAtTrim.m	50
D.5. randU.m	52
D.6. Jfunction.m	52
D.7. RLSESIM.m	52
D.8. RLSEklm.m	53
D.9. SPSA.m	53
D.10.KlongRLSE.m	55
Bibliografía	56

Índice de figuras

1.1. Previsiones del mercado de UAV's según [9]	8
2.1. Esquema del concepto de control adaptativo proruado por P.E. Wellstead en 'Self-Tuning Systems' [10]	9
3.1. Resumen del método SPSA propuesto en [6]	13
4.1. Esquema interno del controlador empleado para encontrar los puntos de equilibrio del sistema	15
4.2. Timón de profundidad frente a TAS a distintos ángulos de balance	16
4.3. Alerones frente a TAS y ángulo de balance	16
4.4. Motor frente a velocidad vertical y TAS a ϕ cte.	17
4.5. Comparación del control de la velocidad entre controlador lqr y controlador clásico	18
4.6. Comparación controlador lqr y controlado clásico en velocidad vertical	18
4.7. Ángulo de balance en control clásico y controlador lqr	19
4.8. Modificación del modelo Simulink	20
4.9. Estimación de B(4,1) mediante RLSE	21
4.10. Nuevo subsistema añadido a 'Controller'	22
4.11. Bloque 'RLSE & LQR' dentro del nuevo subsistema	22
4.12. Función de coste para diversos valores de Q(2,2) y Q(4,4)	23
5.1. Comparación de ángulo de asiento en controlador sin modificar y el nuevo generado por RLSE	24
5.2. Comparación ángulo de asiento en controlador sin modificar y el controlador con ganancia calculada on-line	25
5.3. Comparación ángulo de asiento en controlador sin modificar y el controlador con ganancia calculada on-line con iguales perturbaciones	25
5.4. Comparación de q en controlador sin modificar y el controlador con ganancia calculada on-line con iguales perturbaciones	26
5.5. Parámetros del modelo de perturbación atmosférica	26
5.6. Comparación de ángulo de asiento en controlador sin modificar y la adaptación SPSA . .	27
B.1. Coeficiente de sustentación total	40
B.2. Polar del avión	40
B.3. Eficiencia aerodinámica	40
B.4. Momento de cabeceo	40
B.5. Potencia de control lateral	40
B.6. Momento de guiñada adversa	40
C.1. Modelo del UAV Simulink	41
C.2. Esquema interno del subsistema UAV	42
C.3. Opciones dentro del bloque DATCOM de Simulink (Aerospace Blockset)	43
C.4. Opciones dentro del bloque '6 DOF'	43
C.5. Subsistema 'enviroment'	44
C.6. Subsistema 'Controller'	45
C.7. Subsistema 'State-Space Controller'	46

Índice de abreviaturas

- **DATCOM:** siglas en inglés 'data compendium', programa desarrollado por el ejército del aire estadounidense.
- **DOF:** siglas en inglés 'degrees of freedom', grados de libertad.
- **LQR:** siglas en inglés 'linear-quadratic regulator', regulador cuadrático lineal.
- **PID:** siglas en inglés 'proportional integral and derivative', normalmente referido a un controlador que posea ganancias contra dichas cualidades de la variable de control.
- **RLSE:** siglas en inglés 'recursive least squares estimator', estimador de mínimos cuadrados recursivo.
- **SPSA:** siglas en inglés 'simultaneous perturbation stochastic approximation', aproximación mediante perturbación simultánea estocástica, referido a algoritmo de aproximación del gradiente.
- **TAS:** siglas en inglés 'true air speed'.
- **UAV:** siglas en inglés 'unmanned aerial vehicle', vehículo aéreo no tripulado.

Nomenclatura

- α : ángulo de ataque.
- β : ángulo de resbalamiento.
- γ : ángulo de asiento de la velocidad.
- θ : ángulo de asiento.
- ϕ : ángulo de balance.
- ψ : rumbo.
- p, q, r : Componentes velocidad angular en ejes cuerpo.
- C_L : Coeficiente de sustentación.
- C_D : Coeficiente de resistencia.
- C_m : Coeficiente momento de cabeceo.
- C_l : Coeficiente momento alabeo.
- C_n : Coeficiente momento guiñada.
- C_N : Coeficiente de fuerza normal.
- C_A : Coeficiente de fuerza axial.
- C_Y : Coeficiente fuerza lateral.

- $C_{L\alpha}$: Derivada coeficiente de sustentación respecto a α .
- $C_{m\alpha}$: Derivada coeficiente de momento de cabeceo respecto a α .
- $C_{Y\beta}$: Derivada coeficiente fuerza lateral respecto a β .
- $C_{n\beta}$: Derivada coeficiente momento de guiñada respecto a β (índice de estabilidad estática direccional con mandos fijos).
- $C_{l\beta}$: Derivada del coeficiente de momento de alabeo respecto a β (efecto diedro).
- $C_{l\delta_a}$: potencia control lateral.
- $C_{n\delta_a}$: guiñada adversa.
- $C_{l\delta_r}$: derivada momento alabeo respecto a la deflexión del timón de profundidad.
- $C_{n\delta_r}$: potencia control direccional

Capítulo 1

Motivación y objetivos

En los últimos años el mercado de aeronaves no tripuladas ha emergido con gran fuerza en el sector aeronáutico. Cada año esta parte del sector crece de manera extraordinaria con pequeñas aeronaves de la mano de empresas privadas como puede ser Amazon o Google. Como de aeronaves mayores, normalmente gracias a la inversión de los ejércitos. Los estudios de mercado indican una previsión de crecimiento continua. Sin embargo, aún siendo un sector en auge, existen diversas trabas que impiden que el mercado despegue, una de ellas y probablemente la más importante es la suregulación legal. Actualmente la integración de los vehículos aéreos no tripulados en espacio aéreo no segregado es un futuro deseable pero lejano. Un cambio en dichas leyes es esperable a corto o medio plazo pero para ello los sistemas aéreos no tripulados han de demostrar niveles de aeronavegabilidad iguales o mayores que las aeronaves convencionales.

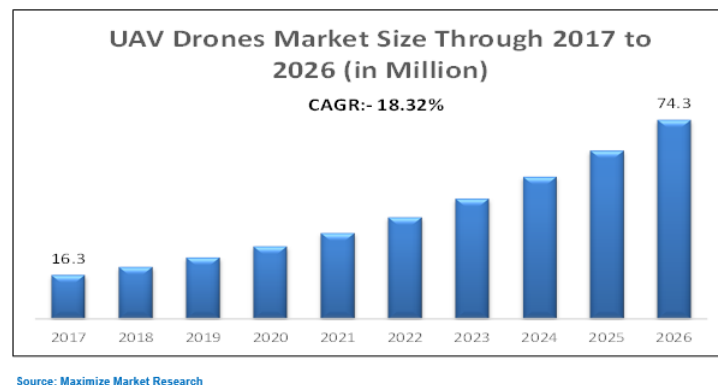


Figura 1.1: Previsiones del mercado de UAV's según [9]

Uno de los aspectos importantes a la hora de conseguir niveles de seguridad operacionales de las aeronaves no tripuladas es la creación de leyes de control robustas, con capacidad para adaptarse a nuevas situaciones e imprevistos durante el vuelo. Este trabajo busca estudiar métodos de adaptación para dichas leyes de control y por tanto aumentar la seguridad de operación de dichas aeronaves.

Los objetivos de este trabajo son:

- Estudio general de mecanismos de adaptación para las leyes de control.
- Modelizado de un UAV y generación de un simulador para sobre él probar las diferentes técnicas de control adaptativo.
- Estudio particular de dos algoritmos de adaptación, uno basado en la estimación de la planta y otro basado en el ajuste de los pesos del controlador.
- Comparación de ambos métodos de adaptación y crítica a sus ventajas e inconvenientes de aplicación a la planta propuesta.

Capítulo 2

Introducción

Tal y como enuncia el título del presente trabajo 'Control adaptativo para aeronaves no tripuladas', sería conveniente comenzar con una pequeña definición de lo que será el área de estudio, es decir, el 'control adaptativo'. Nos referimos a control adaptativo a aquellas técnicas de control de sistemas que tienen en cuenta la variación que pueden experimentar las características de la planta a controlar, o en su defecto, que dichas características sean *a priori* desconocidas o tengan cierto grado de incertidumbre. En esta definición pueden entrar numerosas técnicas de fundamentos muy dispares, como pueden ser la programación de ganancias (por ejemplo para diversas condiciones de trabajo de la planta), o ganancias de realimentación del sistema altas (de forma que las variaciones en la planta o las discrepancias entre la estimación de esta con la realidad tendrían menor influencia en el desempeño final); sin embargo este estudio se centrará en dos técnicas basadas en el auto-ajuste de parámetros.

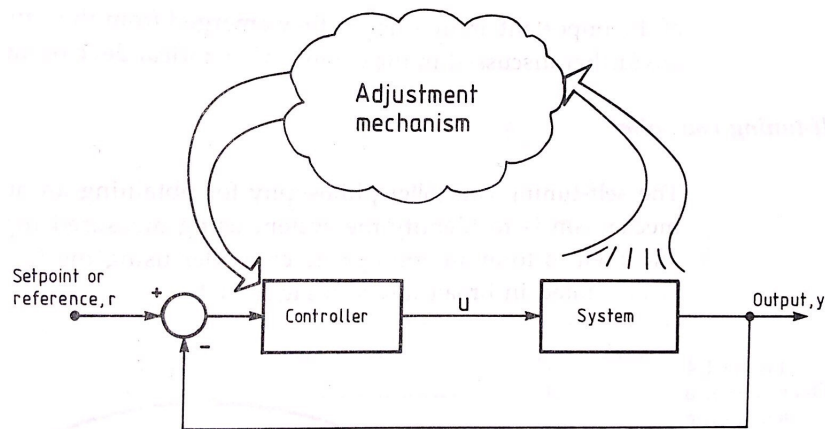


Figura 2.1: Esquema del concepto de control adaptativo propuesto por P.E. Wellstead en 'Self-Tuning Systems' [10]

Las técnicas de auto ajuste de parámetros se basan en, dada una arquitectura de controlador, encontrar los parámetros que consiguen la mejor respuesta del sistema, o en su defecto una respuesta mejor a la que tendrían otras técnicas de control no adaptativo. El concepto 'una respuesta mejor' se basa normalmente en un criterio definido *a priori*, existiendo dos grandes aproximaciones a la hora de seleccionar el criterio de adaptación:

- Controladores con modelo de referencia: en este tipo de esquemas se busca que la respuesta del sistema controlado se parezca a un modelo de referencia dado.
- Controladores basados en la identificación del modelo: en estos esquemas el controlador se ajusta mejorando el conocimiento sobre la planta.

Existen también numerosas aproximaciones con distintos fundamentos a los expuestos, como pueden ser controladores de aprendizaje iterativo donde se puede englobar uno de los métodos que van a ser

sometidos a estudio en este trabajo. La filosofía de este tipo de mecanismo de ajuste consiste en la repetición de un experimento y la minimización de un criterio dado.

En este trabajo se desarrollan dos técnicas de control adaptativo, ambas desde diferentes puntos de partida en el método de adaptación. Primero se realiza una adaptación vía estimación recursiva de la planta, es decir, mejorando el modelo de la planta a controlar a partir de una estimación inicial. Después se estudia un mecanismo de adaptación basado en el ajuste de pesos empleados para sintetizar un controlador LQR partiendo de la estimación primera de la planta y sin modificar dicha estimación.

2.1. Estructura del trabajo

Para comprender el trabajo en su conjunto se recomienda al lector comenzar por los apéndices A, B y C, ya que en ellos se muestra la modelización previa del UAV sobre el cual aplicaremos las técnicas de control adaptativo. Primero se realiza una descripción física y másica de la aeronave sobre la cual trabajaremos. A continuación se estiman las derivadas de estabilidad y control, es decir la aerodinámica de la aeronave. Por último se desarrolla un simulador en Simulink de la dinámica de vuelo del aparato en cuestión. Este ha sido el orden cronológico, como es lógico, seguido a la hora de realizar el trabajo.

Una vez se dispone de las herramientas necesarias se procede al estudio de los controladores adaptativos, primero se realiza una somera argumentación teórica sobre los conceptos básicos de los dos esquemas de adaptación escogidos. Después se calcula unas ganancias para un controlador LQR y sobre este conjunto de sistema-controlador se modifica el sistema y se evalúa el comportamiento de las dos aproximaciones al control adaptativo propuestas.

Por último y como no podía ser de otra manera se evalúa el comportamiento de los métodos de control adaptativos implementados. Se intentará mostrar las ventajas y desventajas de cada uno de ellos así como posibles líneas de estudio futuro.

Todos los modelos de Simulink empleados así como el código de los programas de Matlab pueden encontrarse en el repositorio del trabajo, con acceso mediante este [link](#).

Capítulo 3

Fundamento teórico

3.1. Estimador mínimos cuadrados

Uno de los algoritmos más empleados en la identificación de sistemas es el método de mínimos cuadrados. Esta técnica devuelve el modelo del sistema que mejor se ajusta a los datos pasados que le introducimos minimizando la suma de residuos. Para poder aplicar esta técnica, será necesario llegar a un modelo de nuestro sistema lineal, cuyo procedimiento es el siguiente:

Suponiendo que se tiene un sistema cuya dinámica sea función de un vector de estado, un vector de control y perturbaciones al sistema tal que:

$$\frac{d\mathbf{x}}{dt} = f(\mathbf{x}, \mathbf{u}, \nu) \quad (3.1)$$

Entorno a un punto $\mathbf{x}_0, \mathbf{u}_0$ se puede linealizar el sistema sin más que realizar un desarrollo de Taylor entorno al punto de trabajo.

$$\frac{d\mathbf{x}}{dt} = \mathbf{A} \cdot \mathbf{x} + \mathbf{B} \cdot \mathbf{u} + \nu + o[(\mathbf{x} - \mathbf{x}_0)^2] \quad (3.2)$$

Esta aproximación al sistema será válida siempre que no existan desviaciones apreciables del punto de trabajo. A partir de la ecuación 3.2, discretizándola y reteniendo exclusivamente los dos primeros términos del segundo miembro de la ecuación se obtiene:

$$\begin{aligned} \mathbf{x}^{k+1} &= \mathbf{A} \cdot \mathbf{x}^k + \mathbf{B} \cdot \mathbf{u}^k \\ \mathbf{y}^k &= \mathbf{C} \cdot \mathbf{x}^k \end{aligned} \quad (3.3)$$

Suponiendo a partir de ahora que tenemos acceso a todo el vector de estado, por lo tanto la matriz \mathbf{C} será $\mathbf{C} = \mathbf{I}_{n \times n}$. Así mismo, para continuar con el desarrollo se supone que el sistema es observable.

Para poder aplicar el método directamente, es necesario reordenar la ecuación 3.3 para expresarla como una matriz por un vector. Este paso se denomina 'extender' el método de mínimos cuadrados (ver 'Self tuning systems' cap 3.6 [10]).

$$\mathbf{y}^k = \begin{bmatrix} \mathbf{A} \\ \mathbf{B} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{u} \end{bmatrix} \quad (3.4)$$

Se tomará por simplicidad de notación \mathbf{A} como la matriz compuesta por \mathbf{A} y \mathbf{B} y \mathbf{x} como el vector formado por \mathbf{x} y \mathbf{u} . Con esta nueva notación, recogemos datos en los primeros k pasos del estimador obteniendo el sistema:

$$\begin{aligned} [\mathbf{y}^1, \mathbf{y}^2, \dots, \mathbf{y}^k] &= \mathbf{A} [\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^k] \\ \mathbf{Y}^k &= \mathbf{A} \cdot \mathbf{X}^k \end{aligned} \quad (3.5)$$

Tras operar oportunamente y llamando \mathbf{Y} a la composición de los primeros \mathbf{y}^k vectores y \mathbf{X} a la composición de los primeros \mathbf{x}^k vectores obtenemos el estimador de \mathbf{A} , es decir $\hat{\mathbf{A}}$:

$$\hat{\mathbf{A}} = \mathbf{Y}\mathbf{X}^T [\mathbf{X}\mathbf{X}^T]^{-1} \quad (3.6)$$

Es fácil comprobar que $k \geq n \times m + n \times p$, es decir el número de incógnitas ha de ser menor que el de ecuaciones disponibles.

3.2. Estimador mínimos cuadrados recursivo

Para que este método pueda ser útil en un controlador es necesario poder obtener nuevas estimaciones de $\hat{\mathbf{A}}$ recursivamente a medida que nuevos datos están disponibles. Para ello se definen las siguientes matrices:

$$\begin{cases} \mathbf{B} = \mathbf{Y}\mathbf{X}^T \\ \mathbf{P} = [\mathbf{X}\mathbf{X}^T]^{-1} \end{cases} \quad (3.7)$$

No se debe confundir la matriz \mathbf{B} de la ecuación 3.7 con la matriz \mathbf{B} de las ecuaciones anteriores 3.2, 3.3 y 3.4. Es fácil comprobar que podemos calcular la matriz \mathbf{B} tal como se indica en 3.8.

$$\mathbf{B}^{k+1} = \mathbf{B}^k + [\mathbf{y}^{k+1}] \cdot [\mathbf{x}^{k+1}]^T \quad (3.8)$$

$$[\mathbf{P}^{k+1}]^{-1} = [\mathbf{P}^k]^{-1} + [\mathbf{x}^{k+1}] [\mathbf{x}^{k+1}]^T \quad (3.9)$$

Con el fin de actualizar la matriz \mathbf{P}^{k+1} sin realizar la operación de inversión se hace uso del lemma de inversión de matrices o identidad de **Woodbury**.

$$(\mathbf{A} + \mathbf{BCD})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}(\mathbf{C}^{-1} + \mathbf{DA}^{-1}\mathbf{B})^{-1}\mathbf{DA}^{-1} \quad (3.10)$$

Se recuerda que no se deben confundir las matrices \mathbf{A} y \mathbf{B} con sus matrices homónimas del apartado anterior. En nuestro caso, identificando términos en la ecuación 3.9:

- $\mathbf{A} = \mathbf{P}^{-1}$
- $\mathbf{C} = \mathbf{I}_{m \times m}$
- $\mathbf{B} = [\mathbf{x}^{k+1}]$
- $\mathbf{D} = [\mathbf{x}^{k+1}]^T$

Se calcula \mathbf{P} de la siguiente forma:

$$\mathbf{P}^{k+1} = \mathbf{P}^k \left[\mathbf{I} - \frac{[\mathbf{x}^{k+1}][\mathbf{x}^{k+1}]^T \mathbf{P}^k}{1 + [\mathbf{x}^{k+1}]^T \mathbf{P}^k [\mathbf{x}^{k+1}]} \right] \quad (3.11)$$

Como se puede observar en 3.11, ahora no se tendrá que invertir una matriz sino un término escalar ($1 + [\mathbf{x}^{k+1}]^T \mathbf{P}^k [\mathbf{x}^{k+1}]$). Ahora que se dispone de \mathbf{P}^{k+1} y de \mathbf{B}^{k+1} se puede, asociando los términos de 3.6 y de 3.8 calcular la nueva estimación tal y como se indica en 3.12.

$$\hat{\mathbf{A}}^{k+1} = \mathbf{B}^{k+1} \mathbf{P}^{k+1} \quad (3.12)$$

Esta solución es válida, sin embargo no es muy cómoda a la hora de trabajar con ella, por lo que recuperando las ecuaciones 3.3, 3.8 y 3.9 y operando oportunamente llegamos a:

$$\hat{\mathbf{A}}^{k+1} = \hat{\mathbf{A}}^k + \mathbf{K} \cdot (\mathbf{Y}^k - \hat{\mathbf{A}}^k \cdot \mathbf{X}^k) \quad (3.13)$$

Siendo la ganancia \mathbf{K} :

$$\mathbf{K} = \frac{\mathbf{P}^k [\mathbf{x}^k]}{1 + [\mathbf{x}^k]^T \mathbf{P}^k [\mathbf{x}^k]} \quad (3.14)$$

Ahora combinando 3.11 y 3.14 obtenemos:

$$\mathbf{P}^{k+1} = (\mathbf{I} - \mathbf{K}[\mathbf{x}^k]^T) \mathbf{P}^k \quad (3.15)$$

Un desarrollo matemático más riguroso sobre el empleo del método de mínimos cuadrados aplicado al espacio de los estados puede encontrarse en la publicación 'State-space recursive least-squares' [8]. Son numerosas también las publicaciones que emplean dicho método aplicado a técnicas de control como puede ser 'Online Recursive Closed-Loop State Space Model Identification for Damping Control' [11].

3.3. SPSA

El segundo método para generar un controlador adaptativo se basa en el algoritmo de optimización SPSA (Simultaneous Perturbation Stochastic Approximation). Este es un algoritmo de descenso basado en el gradiente, cuya peculiaridad es necesitar solamente dos medidas de la función objetivo para aproximar el gradiente, independientemente de la dimensión del problema. La demostración del método se puede encontrar en 'Spall'[5]. Además J. C. Spall ha creado una página donde se pueden encontrar casos de aplicación práctica, código Matlab, artículos relacionados etc. [4].

Algorithm 1 Self-tuning LQR.

```

1: initialize  $\theta^0$  such that  $\bar{Q}(\theta^0) = Q$ ,  $\bar{R}(\theta^0) = R$ 
2: for  $i = 0$  to  $i_{\max} - 1$  do ▷  $i_{\max}$  SPSA iterations
3:    $a^i \leftarrow a/(i+1+\bar{a})^\alpha$ 
4:    $c^i \leftarrow c/(i+1)^\gamma$ 
5:   for  $j = 1$  to  $n_g$  do ▷  $n_g$  gradient computations
6:     draw  $\Delta$  from symmetric  $\pm 1$  Bernoulli distribution
7:      $\theta^+ \leftarrow \theta^i + c^i \Delta$ 
8:      $\hat{J}(\theta^+) \leftarrow \text{CostEvaluation}(\theta^+)$ 
9:      $\theta^- \leftarrow \theta^i - c^i \Delta$ 
10:     $\hat{J}(\theta^-) \leftarrow \text{CostEvaluation}(\theta^-)$ 
11:     $g^{i,j}(\theta^i) \leftarrow \text{Equation (14)}$  ▷ gradient computation
12:  end for
13:   $g^i(\theta^i) \leftarrow \text{Average}(g^{i,1}(\theta^i), \dots, g^{i,n_g}(\theta^i))$ 
14:   $\theta^{i+1} \leftarrow \theta^i - a^i g^i(\theta^i)$ 
15: end for
16: LQR design:  $\bar{F}^{\text{final}} \leftarrow \text{lqr}(\bar{A}, \bar{B}, \bar{Q}(\theta^{i_{\max}}), \bar{R}(\theta^{i_{\max}}))$ 
17: return  $\bar{F}^{\text{final}}$ 

Function CostEvaluation( $\theta$ )
18: LQR design:  $\bar{F} \leftarrow \text{lqr}(\bar{A}, \bar{B}, \bar{Q}(\theta), \bar{R}(\theta))$ 
19: update state feedback law (2) with  $F = \bar{F}$ 
20: perform experiment and record  $\{y_k\}$ ,  $\{u_k\}$ 
21: return  $(\sum_{k=0}^{K-1} y_k^T Q y_k + u_k^T R u_k)/K$ 

```

Figura 3.1: Resumen del método SPSA propuesto en [6]

Se seguirá una implementación del método análoga a la propuesta en el artículo 'A Self-Tuning LQR Approach Demonstrated on an Inverted Pendulum' [6]. A continuación se darán un esquema de funcionamiento del método.

$$J = \sum_{k=0}^{K-1} x_k' \cdot Q \cdot x_k + u_k' \cdot R \cdot u_k \quad (3.16)$$

La función objetivo a minimizar está definida por la ecuación 4.8, evaluada en un experimento (definido por un tiempo y condiciones iniciales en nuestro caso) y con las matrices Q y R definidas en un primer momento que serán invariantes. Para ello se supone constante nuestro modelo de la planta (\bar{A}, \bar{B}) y variaremos los pesos del controlador LQR. Se parametrizan las matrices \bar{Q} y \bar{R} de tal forma que ambas matrices se puedan calcular a través del vector de parámetros θ (dichas matrices servirán para sintetizar el controlador en cada paso del algoritmo).

$$\begin{aligned} \bar{Q} &= \bar{Q}(\theta) \\ \bar{R} &= \bar{R}(\theta) \\ \theta &\in \Theta \subseteq \mathbb{R}^p \end{aligned} \quad (3.17)$$

Se buscará por tanto el vector de parámetros que minimice la función objetivo.

$$\min_{\theta \in \Theta} J(\theta) \quad (3.18)$$

Para ello, se calcula el nuevo vector de parámetros en cada iteración como indica la ecuación 3.19.

$$\theta^{i+1} = \theta^i - a^i g^i(\theta^i) \quad (3.19)$$

Donde el gradiente g se ha calculado como el límite incremental definido por 3.20. Para mejorar la estimación del gradiente, este se puede calcular también como la media de los gradientes de n_g experimentos.

$$g^i(\theta^i) = \frac{J(\theta^i + c^i \Delta^i) - J(\theta^i - c^i \Delta^i)}{2 \cdot c^i \Delta^i} \quad (3.20)$$

Donde el vector Δ es de dimensión p y representa una distribución simétrica ± 1 de Bernoulli [2]. Los coeficientes a^i y c^i se calculan a partir de los parámetros del algoritmo a_1, a_2, c_1 y γ tal y como se indica en

$$\begin{aligned} a^i &= \frac{a_1}{i+1+a_2} \\ c^i &= \frac{c_1}{(i+1)^\gamma} \end{aligned} \quad (3.21)$$

En cada iteración se ha de comprobar que el vector de parámetros $\theta^i \in \Theta$ para que las matrices \overline{Q} y \overline{R} sean definidas positivas. En la figura 3.1 se muestra un resumen del método propuesto en el artículo anterior mente citado [6].

Capítulo 4

Procedimiento

4.1. Síntesis previa del controlador

4.2. Trimado de la planta

Para encontrar los puntos de equilibrio para diferentes condiciones de vuelo tenemos dos opciones posibles: la primera consiste en aislar el modelo de fuerzas y momentos de la simulación y encontrar un punto donde estos se anulen empleando alguna herramienta de minimización de Matlab. Sin embargo esta aproximación requiere de tener un modelo disponible y como tal, está expuesto a errores de modelización, por tanto, se ha optado por una aproximación al problema menos técnica y más práctica, encontrando los puntos de equilibrio en la propia simulación. Esto se consigue con un sencillo controlador PID con un rendimiento pobre, pero que nos asegura encontrar el estado de equilibrio.

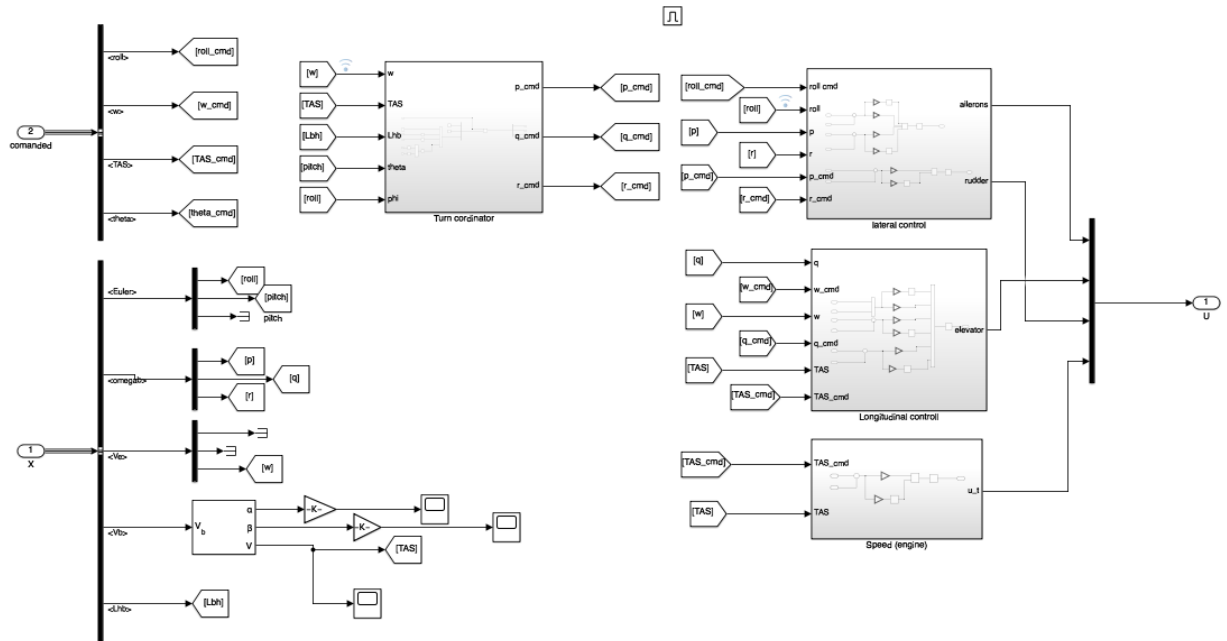


Figura 4.1: Esquema interno del controlador empleado para encontrar los puntos de equilibrio del sistema

Se ha escogido esta forma de encontrar los puntos de equilibrio de la aeronave para simular, lo que serían en un ensayo en vuelo con la aeronave, el control del piloto. Se encontrarían por tanto los puntos de equilibrio a partir del vuelo manual de la aeronave, permitiendo así el trimado de la planta sin un modelo previo de ella.

En nuestro caso hemos trimado nuestro modelo para combinaciones de velocidad ascensional, velocidad

aerodinámica y ángulo de balance en los siguientes puntos:

- $w = [1, 0, -1, -2]$
- $TAS = [8, 10, 12, 14]$
- $\phi = [-20, -15, -5, 0, 10, 15, 20]$

Las simulaciones son realizadas mediante el archivo 'Linerize.m' D.1. El programa también se encarga de guardar las variables de control en dichos puntos. Estos valores serán empleados en las secciones sucesivas. De los datos obtenidos realizamos una interpolación lineal y podemos obtener el vector de control en equilibrio aproximado para cualquier punto intermedio de los anteriormente planteados (ver 'uAtTrim.m' D.4).

Antes de continuar comprobamos los resultados obtenidos, para ello observamos que los mandos de control en los puntos de trimado son coherentes.

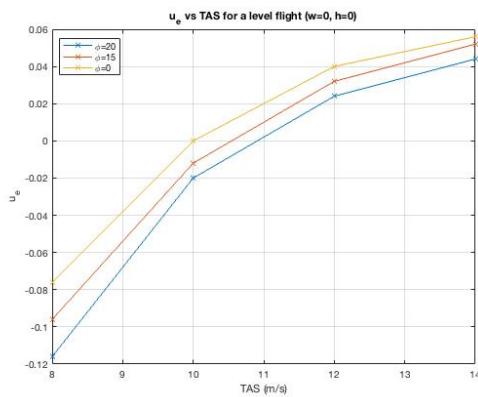


Figura 4.2: Timón de profundidad frente a TAS a distintos ángulos de balance

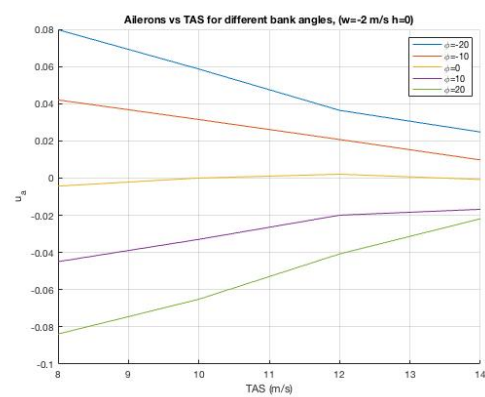


Figura 4.3: Alerones frente a TAS y ángulo de balance

Como podemos observar en la figura 4.2 a más velocidad requerimos una deflexión positiva del timón de dirección, por lo tanto la aeronave es estable como cabía de esperar. En la figura 4.3 podemos ver cómo a mayor ángulo de balance requerimos más alerones e inversamente a mayor velocidad menos alerones para un ángulo de balance dado (a efectividad de mando y densidad constante mayor velocidad implica mayor presión dinámica y menor deflexión para equilibrar los momentos).

Por otra parte en la figura 4.4 observamos el motor necesario para mantener una condición de vuelo dada. Observamos, como es lógico siempre que estemos volando en un régimen normal de vuelo. Además observamos la fuerte dependencia del motor con la velocidad vertical, para velocidades ascendentes mayores (negativas en el gráfico) requerimos mayor demanda de motor.

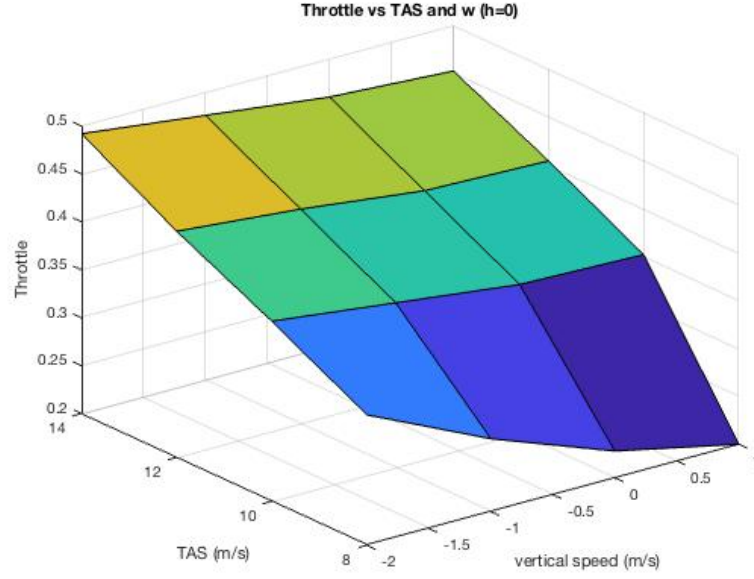


Figura 4.4: Motor frente a velocidad vertical y TAS a ϕ cte.

4.2.1. Linealización de la planta

El siguiente paso a realizar ha sido la linealización de la planta. Para ello, de nuevo, existen dos opciones, a saber, mediante la linealización de un modelo previamente creado (podríamos hacerla con herramientas disponibles en Matlab ej. linmod); por otro lado tenemos la vía escogida anteriormente, que será la elegida ahora también, realizar una linealización de la planta sin disponer de ningún modelo.

Para conseguir linealizar nuestro avión en torno al punto de equilibrio sin disponer del modelo recurrimos al método de mínimos cuadrados. Para ello realizamos una simulación de 10 segundos en la cual añadimos al control una perturbación aleatoria (ver archivo 'randU.m' D.5 y explicación en el apéndice C.3) con una frecuencia de un segundo. Con esto conseguimos que el vector de control no sea proporcional al vector de estado, por lo tanto el sistema pasa a ser observable.

La condición de vuelo escogida para realizar la linealización y sobre la cual trabajaremos de aquí en adelante en el trabajo será vuelo recto y nivelado, a una velocidad del aire de 12 m/s y una velocidad ascensional nula. En este momento realizaremos, por simplicidad, una división del problema, separando el control lateral-direccional del longitudinal. Para ello seleccionaremos los vectores de estados siguientes:

$$x_{long} = \begin{bmatrix} TAS \\ w \\ \theta \\ q \end{bmatrix} \quad x_{lat} = \begin{bmatrix} \phi \\ p \\ r \end{bmatrix} \quad (4.1)$$

Como se puede observar en 4.1 en el control lateral no hemos escogido como variable de estado β como sería normal. Esto es así porque el controlador es suficientemente bueno con estas variables de estado 'reducidas' y de esta forma evitamos la necesidad de introducir un estimador de β al sistema. Los vectores de control estarán formados por:

$$u_{long} = \begin{bmatrix} \delta_e \\ \delta_t \end{bmatrix} \quad u_{lat} = \begin{bmatrix} \delta_a \\ \delta_r \end{bmatrix} \quad (4.2)$$

Cada una de los sistemas en los que hemos dividido el problema tendrán la forma de la ecuación 4.3 . Por tanto calcularemos las matrices A y B tanto longitudinales como laterales. Esto lo realizamos en el archivo 'Linearize.m' D.1.

$$\frac{d}{dt}x = \mathbf{A}x + \mathbf{B}u \quad (4.3)$$

Los resultados de dicho proceso son las siguientes matrices:

$$A_{long} = \begin{bmatrix} -0,1292 & -1,2708 & -15,6620 & 0,2784 \\ -0,2417 & -5,3212 & -63,9929 & -5,3805 \\ 0,0128 & -0,0223 & -0,4895 & 0,8585 \\ 1,1371 & -2,4679 & -51,3030 & -14,1374 \end{bmatrix} \quad B_{long} = \begin{bmatrix} 0,8588 & 10,5359 \\ 4,8716 & -0,3890 \\ -0,7839 & 0,0045 \\ -71,5358 & 0,4542 \end{bmatrix} \quad (4.4)$$

$$A_{lat} = \begin{bmatrix} -0,1620 & 0,9835 & 0,0777 \\ -19,6822 & -0,7702 & 7,9284 \\ 11,9423 & -3,4353 & -6,6726 \end{bmatrix} \quad B_{lat} = \begin{bmatrix} -0,0027 & -0,0361 \\ -1,0878 & -5,1925 \\ 6,3080 & 12,0618 \end{bmatrix} \quad (4.5)$$

4.2.2. Síntesis

Una vez realizado el trimado y linearización de nuestro UAV, procedemos a sintetizar un controlador. Empleamos un controlador LQR a partir de las matrices A y B calculadas previamente y las siguientes matrices Q y R:

$$Q_{long} = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 5 \end{bmatrix} \quad R_{long} = \begin{bmatrix} 1000 & 0 \\ 0 & 100 \end{bmatrix} \quad (4.6)$$

$$Q_{lat} = \begin{bmatrix} 1000 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 10 \end{bmatrix} \quad Q_{lat} = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix} \quad (4.7)$$

Los pesos de estas matrices se han escogido siguiendo un proceso de prueba y error. En el caso de tener que realizar este ejercicio sobre un UAV se facilitaría disponiendo de un modelo aproximado de la planta para estimar unos valores coherentes. Sin embargo, esto último no es imprescindible y se podría realizar mediante ensayos de vuelo, recuperando la filosofía seguida en los apartados anteriores. El cálculo de las ganancias del controlador longitudinal y lateral direccional se realiza también en el archivo 'Linearize.m' D.1.

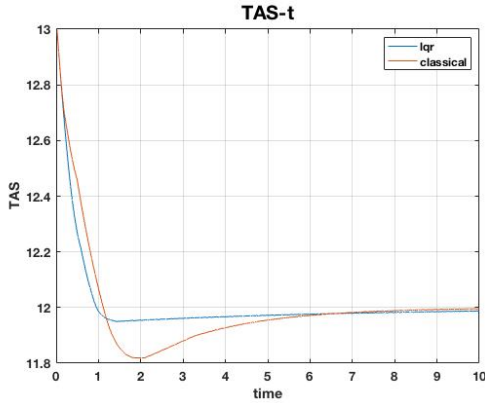


Figura 4.5: Comparación del control de la velocidad entre controlador lqr y controlador clásico

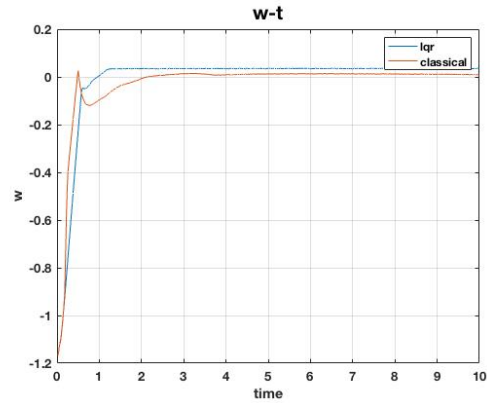


Figura 4.6: Comparación controlador lqr y controlado clásico en velocidad vertical

Realizamos brevemente una comparación del controlador clásico con el que hemos encontrado los puntos de equilibrio y el controlador lqr. Para ello realizamos dos simulaciones partiendo de un punto de no equilibrio y vemos como se comportan uno y otro. Las condiciones de inicio son:

- $\omega_0 = [0 \ 0 \ 0]$
- $Euler_0 = [5 \ 5 \ 0]$ (deg).

- $Vb_0 = [13 \ 1 \ -0,15]$

Como podemos observar en la figura 4.5 y en la figura 4.6 el comportamiento dinámico longitudinal del controlador sintetizado a partir de la linearización de la planta es más que satisfactorio. En el control lateral direccional nos encontramos con una situación similar como podemos ver en la figura 4.7. El nuevo controlador consigue alcanzar el equilibrio en un tiempo menor y con un 'overshoot' menor. Por tanto damos por bueno el trimado, linerización y síntesis del controlador hasta el momento.

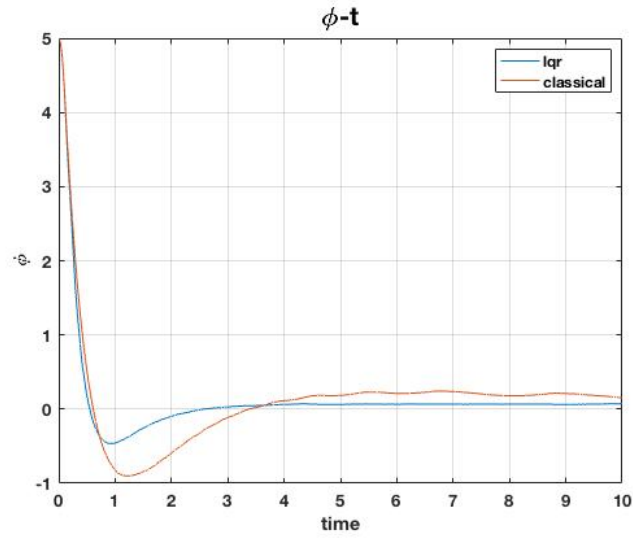


Figura 4.7: Ángulo de balance en control clásico y controlador lqr

4.3. Modelo de la planta real frente a estimación

A partir de este momento nos centraremos en el control longitudinal del UAV. Para poder comparar los dos métodos propuestos para realizar un controlador adaptativo, realizamos una modificación a nuestro modelo y comparamos el desempeño de cada uno de los métodos. La modificación ha consistido en duplicar la potencia de mando del timón de profundidad. Según F. Lewis [7], “el factor de eficiencia de la cola en aeronaves propulsadas por hélice puede exceder el valor de 2.0 para altos valores del coeficiente de tracción”. En nuestro modelo de aeronave no se ha tenido en cuenta el efecto que puede tener la hélice sobre la cola, solo se tiene en cuenta a través del modelo DATCOM la influencia del ala y fuselaje sobre ella.

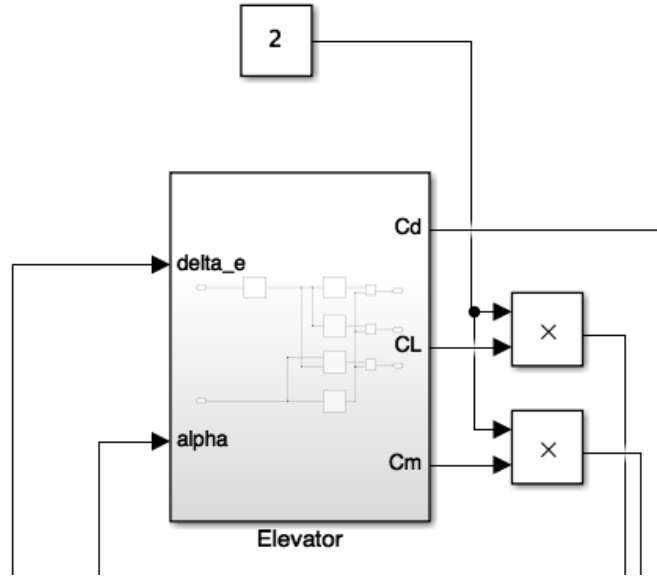


Figura 4.8: Modificación del modelo Simulink

En la figura 4.8 vemos la modificación realizada a la planta, esta modificación se encuentra dentro del modelo de Simulink en UAV / Aero / Control Forces and Moments / Aero Coefficients. Para más información sobre el modelo Simulink dirigirse a la sección C

4.4. Controladores adaptativos

Una vez realizada la modificación de la planta, estudiaremos cómo las dos aproximaciones al control adaptativo se comportan. Para ello definiremos como criterio la función de coste del controlador lqr (eq. 4.8). Siendo $Q = Q_{long}$ y $R = R_{long}$ (las matrices empleadas anteriormente para la síntesis del controlador).

$$J = \sum_{k=0}^{k-1} x'_k \cdot Q \cdot x_k + u'_k \cdot R \cdot u_k \quad (4.8)$$

Esta función se evalúa mediante la función 'Jfunction.m' ver D.6. Existe un inconveniente evaluando el desempeño de los controladores mediante esta función, y es que es dependiente del tiempo de la simulación. Por ello realizaremos las comparaciones siempre con simulaciones de 2 segundos.

4.4.1. Adaptación mediante RLSE

Estimación off-line

En un principio no se pudo realizar el cálculo de la ganancia del controlador en tiempo real. Esto se debía a que la función `lqr` empleada para calcular dicha matriz en apartados anteriores no se puede emplear dentro del modelo de Simulink (la función `lqr` se encuentra en una librería externa y Simulink no puede generar código a partir de ella). Sin embargo un trabajo posterior ha permitido calcular dicha ganancia en tiempo real, generando un sistema Matlab interpretado durante la simulación y no compilado. Se expone ahora por tanto, la adaptación 'off-line' y en el apartado siguiente la adaptación 'on-line'.

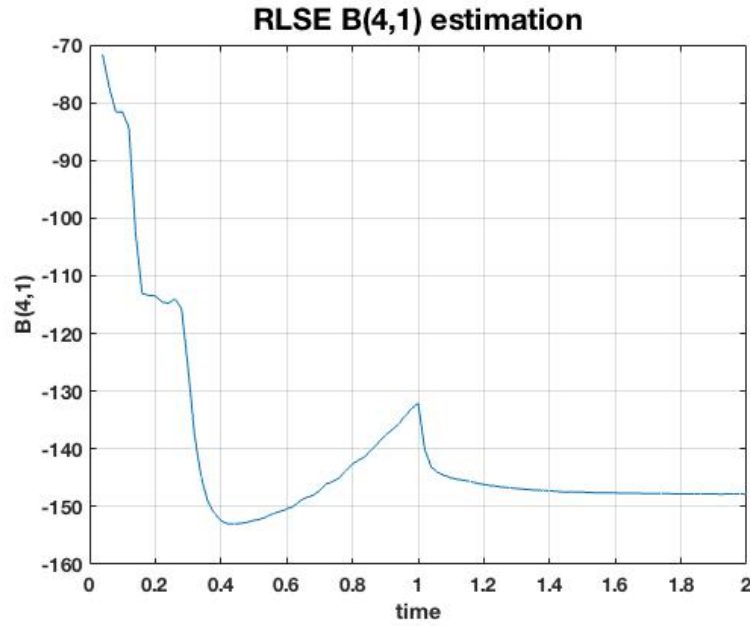


Figura 4.9: Estimación de $B(4,1)$ mediante RLSE

Realizamos una simulación durante 2 segundos con las condiciones iniciales empleadas anteriormente, y con el subsistema de entrada al mando aleatoria empleado en la linearización de la planta. Cuando empleamos este método, recalculamos a medida que nuevos datos están disponibles las ecuaciones que definen la dinámica de la planta. La modificación realizada en la planta hará que en la matriz B el elemento $(4,1)$ duplique su valor, manteniendo prácticamente constantes el resto de valores. Esto significa que la influencia del mando del timón de profundidad sobre q es el doble que anteriormente. Podemos observar en la figura 4.9, cómo este parámetro empieza con un valor de -70 y al final de la estimación tiene un valor aproximadamente de -150. Esto es coherente con el supuesto realizado anteriormente.

El cálculo a posterior empleando un estimador de mínimos cuadrados recursivo y el cálculo de la nueva ganancia del controlador se realiza en el archivo 'RLSESIM.m' D.7. El método de mínimos cuadrados recursivo se implementa como un filtro Kalman en el archivo 'RLSEklm.m' D.8.

El valor de la función de coste (eq. 4.8) para el controlador calculado en el apartado 4.2.2 en una simulación de 2 segundos es de $1.5083e+5$ mientras que con la nueva estimación es de $1.4876e+5$.

4.4.2. Adaptación mediante SPSA

Aplicamos el método SPSA para encontrar el mínimo de nuestra función partiendo del controlador lqr generado a partir del modelo de la planta sin modificar. En nuestro caso hemos parametrizado la matriz Q como podemos ver en 4.9.

$$Q = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 10 \cdot \theta_1 & 0 & 0 \\ 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 5 \cdot \theta_2 \end{bmatrix} \quad (4.9)$$

Los parámetros del método son los siguientes:

a_1	0.035	α	1	c_1	0.2
a_2	50	γ	0.4		

EL valor de a_1 se calcula en el primer bucle como indica la ecuación 4.10. De esta forma nos aseguramos que tenga un valor razonable para la primera iteración.

$$a_1 = \frac{a_2 * 0,2}{\max(abs(g))} \quad (4.10)$$

Realizamos 10 experimentos y calculamos el gradiente ponderándolo en dos iteraciones. Cada evaluación de la función de coste tarda 2 segundos por lo que tenemos que la optimización al final dura 40 segundos (10 experimentos, 2 iteraciones, 2 evaluaciones por iteración). Al final de la optimización hemos actualizado los pesos de la matriz Q como podemos ver en 4.11.

$$Q = \begin{bmatrix} 10,0000 & 0 & 0 & 0 \\ 0 & 16,3955 & 0 & 0 \\ 0 & 0 & 5,0000 & 0 \\ 0 & 0 & 0 & 6,0069 \end{bmatrix} \quad (4.11)$$

El valor de nuestra función objetivo (eq. 4.8) obtenida al final de la optimización es de $1.5071e+5$. En la figura 4.12 se ha representado la función de coste frente a los parámetros $Q(2,2)$ y $Q(4,4)$. También se puede ver el punto en el cual comienza el algoritmo SPSA y dónde termina. Todo este procedimiento se puede ver en el archivo 'SPSA.m' D.9.

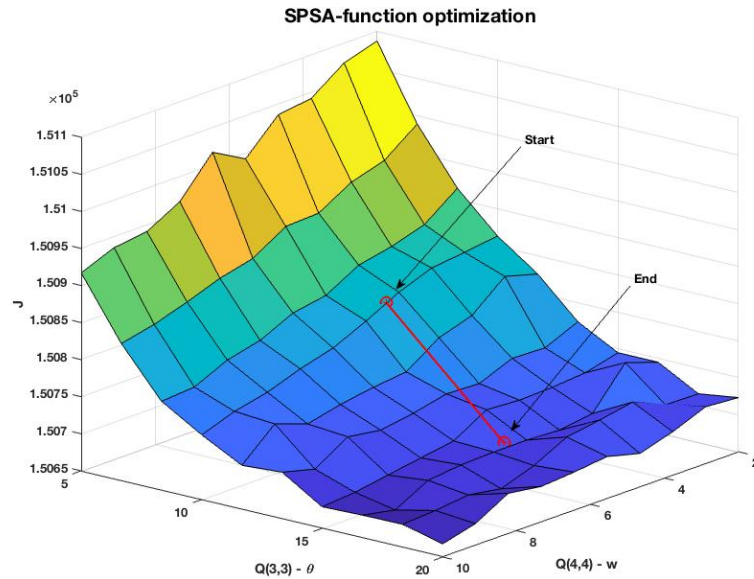


Figura 4.12: Función de coste para diversos valores de $Q(2,2)$ y $Q(4,4)$

Capítulo 5

Resultados, análisis y comparaciones

5.1. Adaptación mediante RLSE

5.1.1. Adaptación off-line

Como hemos visto en apartados anteriores, este procedimiento actualiza el modelo de la planta para sintetizar el controlador. Se puede ver en la figura 4.9 se estima rápidamente la variación en el elemento $B(4,1)$, es decir la influencia del timón de dirección respecto a q . Además, se puede ver en la figura 5.1 la respuesta del nuevo controlador es mejor que la del original. Además la función objetivo bajó de $1.5083e+5$ a $1.4876e+5$, lo que suponen una mejora del 1,4%.

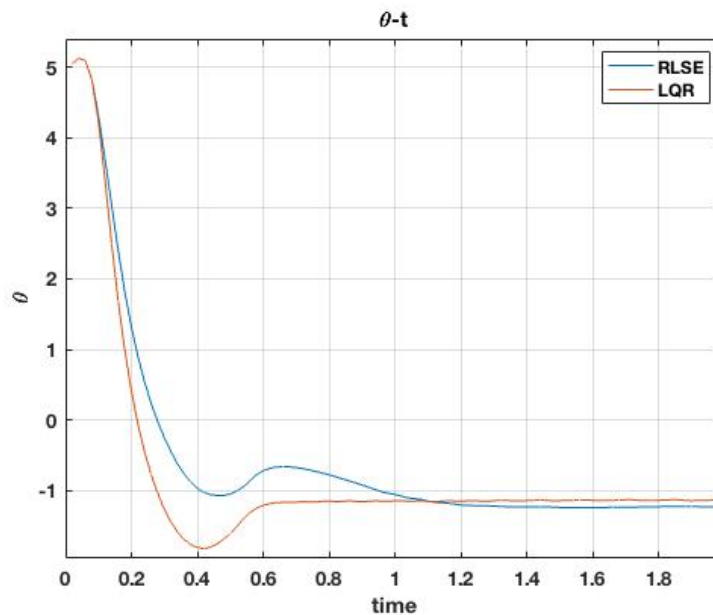


Figura 5.1: Comparación de ángulo de asiento en controlador sin modificar y el nuevo generado por RLSE

Se aprecia también en la figura cómo no se alcanza la condición de vuelo deseada, esto será común en ambos métodos de adaptación. Se debe a que el trimado se ha realizado con la planta sin modificar, por lo tanto el vector de control en el equilibrio para la condición deseada ha variado y no existe ningún integrador en el controlador. Queda por tanto para trabajos futuros implementar integradores en el control para alcanzar el punto deseado y actualizar los controles en equilibrio en tiempo real.

La respuesta del nuevo controlador en TAS y w son muy similares, además como el vuelo se realiza con alas a nivel lo que implica que $q = d/dt\theta$, por lo tanto la gráfica q - t no aporta información extra.

5.1.2. Adaptación on-line

En la figura 5.2 podemos observar el desempeño del controlador de ganancia variable con condiciones de simulación análogas a las anteriores. El 'overshoot' producido por este controlador es menor que el conseguido por el controlador original. Sin embargo también se aprecia que el nuevo es más lento, aunque el error estático es menor.

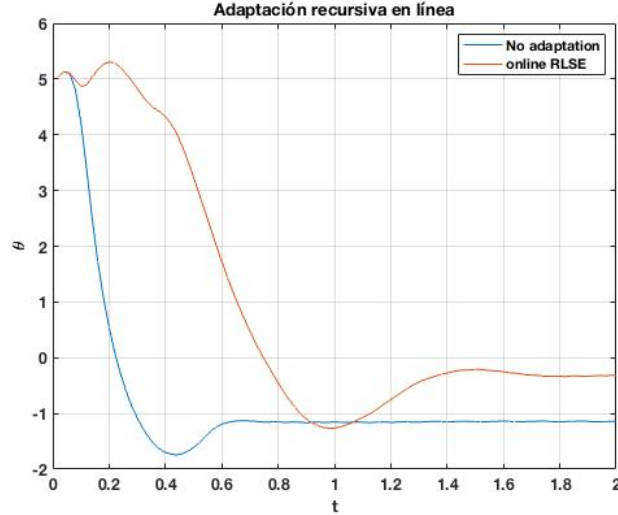


Figura 5.2: Comparación ángulo de asiento en controlador sin modificar y el controlador con ganancia calculada on-line

Tal y como se dijo en la sección 4.4.1 para evaluar el desempeño de este algoritmo realizamos una simulación de 10 segundos con una perturbación de viento. Se ha empleado un modelo de perturbación atmosférica continuo, los parámetros de este modelo se pueden ver en la figura 5.5. Como se puede observar en la figura 5.3 la respuesta de este controlador es mejor que el original. Conserva las cualidades anteriores, menor error estático (o error medio ahora) y un rechazo ante perturbaciones menor (antes veíamos un 'overshoot' menor).

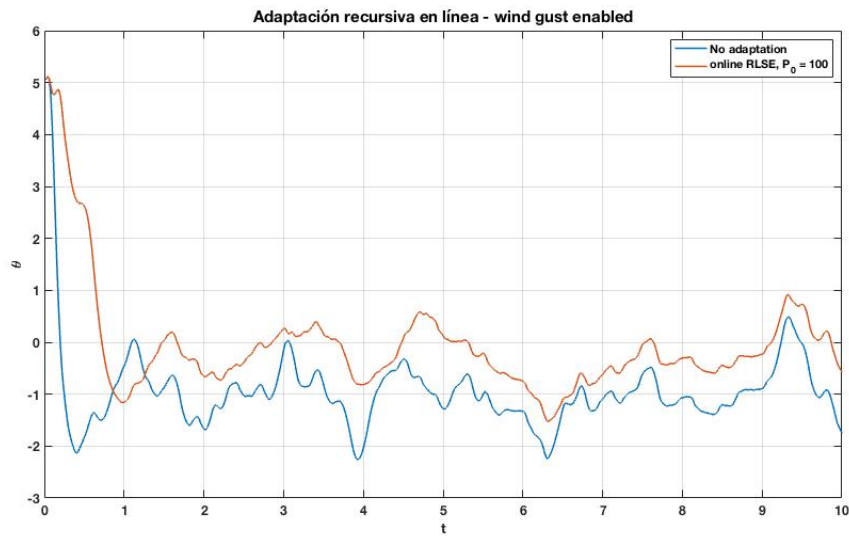


Figura 5.3: Comparación ángulo de asiento en controlador sin modificar y el controlador con ganancia calculada on-line con iguales perturbaciones

El valor de la función de coste para esta simulación con el controlador original es de $7.3177e+5$ mientras que el nuevo es de $7.4214e+5$, obteniendo una mejoría del 1.41 %. En la figura 5.4 se representa la velocidad angular de cabeceo de ambos controladores. En esta figura se aprecia más claramente cómo la capacidad de rechazo de perturbaciones del nuevo controlador ha mejorado.

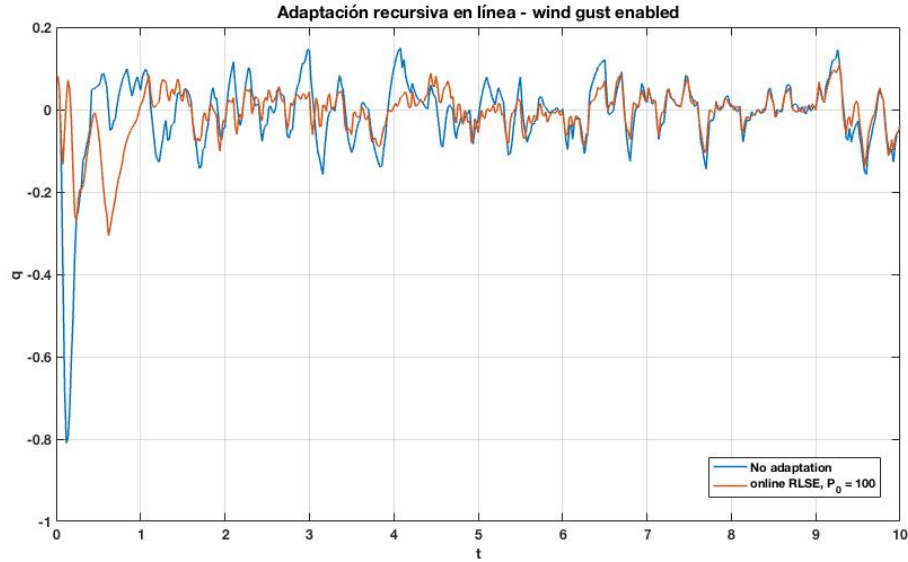


Figura 5.4: Comparación de q en controlador sin modificar y el controlador con ganancia calculada on-line con iguales perturbaciones

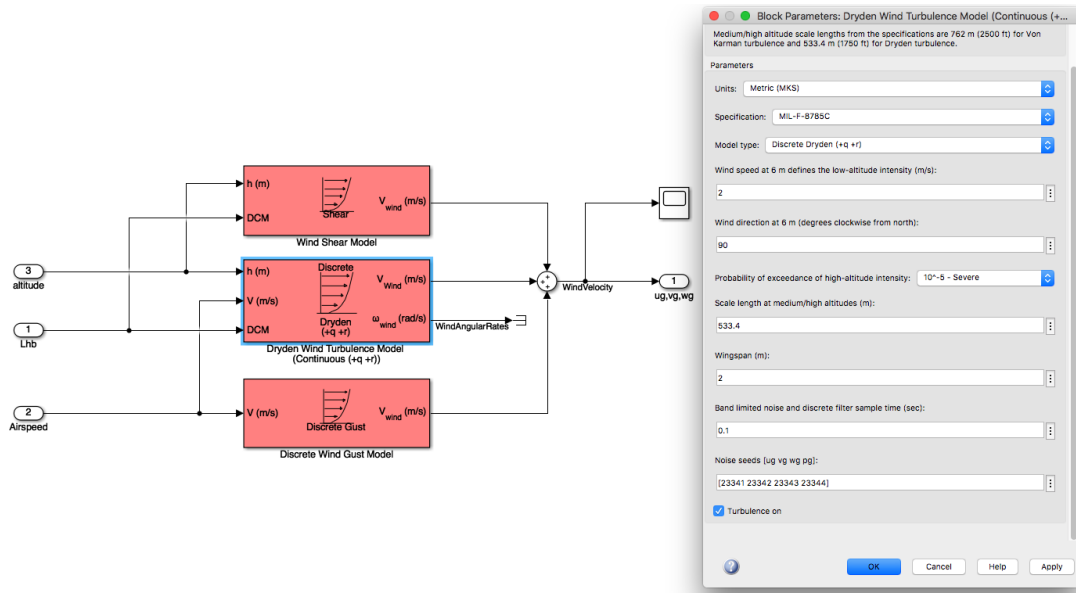


Figura 5.5: Parámetros del modelo de perturbación atmosférica

5.2. Adaptación mediante SPSA

Como podemos observar en la figura 5.6 la respuesta del controlador sintetizado tras el proceso de adaptación de los parámetros es similar al controlador no modificado. Tarda algo menos en alcanzar el equilibrio y tiene un 'overshoot' menor. Además el error estacionario es algo menor que en el controlador sin modificar (como cabe esperar al aumentar el parámetro $Q(2,2)$).

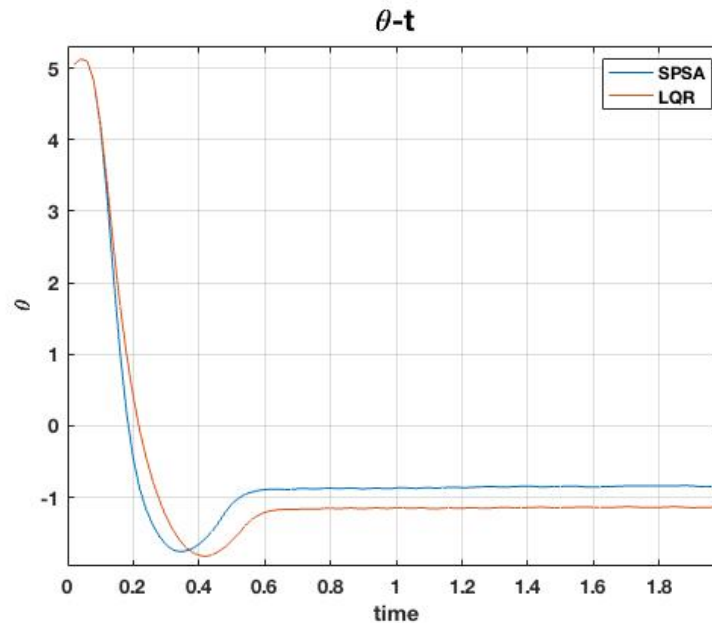


Figura 5.6: Comparación de ángulo de asiento en controlador sin modificar y la adaptación SPSA

5.3. Comparación de ambos métodos

Se ha visto cómo ambos métodos de adaptación han respondido en el sentido esperado, mejorando el rendimiento del controlador original. Si bien es cierto que dicha mejora no ha sido muy sustancial, sí puede llegar a ser importante, incluso crítica en situaciones de vuelo real, como puede ser el fallo de un servo o un fallo de algún componente estructural de la aeronave. Ambas aproximaciones tienen sus ventajas e inconvenientes que intentaremos resumir brevemente.

- En cuanto al **número de parámetros**, vemos que si empleamos el método RLSE solo tendremos que preocuparnos de un parámetro, la matriz P , la cual, habrá que controlar en tiempo real que tenga un valor alto cuando el error de estimación de la planta también lo sea. Por otro lado, en la aproximación SPSA tenemos 5 parámetros que tendremos que escoger en función de la planta, además de la necesidad de parametrizar las matrices Q y R .
- Respecto a **tiempo de adaptación** sin lugar a dudas la estimación recursiva de la planta es mucho más rápida que la adaptación mediante SPSA (RLSE orden de segundos, SPSA orden de cientos de segundos).
- Si nos fijamos en la **capacidad de adaptación**, entendida como la capacidad del método para variar el comportamiento del controlador frente a cambios en la planta, podemos decir que es mejor el método RLSE. Sin embargo esto tiene su peligro también y es que puede ser que la estimación de la planta se empeore y por tanto empeore sustancialmente el comportamiento.
- La **complejidad y capacidad de procesamiento** para poder realizar la adaptación de ambos métodos en tiempo real difiere mucho, ya que el método RLSE implica invertir una matriz (en este

caso de 6x6) lo cual es computacionalmente costoso. Mientras tanto el algoritmo SPSA podemos decir que es computacionalmente bastante 'ligero'.

En ambas aproximaciones se deberán realizar comprobaciones como puede ser el signo de algunos elementos de las matrices A y B, o su magnitud en el método RLSE, o como el signo de los elementos del vector de parámetros en el método SPSA.

5.4. Trabajo futuro

El trabajo realizado hasta el momento solamente pone de manifiesto la viabilidad de la aplicación de las técnicas de control adaptativas a aeronaves no tripuladas. Como tal es preciso decir que el trabajo inmediatamente posterior al realizado, sería la implementación de dichos métodos en una aeronave y la comprobación empírica de los resultados obtenidos en este estudio. Además, sería interesante profundizar en mayor medida en numerosos aspectos como pueden ser los siguientes:

- Estudio de una ley de decisión que permita activar el mecanismo de adaptación. Ambas técnicas estudiadas en este trabajo mejoran la respuesta dinámica del UAV cuando la modelización del mismo no es óptima; sin embargo el empleo de dichas técnicas de manera continuada penalizan el desempeño de la aeronave (ya sea por la excitación continuada necesaria para RLSE como la implementación de ganancias sub-óptimas dado un modelo para el mecanismo de adaptación basado en SPSA).
- La aplicación del método basado en el algoritmo SPSA tiene una gran ventaja, como ya se enunció en la sección 3.3, frente a otros algoritmos de optimización cuando el número de variables a optimizar es elevado. En el caso de estudio propuesto, se ha empleado un modelo con dos variables a optimizar, por lo que sería interesante realizar:
 - a) Un estudio más exhaustivo sobre la parametrización del problema de optimización, es decir, sobre qué elementos de Q y R la función de coste J es más sensible.
 - b) Ampliar el método propuesto y considerar un sistema de más de dos parámetros a optimizar, sacando con ello más provecho de las ventajas del algoritmo SPSA.
 - c) Comparar el algoritmo propuesto con otros de optimización basados en el gradiente o en métodos eurísticos, sin fijar el número de parámetros a optimizar, es decir, ya sea con dos, tres o 'n' variables (es razonable pensar que dependiendo del número de variables escogidas será más eficiente un método u otro).
- En el trabajo expuesto se ha realizado una aplicación diferencial de los métodos, es decir, aplicamos uno u otro desde un principio y comparamos resultados de ambos por separado. Queda pues para trabajos futuros el estudio de la aplicación de un método de identificación de la planta seguido de un método de ajuste de los parámetros del controlador.

Capítulo 6

Conclusiones

Primera

Las técnicas de control adaptativo son una herramienta disponible actualmente para aumentar la seguridad de operación y tolerancia al daño de las aeronaves no tripuladas. El coste computacional de dichas técnicas es asequible, gracias a la gran capacidad de cálculo disponible en un peso reducido.

Segunda

Los métodos de adaptación basados en la estimación recursiva de la planta son rápidos pero computacionalmente pesados. Ofrecen una gran capacidad de adaptación, esto también conlleva un peligro de divergencia de la planta si la estimación no es correcta.

Tercera

Los métodos de adaptación basados en el ajuste de los pesos del controlador son lentos, pero computacionalmente ligeros. Tiene menor capacidad de adaptación, siendo por tanto también menos susceptibles a una divergencia en el comportamiento de la planta.

Cuarta

Las técnicas de control adaptativo requieren de perturbaciones en la condición de equilibrio de la planta para poder mejorar la respuesta del sistema. La perturbación continuada de la planta no es deseable por lo que el desarrollo de dichas técnicas ha de ir ligado a un mecanismo de activación de las mismas.

Apéndice A

Modelo físico y másico del UAV

A.1. Modelo másico, cálculo de propiedades

A continuación se muestra el código empleado para realizar los cálculos de posición del centro de gravedad y los momentos de inercia de la aeronave.

```
1 %=====
2 % TFG - Adaptative control for UAVs %
3 % 10/08/2019 %
4 % %
5 % CG, INERTIAS AND WEIGHTS ESTIMATIONS %
6 % %
7 % Dom nguez Alegre, Carlos F. %
8 %=====
9 % SI units
10 % Dom nguez Alegre, Carlos F.
11 % 2019
12
13 Wwing = 0.47; % wing mass
14 Wfus = 0.650; % fuselage and tail mass
15 Wtail = Wfus*0.20; % tail mass (from tail mass fraction)
16 Wbat = 0.18; % batery mass
17 Wplane = Wbat+Wfus+Wwing;
18
19 lt = 0.70; % tail-cg_fuselage length
20 B = 2; % wingspan
21 c = 0.23; % MAC
22 e = 0.02; % profile thickness
23
24
25 % Lengths referenced to a body axis attached to the nose of the plane, x
26 % positive forward, z positive down
27
28 xcgBat = [-0.17,0,0];
29 xcgWing= [-0.31,0,-0.05];
30 xcgFus = [-0.34,0,0];
31
32 l1Bat = 0.1; % batery length
33 l2Bat = 0.035; % batery height
34 l3Bat = 0.025; % batery width
35
36
37 Bfus = 0.06; % characteristic fuselage width
38 Lfus = 1.1; % characteristic fuselage length
39
40
41 % WING MOMENT OF INERTIA ESTIMATION
42 % Approximated by a prism of the same thickness and length. The width will
43 % be taken as 50% of the chord. The CDG of the wing is located at 1/3 of
44 % the chord.
45 Iw = zeros(3,3);
46 Iw(1,1) = (1/12)*Wwing*(e^2+B^2);
47 Iw(2,2) = (1/12)*Wwing*(e^2+(0.5*c)^2);
48 Iw(3,3) = (1/12)*Wwing*(B^2+(0.5*c)^2);
```

```

49
50
51
52 % BATTERY MOMENT OF INERTIA ESTIMATION
53 Ib = zeros(3,3);
54 Ib(1,1) = (1/12)*Wbat*(l2Bat^2+l3Bat^2);
55 Ib(2,2) = (1/12)*Wbat*(l1Bat^2+l2Bat^2);
56 Ib(3,3) = (1/12)*Wbat*(l1Bat^2+l3Bat^2);
57
58
59
60 % FUSELAGE MOMENT OF INERTIA ESTIMATION
61 % For the Iyy and Izz moments of inertia the fuselage is considered as two
62 % different solids, first a stretch of fuselage of length xcgFus(1) and
63 % another representing the tail. The moment of inertia of the tail section
64 % within the body axis of the tail will be dispise. Ixx moment of inertia
65 % wil be estimated as a homogeneous prism.
66
67 If = zeros(3,3);
68 If(1,1) = (1/6)*Wfus*Bfus^2;
69
70
71 It = zeros(3,3);
72 It(2,2) = Wtail*lt^2;
73 It(3,3) = Wtail*lt^2;
74
75
76 Wff = Wfus-Wtail;
77 Iff_cg = zeros(3,3);
78 Iff_cg(2,2) = (1/12)*Wff*(Bfus^2+xcgFus(1)^2);
79 Iff_cg(3,3) = Iff_cg(2,2);
80
81 lff = lt*Wtail/Wff;
82
83 Iff = zeros(3,3);
84 Iff(2,2) = Iff_cg(2,2) + Wff*lff^2; %steiner
85 Iff(3,3) = Iff_cg(3,3) + Wff*lff^2;
86
87 If = Iff + It +If;
88
89
90
91 % COG AND MOMENTS OF INERTIA ESTIMATION OF THE FUSELAGE
92
93 XCG = (xcgBat*Wbat + xcgFus*Wfus + xcgWing*Wwing)/Wplane;
94
95 %Fuselage
96 OG = xcgFus-XCG;
97 If_cg = If + Wfus*(norm(OG)^2*eye(3,3) - kron(OG,OG'))
98
99 %Battery
100 OG = xcgBat-XCG;
101 Ib_cg = Ib + Wbat*(norm(OG)^2*eye(3,3) - kron(OG,OG'))
102
103 %Wing
104 OG = xcgWing-XCG;
105 Iw_cg = Iw + Wwing*(norm(OG)^2*eye(3,3) - kron(OG,OG'))
106
107
108 % TOTAL
109 Icg = zeros(3,3);
110 Icg = If_cg + Ib_cg + Iw_cg;
111
112
113 %-----
114 % Data out
115 display("C lculo de pesos y tensor de inercia del UAV")
116 display(Icg);
117 display(Wplane);
118 display(XCG);
119
120

```

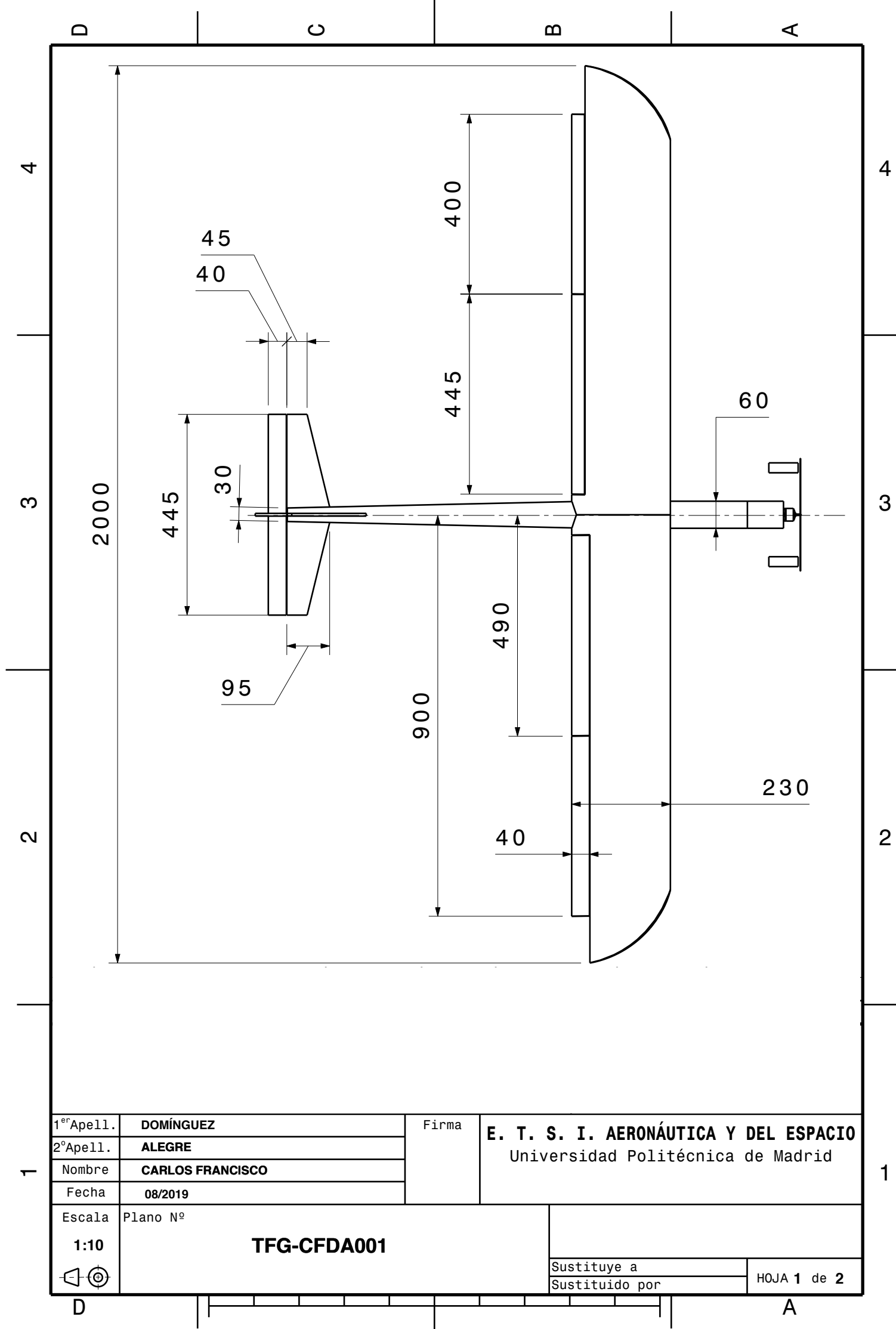
```


121 myfile = fopen("UAV_Icg.txt", 'w')
122 fmt = '%1.4f %1.4f %1.4f\n'; fprintf(myfile,fmt,Icg); fclose(myfile);
123
124 myfile = fopen("UAV_mass.txt", 'w');
125 fmt = '%1.4f\n'; fprintf(myfile,fmt,Wplane);fclose(myfile);
126
127
128 myfile = fopen("UAV_Xcg.txt", 'w');
129 fmt = '%1.4f\n'; fprintf(myfile,fmt,XCG);fclose(myfile);

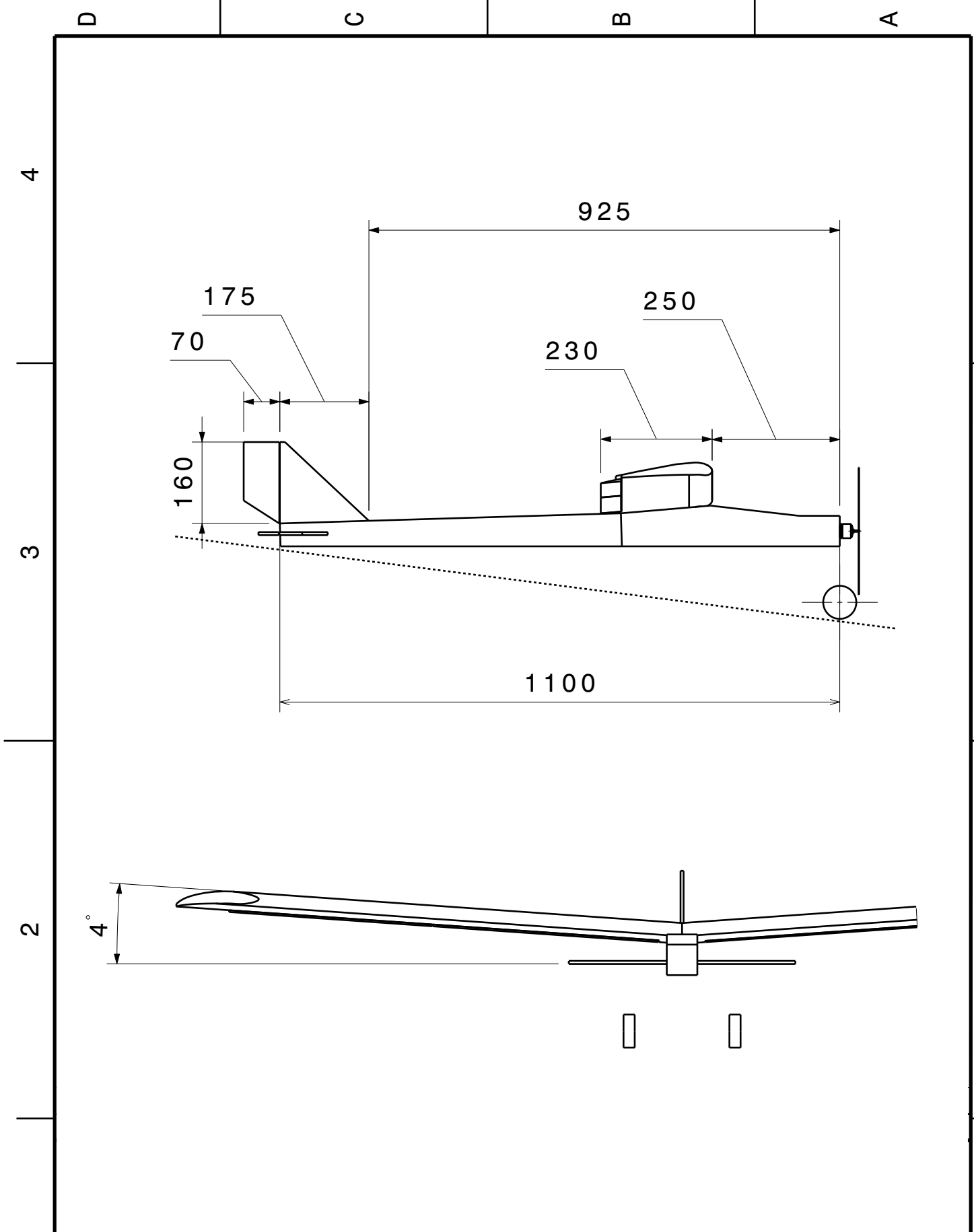
```

A.2. Modelo físico

Se muestran a continuación unos esquemas de la aeronave modelizada con las medidas empleadas en el modelado necesario para DATCOM (ver apéndice B).



1 ^{er} Apell.	DOMÍNGUEZ	Firma	E. T. S. I. AERONÁUTICA Y DEL ESPACIO Universidad Politécnica de Madrid	
2ºApell.	ALEGRE			
Nombre	CARLOS FRANCISCO			
Fecha	08/2019			
Escala	Plano Nº			
1:10	TFG-CFDA001			
			Sustituye a	
			Sustituido por	
				HOJA 1 de 2



1 ^{er} Apell.	DOMÍNGUEZ	Firma	E. T. S. I. AERONÁUTICA Y DEL ESPACIO Universidad Politécnica de Madrid
2 ^o Apell.	ALEGRE		
Nombre	CARLOS FRANCISCO		
Fecha	08/2019		
Escala	Plano N°		
1:10	TFG-CFDA001		
	Sustituye a		HOJA 2 de 2
	Sustituido por		

Apéndice B

Modelo DATCOM del UAV

La estimación del modelo aerodinámico del UAV se ha realizado mediante el programa 'Digital DATCOM', el programa consiste en un compendio de métodos de estimación de las derivadas de estabilidad y control para aeronaves. El programa representa una valiosa herramienta a la hora de modelizar la dinámica de vuelo de una aeronave, especialmente para fases tempranas de diseño o en el caso que nos concierne, para una estimación previa de la planta a controla. Para más información sobre su uso y limitaciones referirse al manual volumen I [1].

Archivo de entrada

```
1 $FLTCOM NMACH = 1.0, MACH(1)=0.042, NALPHA = 10.0,
2     ALSCHD(1) = -4.0, -2.0, 0.0, 2.0, 4.0,
3     6.0, 8.0, 10.0, 12.0, 14.0,
4     NALT=1.0, ALT(1)=0.0, WT=12.74$
5 $OPTINS SREF=0.414, CBARR=0.23, BLREF=1.0$
6 $SYNTHS XCG=0.305, ZCG=0.0, XW=0.255, ZW=0.04,
7     ALIW=4.5, XH=1.0, ZH=0.0,
8     XV=0.915, ZV=0.0$
9 $BODY NX = 5.0,
10     X(1)=0.0,0.08,0.23,0.43,1.0,
11     S(1)=0.0036,0.0036,0.0048,0.00325,0.00135$
12 NACA-W-24.12
13 $WGPLNF CHRDT=0.1, SSPN=1.0, SSPNOP=0.2,
14     SSPNE=0.975, SAVSI=0.0, SAVSO=0.0,
15     CHSTAT=1.0, DHDADO=4.0, CHRDBP=0.23,
16     CHRDR=0.23,
17     TWISTA=0.0, DHDADI=4.0, TYPE=1.0$
18 NACA-H-6-66-009
19 $HTPLNF CHRDT=0.085, SSPN=0.2225,
20     SSPNE=0.2075, SAVSI=0.0,
21     CHSTAT=1.0,
22     CHRDR=0.14,
23     TYPE=1.0$
24 NACA-V-6-66-009
25 $VTPLNF CHRDT=0.07, SSPN=0.19,
26     SSPNE=0.17, SAVSI=43.36,
27     CHSTAT=0.0,
28     CHRDR=0.215,
29     TYPE=1.0$
30 $ASYFLP STYPE=4.0, NDELTA=9.0, SPANFI=0.5,
31     SPANFO=0.9,
32     DELTAR(1)=-20.,-15.,-10.,-5.,0.,
33     5.,10.,15.,20.,
34     DELTAL(1)=20.,15.,10.,5.,0.,
35     -5.,-10.,-15.,-20.,
36     CHRDFI=0.03, CHRDFO=0.03$
37 DIM M
38 DAMP
39 CASEID CFDA UAV - CNda
40 SAVE
41 NEXT CASE
```

```

42 $SYMFLP FTYPE=1.0, NDELTA=9.0, SPANFI=0.025,
43     SPANFO=0.2225,
44     DELTA(1)=-20.,-15.,-10.,-5.,0.,
45     5.,10.,15.,20.,
46     CHRDFI=0.04,CHRDF0=0.04, NTYPE=1.$
47 CASEID CFD UAV-CM

```

B.1. Archivo de salida

```

1  THIS SOFTWARE AND ANY ACCOMPANYING DOCUMENTATION
2  IS RELEASED "AS IS". THE U.S. GOVERNMENT MAKES NO
3  WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, CONCERNING
4  THIS SOFTWARE AND ANY ACCOMPANYING DOCUMENTATION,
5  INCLUDING, WITHOUT LIMITATION, ANY WARRANTIES OF
6  MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.
7  IN NO EVENT WILL THE U.S. GOVERNMENT BE LIABLE FOR ANY
8  DAMAGES, INCLUDING LOST PROFITS, LOST SAVINGS OR OTHER
9  INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE
10 USE, OR INABILITY TO USE, THIS SOFTWARE OR ANY
11 ACCOMPANYING DOCUMENTATION, EVEN IF INFORMED IN ADVANCE
12 OF THE POSSIBILITY OF SUCH DAMAGES.
13 *****
14 *   USAF STABILITY AND CONTROL DIGITAL DATCOM   *
15 *   PROGRAM REV. JAN 96 DIRECT INQUIRIES TO:    *
16 *   WRIGHT LABORATORY (WL/FIGC) ATTN: W. BLAKE  *
17 *   WRIGHT PATTERSON AFB, OHIO 45433           *
18 *   PHONE (513) 255-6764, FAX (513) 258-4054   *
19 *****
20 Preparing to start the big loop
21 At 1000
22 1 CONERR - INPUT ERROR CHECKING
23 0 ERROR CODES - N* DENOTES THE NUMBER OF OCCURENCES OF EACH ERROR
24 0 A - UNKNOWN VARIABLE NAME
25 0 B - MISSING EQUAL SIGN FOLLOWING VARIABLE NAME
26 0 C - NON-ARRAY VARIABLE HAS AN ARRAY ELEMENT DESIGNATION - (N)
27 0 D - NON-ARRAY VARIABLE HAS MULTIPLE VALUES ASSIGNED
28 0 E - ASSIGNED VALUES EXCEED ARRAY DIMENSION
29 0 F - SYNTAX ERROR
30
31 0***** INPUT DATA CARDS *****
32
33 $FLTCON NMACH = 1.0, MACH(1)=0.042, NALPHA = 10.0,
34     ALSCHD(1) = -4.0, -2.0, 0.0, 2.0, 4.0,
35     6.0, 8.0, 10.0, 12.0, 14.0,
36     NALT=1.0, ALT(1)=0.0, WT=12.74$
37 $OPTINS SREF=0.414, CBARR=0.23, BLREF=1.0$
38 $SYNTHS XCG=0.305, ZCG=0.0, XW=0.255, ZW=0.04,
39     ALIW=4.5, XH=1.0, ZH=0.0,
40     XV=0.915, ZV=0.0$
41 $BODY NX = 5.0,
42     X(1)=0.0,0.08,0.23,0.43,1.0,
43     S(1)=0.0036,0.0036,0.0048,0.00325,0.00135$
44 NACA-W-24.12
45 $WGPLNF CHRDP=0.1, SSPN=1.0, SSPNOP=0.2,
46     SSPNE=0.975, SAVSI=0.0, SAVSO=0.0,
47     CHSTAT=1.0, DHDADO=4.0, CHRDBP=0.23,
48     CHRDR=0.23,
49     TWISTA=0.0, DHDADI=4.0, TYPE=1.0$
50 NACA-H-6-66-009
51 $HTPLNF CHRDP=0.085, SSPN=0.2225,
52     SSPNE=0.2075, SAVSI=0.0,
53     CHSTAT=1.0,
54     CHRDR=0.14,
55     TYPE=1.0$
56 NACA-V-6-66-009
57 $VTPLNF CHRDP=0.07, SSPN=0.19,
58     SSPNE=0.17, SAVSI=43.36,
59     CHSTAT=0.0,
60     CHRDR=0.215,
61     TYPE=1.0$
62 $ASYFLP STYPE=4.0, NDELTA=9.0, SPANFI=0.5,
63     SPANFO=0.9,
64     DELTAR(1)=-20.,-15.,-10.,-5.,0.,
65     5.,10.,15.,20.,
66     DELTAL(1)=20.,15.,10.,5.,0.,
67     -5.,-10.,-15.,-20.,
68     CHRDFI=0.03,CHRDF0=0.03$
69 DIM M
70 DAMP
71 CASEID CFDA UAV - CNda
72 SAVE
73 NEXT CASE
74 $SYMFLP FTYPE=1.0, NDELTA=9.0, SPANFI=0.025,
75     SPANFO=0.2225,
76     DELTA(1)=-20.,-15.,-10.,-5.,0.,
77     5.,10.,15.,20.,
78     CHRDFI=0.04,CHRDF0=0.04, NTYPE=1.$
79 CASEID CFD UAV-CM
80 1 THE FOLLOWING IS A LIST OF ALL INPUT CARDS FOR THIS CASE.
81 0
82 $FLTCON NMACH = 1.0, MACH(1)=0.042, NALPHA = 10.0,
83     ALSCHD(1) = -4.0, -2.0, 0.0, 2.0, 4.0,
84     6.0, 8.0, 10.0, 12.0, 14.0,
85     NALT=1.0, ALT(1)=0.0, WT=12.74$
86 $OPTINS SREF=0.414, CBARR=0.23, BLREF=1.0$
87 $SYNTHS XCG=0.305, ZCG=0.0, XW=0.255, ZW=0.04,
88     ALIW=4.5, XH=1.0, ZH=0.0,
89     XV=0.915, ZV=0.0$
90 $BODY NX = 5.0,
91     X(1)=0.0,0.08,0.23,0.43,1.0,
92     S(1)=0.0036,0.0036,0.0048,0.00325,0.00135$
93 NACA-W-24.12
94 $WGPLNF CHRDP=0.1, SSPN=1.0, SSPNOP=0.2,
95     SSPNE=0.975, SAVSI=0.0, SAVSO=0.0,
96     CHSTAT=1.0, DHDADO=4.0, CHRDBP=0.23,

```

```

97   CHRDR=0.23,
98   TWISTA=0.0, DHDADI=4.0, TYPE=1.0$
99   NACA-H-6-66-009
100  $HTPLNF CHRDR=0.085, SSPN=0.2225,
101  SSPNE=0.2075, SAVSI=0.0,
102  CHSTAT=1.0,
103  CHRDR=0.14,
104  TYPE=1.0$
105  NACA-V-6-66-009
106  $VTPLNF CHRDR=0.07, SSPN=0.19,
107  SSPNE=0.17, SAVSI=43.36,
108  CHSTAT=0.0,
109  CHRDR=0.215,
110  TYPE=1.0$
111  $ASYFLP STYPE=4.0, NDELTA=9.0, SPANFI=0.5,
112  SPANFO=0.9,
113  DELTAR(1)=-20.,-15.,-10.,-5.,0.,
114  5.,10.,15.,20.,
115  DELTAL(1)=20.,15.,10.,5.,0.,
116  -5.,-10.,-15.,-20.,
117  CHRDFI=0.03,CHRDF0=0.03$
118  DIM M
119  DAMP
120  CASEID CFDA UAV - CNda
121  SAVE
122  NEXT CASE
123  0 INPUT DIMENSIONS ARE IN M , SCALE FACTOR IS 1.0000
124
125  Return to main program from M01001
126  1 AUTOMATED STABILITY AND CONTROL METHODS PER APRIL 1976 VERSION OF DATCOM
127  WING SECTION DEFINITION
128  0 IDEAL ANGLE OF ATTACK = 0.50437 DEG.
129
130  ZERO LIFT ANGLE OF ATTACK = -0.94733 DEG.
131
132  IDEAL LIFT COEFFICIENT = 0.15390
133
134  ZERO LIFT PITCHING MOMENT COEFFICIENT = -0.01717
135
136  MACH ZERO LIFT-CURVE-SLOPE = 0.10004 /DEG.
137
138  LEADING EDGE RADIUS = 0.00000 FRACTION CHORD
139
140  MAXIMUM AIRFOIL THICKNESS = 0.00000 FRACTION CHORD
141
142  DELTA-Y = 0.00000 PERCENT CHORD
143
144
145  0 MACH= 0.0420 LIFT-CURVE-SLOPE = 0.10013 /DEG. XAC = 0.25930
146  1 AUTOMATED STABILITY AND CONTROL METHODS PER APRIL 1976 VERSION OF DATCOM
147  HORIZONTAL TAIL SECTION DEFINITION
148  0 IDEAL ANGLE OF ATTACK = 0.00000 DEG.
149
150  ZERO LIFT ANGLE OF ATTACK = -0.00000 DEG.
151
152  IDEAL LIFT COEFFICIENT = 0.00000
153
154  ZERO LIFT PITCHING MOMENT COEFFICIENT = 0.00000
155
156  MACH ZERO LIFT-CURVE-SLOPE = 0.09870 /DEG.
157
158  LEADING EDGE RADIUS = 0.00559 FRACTION CHORD
159
160  MAXIMUM AIRFOIL THICKNESS = 0.09000 FRACTION CHORD
161
162  DELTA-Y = 1.52747 PERCENT CHORD
163
164
165  0 MACH= 0.0420 LIFT-CURVE-SLOPE = 0.09881 /DEG. XAC = 0.26230
166  1 AUTOMATED STABILITY AND CONTROL METHODS PER APRIL 1976 VERSION OF DATCOM
167  VERTICAL TAIL SECTION DEFINITION
168  0 IDEAL ANGLE OF ATTACK = 0.00000 DEG.
169
170  ZERO LIFT ANGLE OF ATTACK = -0.00000 DEG.
171
172  IDEAL LIFT COEFFICIENT = 0.00000
173
174  ZERO LIFT PITCHING MOMENT COEFFICIENT = 0.00000
175
176  MACH ZERO LIFT-CURVE-SLOPE = 0.09870 /DEG.
177
178  LEADING EDGE RADIUS = 0.00559 FRACTION CHORD
179
180  MAXIMUM AIRFOIL THICKNESS = 0.09000 FRACTION CHORD
181
182  DELTA-Y = 1.52747 PERCENT CHORD
183
184
185  0 MACH= 0.0420 LIFT-CURVE-SLOPE = 0.09881 /DEG. XAC = 0.26230
186  Return to main program from M50062
187  Return to main program from M02002
188  Return to main program from M51063
189
190  1 AUTOMATED STABILITY AND CONTROL METHODS PER APRIL 1976 VERSION OF DATCOM
191  CHARACTERISTICS AT ANGLE OF ATTACK AND IN SIDESLIP
192  WING-BODY-VERTICAL TAIL-HORIZONTAL TAIL CONFIGURATION
193  CFDA UAV - CNda
194
195  ----- FLIGHT CONDITIONS ----- REFERENCE DIMENSIONS -----
196  MACH ALTITUDE VELOCITY PRESSURE TEMPERATURE REYNOLDS REF. REFERENCE LENGTH MOMENT REF. CENTER
197  NUMBER M M/SEC N/ M**2 DEG K NUMBER AREA LONG. LAT. HORIZ VERT
198  0 0.042 0.00 14.29 1.0133E+05 288.150 9.7397E+05 0.414 0.230 1.000 0.305 0.000
199  0
200  0 ALPHA CD CL CM CN CA XCP CLA CMA CYB CNB CLB
201  0
202  -4.0 0.024 0.076 0.1015 0.074 0.029 1.364 9.205E-02 -2.149E-02 -2.467E-03 1.106E-03 -2.391E-03
203  -2.0 0.035 0.260 0.0520 0.258 0.044 0.201 8.995E-02 -2.269E-02 -2.408E-03
204  0.0 0.057 0.436 0.0107 0.436 0.057 0.025 8.571E-02 -2.042E-02 -2.422E-03
205  2.0 0.086 0.603 -0.0296 0.605 0.065 -0.049 7.395E-02 -2.145E-02 -2.434E-03
206  4.0 0.114 0.732 -0.0751 0.738 0.063 -0.102 5.450E-02 -2.333E-02 -2.437E-03
207  6.0 0.135 0.820 -0.1229 0.830 0.049 -0.148 4.668E-02 -2.494E-02 -2.430E-03
208  8.0 0.163 0.919 -0.1749 0.932 0.033 -0.188 3.959E-02 -2.593E-02 -2.426E-03
209  10.0 0.180 0.979 -0.2267 0.995 0.007 -0.228 3.310E-02 -2.684E-02 -2.413E-03
210  12.0 0.203 1.051 -0.2822 1.070 -0.020 -0.264 3.723E-02 -2.779E-02 -2.399E-03

```

```

211 14.0 0.231 1.128 -0.3378 1.150 -0.049 -0.294 3.958E-02 -2.783E-02 -2.384E-03
212 0 ALPHA Q/QINF EPSLON D(EPSLON)/D(ALPHA)
213 0
214 -4.0 0.989 0.534 0.359
215 -2.0 0.920 1.252 0.353
216 0.0 0.923 1.946 0.336
217 2.0 0.993 2.597 0.302
218 4.0 1.000 3.156 0.250
219 6.0 1.000 3.596 0.201
220 8.0 1.000 3.958 0.177
221 10.0 1.000 4.305 0.157
222 12.0 1.000 4.585 0.151
223 14.0 1.000 4.907 0.161
224 1 AUTOMATED STABILITY AND CONTROL METHODS PER APRIL 1976 VERSION OF DATCOM
225 DYNAMIC DERIVATIVES
226 WING-BODY-VERTICAL TAIL-HORIZONTAL TAIL CONFIGURATION
227 CFDA UAV - CNda
228
229 ----- FLIGHT CONDITIONS ----- REFERENCE DIMENSIONS -----
230 MACH ALTITUDE VELOCITY PRESSURE TEMPERATURE REYNOLDS REF. REFERENCE LENGTH MOMENT REF. CENTER
231 NUMBER M M/SEC N/ M**2 DEG K NUMBER AREA LONG. LAT. HORIZ VERT
232 0 0.042 0.00 14.29 1.0133E+05 288.150 9.7397E+05 1/ M M**2 M M M M
233 0 0.042 0.00 14.29 1.0133E+05 288.150 9.7397E+05 0.414 0.230 1.000 0.305 0.000
234 DYNAMIC DERIVATIVES (PER DEGREE)
235 -----PITCHING----- ACCELERATION----- ROLLING----- YAWING-----
236 0 ALPHA CLQ CMQ CLAD CMAD CLP CYP CNP CNR CLR
237 0
238 -4.0 1.093E-01 -1.903E-01 1.976E-02 -6.460E-02 -3.585E-02 -3.901E-03 8.193E-04 -1.929E-03 8.953E-04
239 -2.0 1.805E-02 -5.902E-02 -3.484E-02 -3.930E-03 1.608E-03 -2.162E-03 1.605E-03
240 0.0 1.725E-02 -5.640E-02 -3.332E-02 -4.020E-03 2.104E-03 -2.551E-03 2.292E-03
241 2.0 1.670E-02 -5.460E-02 -2.809E-02 -4.362E-03 1.361E-03 -3.062E-03 2.934E-03
242 4.0 1.389E-02 -4.541E-02 -1.960E-02 -5.028E-03 -8.246E-04 -3.530E-03 3.389E-03
243 6.0 1.116E-02 -3.647E-02 -1.662E-02 -5.393E-03 -1.837E-03 -3.866E-03 3.658E-03
244 8.0 9.865E-03 -3.225E-02 -1.364E-02 -5.866E-03 -3.140E-03 -4.295E-03 3.982E-03
245 10.0 8.717E-03 -2.850E-02 -1.094E-02 -6.224E-03 -4.242E-03 -4.526E-03 4.119E-03
246 12.0 8.372E-03 -2.737E-02 -1.298E-02 -6.192E-03 -3.923E-03 -4.853E-03 4.325E-03
247 14.0 8.968E-03 -2.932E-02 -1.392E-02 -6.292E-03 -3.977E-03 -5.225E-03 4.552E-03
248 0*** VEHICLE WEIGHT = 12.74 N.
249 0*** LEVEL FLIGHT LIFT COEFFICIENT = 1.09405
250 1 AUTOMATED STABILITY AND CONTROL METHODS PER APRIL 1976 VERSION OF DATCOM
251 CHARACTERISTICS OF HIGH LIFT AND CONTROL DEVICES
252 WING PLAIN TRAILING-EDGE FLAP CONFIGURATION
253 CFDA UAV - CNda
254 ----- FLIGHT CONDITIONS ----- REFERENCE DIMENSIONS -----
255 MACH ALTITUDE VELOCITY PRESSURE TEMPERATURE REYNOLDS REF. REFERENCE LENGTH MOMENT REF. CENTER
256 NUMBER M M/SEC N/ M**2 DEG K NUMBER AREA LONG. LAT. HORIZ VERT
257 0 0.042 0.00 14.29 4.4198E+01 933.606 9.0485E+04 1/ M M**2 M M M M
258 0 0.042 0.00 14.29 4.4198E+01 933.606 9.0485E+04 0.414 0.230 1.000 0.305 0.000
259 0 -----YAWING MOMENT COEFFICIENT,CN,DUE TO CONTROL DEFLECTION-----
260 0(DELTA-D) = 40.0 30.0 20.0 10.0 0.0 -10.0 -20.0 -30.0 -40.0
261 0ALPHA
262 0
263 -4.0 -1.548E-03 -1.272E-03 -8.660E-04 -4.330E-04 0.000E+00 4.330E-04 8.660E-04 1.272E-03 1.548E-03
264 -2.0 -3.649E-03 -2.999E-03 -2.041E-03 -1.021E-03 0.000E+00 1.021E-03 2.041E-03 2.999E-03 3.649E-03
265 0.0 -5.689E-03 -4.676E-03 -3.182E-03 -1.591E-03 0.000E+00 1.591E-03 3.182E-03 4.676E-03 5.689E-03
266 2.0 -7.607E-03 -6.251E-03 -4.255E-03 -2.127E-03 0.000E+00 2.127E-03 4.255E-03 6.251E-03 7.607E-03
267 4.0 -9.017E-03 -7.410E-03 -5.044E-03 -2.522E-03 0.000E+00 2.522E-03 5.044E-03 7.410E-03 9.017E-03
268 6.0 -9.914E-03 -8.148E-03 -5.545E-03 -2.773E-03 0.000E+00 2.773E-03 5.545E-03 8.148E-03 9.914E-03
269 8.0 -1.096E-02 -9.010E-03 -6.132E-03 -3.066E-03 0.000E+00 3.066E-03 6.132E-03 9.010E-03 1.096E-02
270 10.0 -1.150E-02 -9.450E-03 -6.432E-03 -3.216E-03 0.000E+00 3.216E-03 6.432E-03 9.450E-03 1.150E-02
271 12.0 -1.222E-02 -1.004E-02 -6.836E-03 -3.418E-03 0.000E+00 3.418E-03 6.836E-03 1.004E-02 1.222E-02
272 14.0 -1.300E-02 -1.068E-02 -7.271E-03 -3.636E-03 0.000E+00 3.636E-03 7.271E-03 1.068E-02 1.300E-02
273 0
274 0 DELTA DELTAR (CL)ROLL
275 0
276 20.0 -20.0 1.0774E-01
277 15.0 -15.0 8.8540E-02
278 10.0 -10.0 6.0262E-02
279 5.0 -5.0 3.0131E-02
280 0.0 0.0 0.0000E+00
281 -5.0 5.0 -3.0131E-02
282 -10.0 10.0 -6.0262E-02
283 -15.0 15.0 -8.8540E-02
284 -20.0 20.0 -1.0774E-01
285 Return to main program from M57071
286 1 THE FOLLOWING IS A LIST OF ALL INPUT CARDS FOR THIS CASE.
287 0
288 $SYNMLP FTYPE=1.0, NDELTA=9.0, SPANFI=0.025,
289 SPANFO=0.2225,
290 DELTA(1)=-20.,-15.,-10.,-5.,0.,
291 5.,10.,15.,20.,
292 CHRDFI=0.04,CHRDFO=0.04, NTYPE=1.$
293 CASEID CFD UAV-CM
294 0 INPUT DIMENSIONS ARE IN M, SCALE FACTOR IS 1.0000
295
296 Return to main program from M01001
297 1 AUTOMATED STABILITY AND CONTROL METHODS PER APRIL 1976 VERSION OF DATCOM
298 WING SECTION DEFINITION
299 0 IDEAL ANGLE OF ATTACK = 0.50437 DEG.
300
301 ZERO LIFT ANGLE OF ATTACK = -0.94733 DEG.
302
303 IDEAL LIFT COEFFICIENT = 0.15390
304
305 ZERO LIFT PITCHING MOMENT COEFFICIENT = -0.01717
306
307 MACH ZERO LIFT-CURVE-SLOPE = 0.10004 /DEG.
308
309 LEADING EDGE RADIUS = 0.00000 FRACTION CHORD
310
311 MAXIMUM AIRFOIL THICKNESS = 0.00000 FRACTION CHORD
312
313 DELTA-Y = 0.00000 PERCENT CHORD
314
315
316 0 MACH= 0.0420 LIFT-CURVE-SLOPE = 0.10013 /DEG. XAC = 0.25930
317 1 AUTOMATED STABILITY AND CONTROL METHODS PER APRIL 1976 VERSION OF DATCOM
318 HORIZONTAL TAIL SECTION DEFINITION
319 0 IDEAL ANGLE OF ATTACK = 0.00000 DEG.
320
321 ZERO LIFT ANGLE OF ATTACK = -0.00000 DEG.
322
323 IDEAL LIFT COEFFICIENT = 0.00000
324

```

```

325 ZERO LIFT PITCHING MOMENT COEFFICIENT = 0.00000
326
327 MACH ZERO LIFT-CURVE-SLOPE = 0.09870 /DEG.
328
329 LEADING EDGE RADIUS = 0.00559 FRACTION CHORD
330
331 MAXIMUM AIRFOIL THICKNESS = 0.09000 FRACTION CHORD
332
333 DELTA-Y = 1.52747 PERCENT CHORD
334
335
336 MACH= 0.0420 LIFT-CURVE-SLOPE = 0.09881 /DEG. XAC = 0.26230
337 1 AUTOMATED STABILITY AND CONTROL METHODS PER APRIL 1976 VERSION OF DATCOM
338 VERTICAL TAIL SECTION DEFINITION
339 0 IDEAL ANGLE OF ATTACK = 0.00000 DEG.
340
341 ZERO LIFT ANGLE OF ATTACK = -0.00000 DEG.
342
343 IDEAL LIFT COEFFICIENT = 0.00000
344
345 ZERO LIFT PITCHING MOMENT COEFFICIENT = 0.00000
346
347 MACH ZERO LIFT-CURVE-SLOPE = 0.09870 /DEG.
348
349 LEADING EDGE RADIUS = 0.00559 FRACTION CHORD
350
351 MAXIMUM AIRFOIL THICKNESS = 0.09000 FRACTION CHORD
352
353 DELTA-Y = 1.52747 PERCENT CHORD
354
355
356 MACH= 0.0420 LIFT-CURVE-SLOPE = 0.09881 /DEG. XAC = 0.26230
357 Return to main program from M50062
358 Return to main program from M02002
359 Return to main program from M51063
360 1 AUTOMATED STABILITY AND CONTROL METHODS PER APRIL 1976 VERSION OF DATCOM
361 CHARACTERISTICS AT ANGLE OF ATTACK AND IN SIDESLIP
362 WING-BODY-VERTICAL TAIL-HORIZONTAL TAIL CONFIGURATION
363 CFD UAV-CM
364
365 ----- FLIGHT CONDITIONS ----- REFERENCE DIMENSIONS -----
366 MACH ALTITUDE VELOCITY PRESSURE TEMPERATURE REYNOLDS REF. REFERENCE LENGTH MOMENT REF. CENTER
367 NUMBER M M/SEC N/ M**2 DEG K NUMBER AREA LONG. LAT. HORIZ VERT
368 0 0.042 0.00 14.29 1.0133E+05 288.150 9.7397E+05 1/ M M**2 M M M M
369 0 0.414 0.230 1.000 0.305 0.000
370 0 -----DERIVATIVE (PER DEGREE)-----
371 0 ALPHA CD CL CM CN CA XCP CLA CMA CYB CNB CLB
372 0
373 -4.0 0.024 0.076 0.1015 0.074 0.029 1.364 9.205E-02 -2.149E-02 -2.467E-03 1.106E-03 -2.391E-03
374 -2.0 0.035 0.260 0.0520 0.258 0.044 0.201 8.995E-02 -2.269E-02 -2.408E-03
375 0.0 0.057 0.436 0.0107 0.436 0.057 0.025 8.571E-02 -2.042E-02 -2.422E-03
376 2.0 0.086 0.603 -0.0296 0.605 0.065 -0.049 7.395E-02 -2.145E-02 -2.434E-03
377 4.0 0.114 0.732 -0.0751 0.738 0.063 -0.102 5.450E-02 -2.333E-02 -2.437E-03
378 6.0 0.135 0.820 -0.1229 0.830 0.049 -0.148 4.668E-02 -2.494E-02 -2.430E-03
379 8.0 0.163 0.919 -0.1749 0.932 0.033 -0.188 3.959E-02 -2.593E-02 -2.426E-03
380 10.0 0.180 0.979 -0.2267 0.995 0.007 -0.228 3.310E-02 -2.684E-02 -2.413E-03
381 12.0 0.203 1.051 -0.2822 1.070 -0.020 -0.264 3.723E-02 -2.779E-02 -2.399E-03
382 14.0 0.231 1.128 -0.3378 1.150 -0.049 -0.294 3.958E-02 -2.783E-02 -2.384E-03
383 0 ALPHA Q/QINF EPSLON D(EPSLON)/D(ALPHA)
384 0
385 -4.0 0.989 0.534 0.359
386 -2.0 0.920 1.252 0.353
387 0.0 0.923 1.946 0.336
388 2.0 0.993 2.597 0.302
389 4.0 1.000 3.156 0.250
390 6.0 1.000 3.596 0.201
391 8.0 1.000 3.958 0.177
392 10.0 1.000 4.305 0.157
393 12.0 1.000 4.585 0.151
394 14.0 1.000 4.907 0.161
395 0*** VEHICLE WEIGHT = 12.74 N.
396 0*** LEVEL FLIGHT LIFT COEFFICIENT = 1.09405
397 1 AUTOMATED STABILITY AND CONTROL METHODS PER APRIL 1976 VERSION OF DATCOM
398 CHARACTERISTICS OF HIGH LIFT AND CONTROL DEVICES
399 TAIL PLAIN TRAILING-EDGE FLAP CONFIGURATION
400 CFD UAV-CM
401 ----- FLIGHT CONDITIONS ----- REFERENCE DIMENSIONS -----
402 MACH ALTITUDE VELOCITY PRESSURE TEMPERATURE REYNOLDS REF. REFERENCE LENGTH MOMENT REF. CENTER
403 NUMBER M M/SEC N/ M**2 DEG K NUMBER AREA LONG. LAT. HORIZ VERT
404 0 0.042 0.00 14.29 4.4198E+01 933.606 9.0485E+04 1/ M M**2 M M M M
405 0 0.414 0.230 1.000 0.305 0.000
406 0 -----DERIVATIVES (PER DEGREE)-----
407 0 DELTA D(CD) D(CM) D(CD MAX) D(CD MIN) (CLA)D (CH)A (CH)D
408 0
409
410 -20.0 -0.068 0.1995 0.049 0.00462 NDM -5.543E-03 -1.421E-02
411 -15.0 -0.063 0.1863 0.039 0.00310 NDM -1.347E-02
412 -10.0 -0.044 0.1304 0.028 0.00155 NDM -1.329E-02
413 -5.0 -0.022 0.0652 0.014 0.00076 NDM -1.329E-02
414 0.0 0.000 -0.0001 0.000 0.00000 NDM -1.329E-02
415 5.0 0.022 -0.0652 0.014 0.00076 NDM -1.329E-02
416 10.0 0.044 -0.1304 0.028 0.00155 NDM -1.329E-02
417 15.0 0.063 -0.1863 0.039 0.00310 NDM -1.347E-02
418 20.0 0.068 -0.1999 0.049 0.00462 NDM -1.421E-02
419 0 *** NOTE * HINGE MOMENT DERIVATIVES ARE BASED ON TWICE THE AREA-MOMENT OF THE CONTROL ABOUT ITS HINGE LINE
420
421 0 ----- INDUCED DRAG COEFFICIENT INCREMENT , D(CDI) , DUE TO DEFLECTION -----
422 0 DELTA = -20.0 -15.0 -10.0 -5.0 0.0 5.0 10.0 15.0 20.0
423 ALPHA
424 0
425 -4.0 5.11E-03 4.63E-03 2.71E-03 1.05E-03 -1.47E-06 -4.26E-04 -2.33E-04 4.26E-04 6.29E-04
426 -2.0 4.31E-03 3.88E-03 2.19E-03 7.83E-04 -9.45E-07 -1.63E-04 2.93E-04 1.18E-03 1.43E-03
427 0.0 -5.33E-04 -6.66E-04 -9.97E-04 -8.08E-04 2.24E-06 1.43E-03 3.48E-03 5.73E-03 6.27E-03
428 2.0 2.65E-03 2.32E-03 1.10E-03 2.38E-04 1.45E-07 3.82E-04 1.38E-03 2.74E-03 3.09E-03
429 4.0 1.75E-03 1.48E-03 5.03E-04 -5.80E-05 7.37E-07 6.78E-04 1.97E-03 3.58E-03 3.99E-03
430 6.0 7.76E-04 5.64E-04 -1.37E-04 -3.78E-04 1.38E-06 9.98E-04 2.61E-03 4.50E-03 4.96E-03
431 8.0 -2.47E-04 -3.97E-04 -8.09E-04 -7.14E-04 2.05E-06 1.33E-03 3.29E-03 5.46E-03 5.99E-03
432 10.0 -1.28E-03 -1.37E-03 -1.49E-03 -1.05E-03 2.73E-06 1.67E-03 3.97E-03 6.43E-03 7.02E-03
433 12.0 -2.35E-03 -2.38E-03 -2.19E-03 -1.41E-03 3.43E-06 2.03E-03 4.67E-03 7.44E-03 8.10E-03
434 14.0 -3.40E-03 -3.36E-03 -2.88E-03 -1.75E-03 4.12E-06 2.37E-03 5.36E-03 8.42E-03 9.14E-03
435 0***NDM PRINTED WHEN NO DATCOM METHODS EXIST
436 Return to main program from M57071
437 1 END OF JOB.

```

B.2. Derivadas de estabilidad y control

Comprobamos que las derivadas de estabilidad y control obtenidas del modelo DATCOM, asegurándonos que tienen signos y valores coherentes:

- $Cl_\beta < 0$ debido al efecto diedro.
- $Cn_\beta < 0$ como corresponde a la configuración tradicional de una aeronave.
- $Cm_\alpha < 0$ índice de estabilidad estática longitudinal con mandos fijos (estable).
- $Cm_{\delta_e} < 0$ Potencia control longitudinal (deflexión positiva produce momentos negativos).
- $Cy_\beta < 0$ como corresponde al criterio de signos.

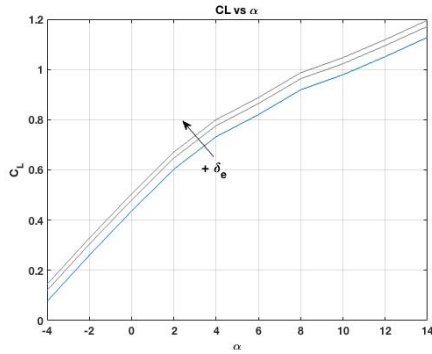


Figura B.1: Coeficiente de sustentación total

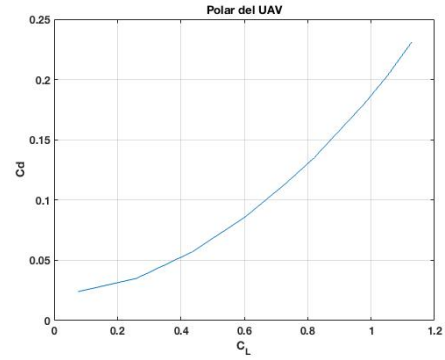


Figura B.2: Polar del avión

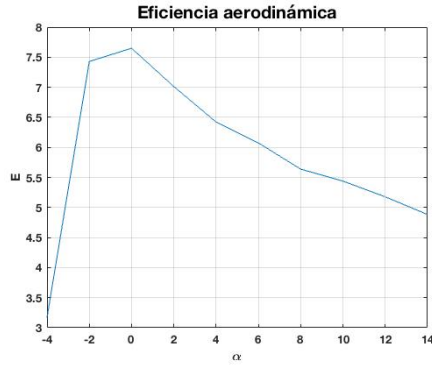


Figura B.3: Eficiencia aerodinámica

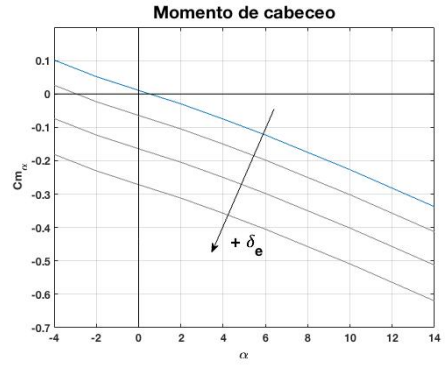


Figura B.4: Momento de cabeceo

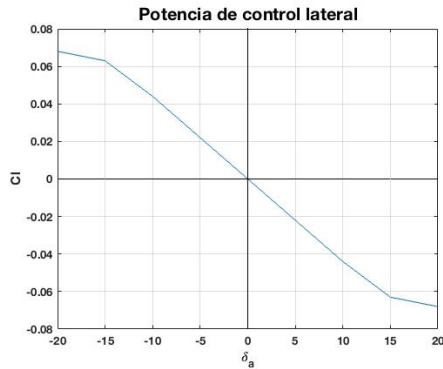


Figura B.5: Potencia de control lateral

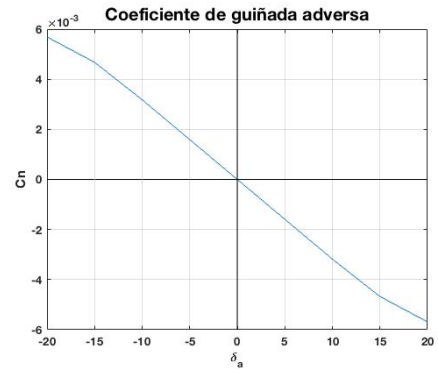


Figura B.6: Momento de guiñada adversa

Apéndice C

Modelo Simulink UAV

Se ha tomado como partida el ejemplo dispuesto por Matlab [3] generalizándolo de 3 grados de libertad (DOF de ahora en adelante) a 6 DOF, e incluyendo la modelización de los mandos de control de la aeronave.

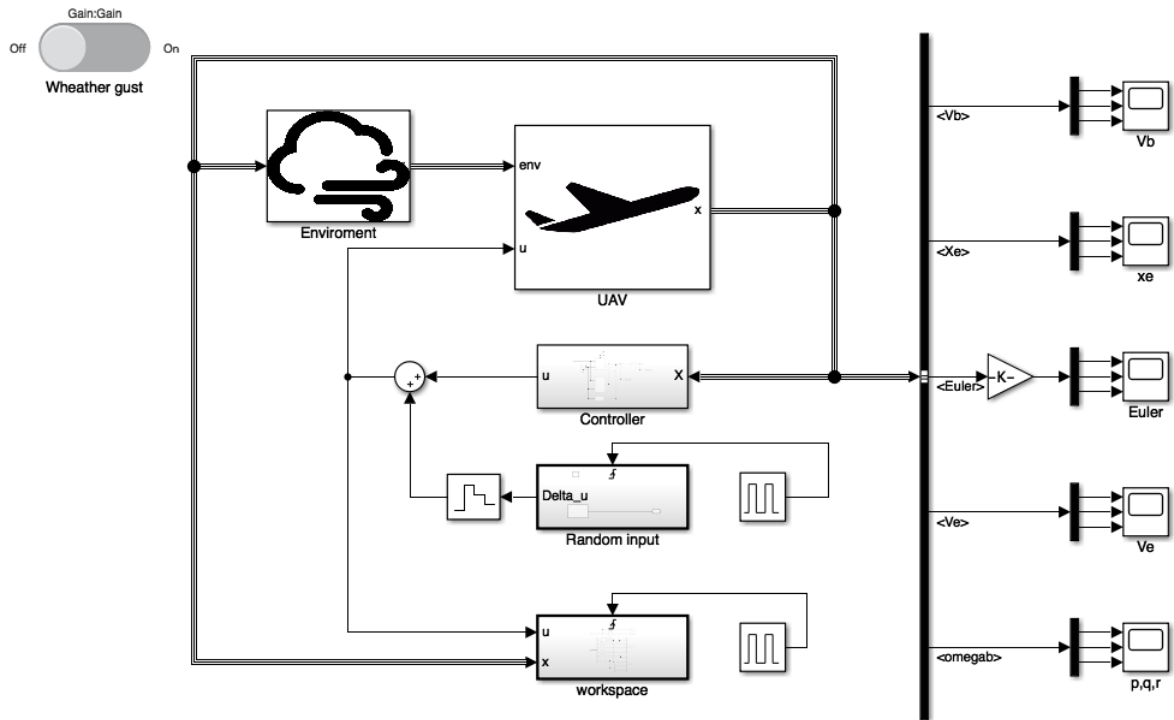


Figura C.1: Modelo del UAV Simulink

A continuación se describe el funcionamiento de cada uno de los subsistemas que componen este modelo de Simulink.

C.1. Subsistema 'UAV'

Este sistema representa la aeronave, las entradas a él son las variables de entorno (densidad, temperatura, a , velocidad del viento), y las variables de mando normalizadas ($u \in \mathbb{R}^4 | \forall u_i \in [-1, 1]$, excepto u_4 que está acotado de $[0, 1]$). Como salida tenemos el bus de datos X , en el cual encontramos las siguientes variables:

- **Ve**: velocidad del avión en ejes tierra.
- **Vb**: componentes de la velocidad del viento en ejes cuerpo.

- **omegab**: velocidad angular en ejes cuerpo.
- **domegab**: aceleración angular en ejes cuerpo.
- **Euler**: ángulos de Euler.
- **LBH**: Matriz de cambio de ejes cuerpo-horizonte local.
- **Xe**: posición del avión respecto a los ejes horizonte local.
- **Abe**: aceleración en ejes tierra.
- **Abb**: aceleración en ejes cuerpo (incluido efectos no inerciales).

Como se puede observar, en el bus de datos X tenemos variables no independientes entre ellas, como pueden ser la matriz de cambio de base con los ángulos de Euler o las velocidades respecto al suelo y al aire si no existe viento. Se han escogido estas variables por comodidad, de esta forma no se necesita recalcular dichas variables en otras partes del modelo.

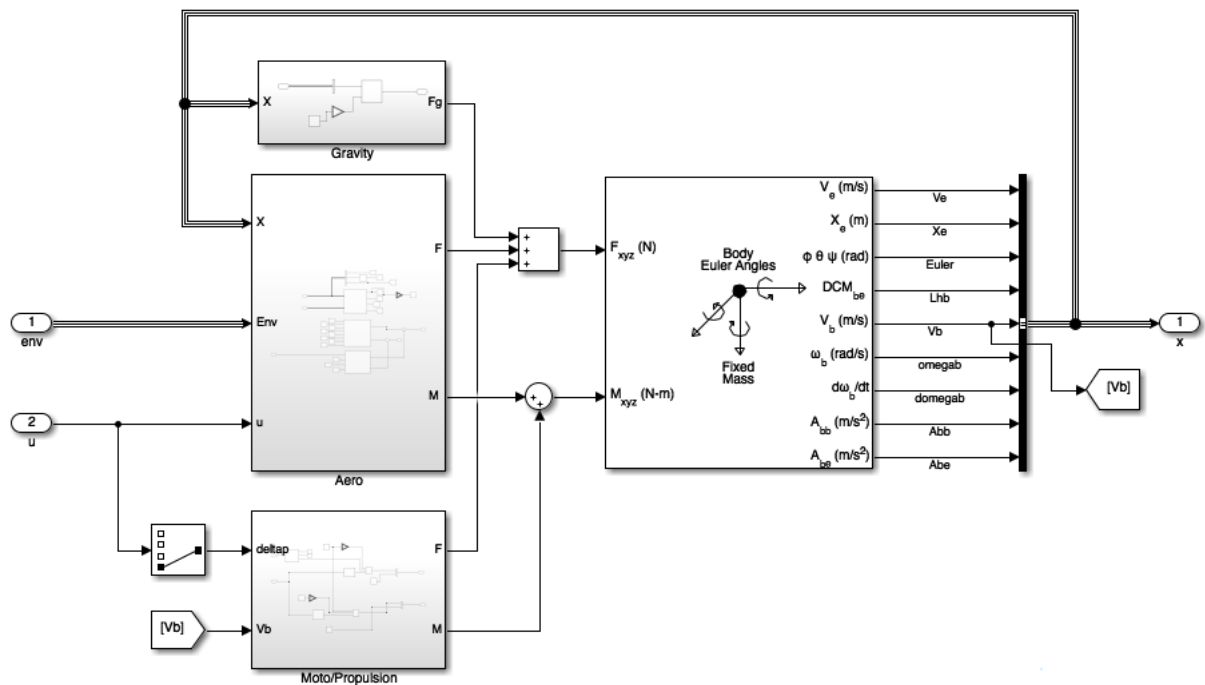


Figura C.2: Esquema interno del subsistema UAV

Como se puede observar en la figura C.2, dentro del modelo UAV tenemos el siguiente esquema de funcionamiento, primero calculamos las fuerzas y momentos sobre el centro de gravedad de la aeronave, después calculamos la posición, velocidad y aceleración del vehículo. Dentro del cálculo de fuerzas y momentos se han discernido según su origen, gravedad, aerodinámica o motora. Todas las fuerzas y momentos están expresados ya en ejes cuerpo.

El bloque 'Gravity' calcula las componentes de la fuerza gravitatoria en ejes cuerpo.

El bloque 'Aero' calcula las fuerzas y momentos debido a las fuerzas aerodinámicas, dentro de él encontramos:

- **'Derived'**: donde se calculan magnitudes derivadas del bus 'X' y de la información de 'enviroment', como pueden ser el número de mach, el ángulo de ataque, presión dinámica... Estas variables serán empleadas en los bloques siguientes.

- **'digital DATCOM forces and moments':** este es un bloque disponible en el 'Aerospace Blockset' de Simulink. Como podemos ver en la figura C.3 se configura el bloque para que calcule las fuerzas y momentos sobre la aeronave a partir del modelo DATCOM elaborado anteriormente (ver B). Para cargar en memoria las derivadas de estabilidad y control generadas a partir del modelo DATCOM se utiliza el programa 'loadDATCOMmodel.m' (D.4).

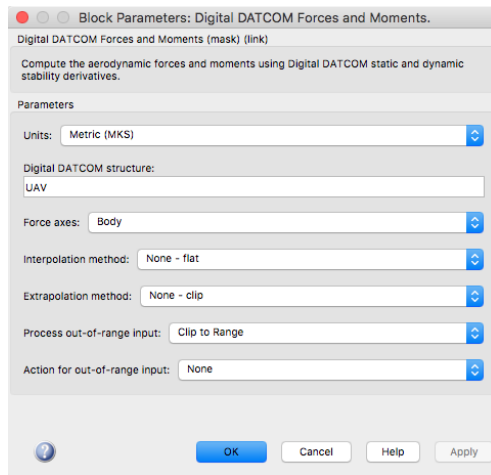


Figura C.3: Opciones dentro del bloque DATCOM de Simulink (Aerospace Blockset)

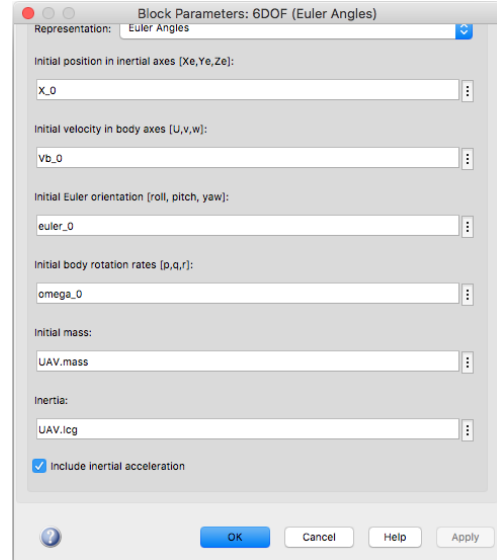


Figura C.4: Opciones dentro del bloque '6 DOF'

- **'Control Forces and Moments:'** en este subsistema se calculan las fuerzas y momentos debidas a las superficies de control. Para ello se obtienen primero las derivadas C_d , C_y , C_L , C_l , C_m y C_n y luego se calculan fuerzas y momentos con el bloque 'Aerodynamic Forces and Moments' del 'Aerospace Blockset' de Simulink. Las contribuciones de los alerones y el timón de profundidad a dichas derivadas se obtienen a partir del modelo DATCOM, sin embargo, el programa digitalDATCOM no estima dichas derivadas correspondientes al timón de dirección. Por lo tanto se ha añadido al modelo el coeficiente C_n y C_l debido a la deflexión del timón de dirección suponiendo este como una placa plana de coeficiente de sustentación ideal. Para calcular el coeficiente C_l se tiene en cuenta el brazo de momento que tiene el timón respecto al eje x de estabilidad (ver 'loadDATCOMmodel.m' D.4).

En el bloque 'MotoPropulsion' se calcula la fuerza y momentos producidos por el motor. Se ha supuesto un rendimiento propulsivo unitario y se calcula la tracción de la hélice a partir de la potencia entregada por el motor y la velocidad de vuelo. Así mismo, el momento producido se calcula idealmente como la potencia desarrollada por el motor dividido entre la velocidad angular del mismo. El modelo motopropulsivo es poco detallado y simple, sin embargo, suficiente para estudiar la dinámica de la aeronave.

Por último, dentro del sistema 'UAV' tenemos el bloque '6 DOF (Euler Angles)' perteneciente al 'Aerospace Blockset'. Este bloque integra las ecuaciones del movimiento del avión a partir de su posición y velocidad (o en su defecto posición y velocidad iniciales), y las fuerzas y momentos calculados anteriormente. Como podemos observar en la figura C.4, el bloque se inicializa con una posición, velocidad y actitud, y además es necesario añadir la masa y el tensor de inercia de la aeronave (calculados previamente en A.1) .

C.2. Subsistema 'Enviroment'

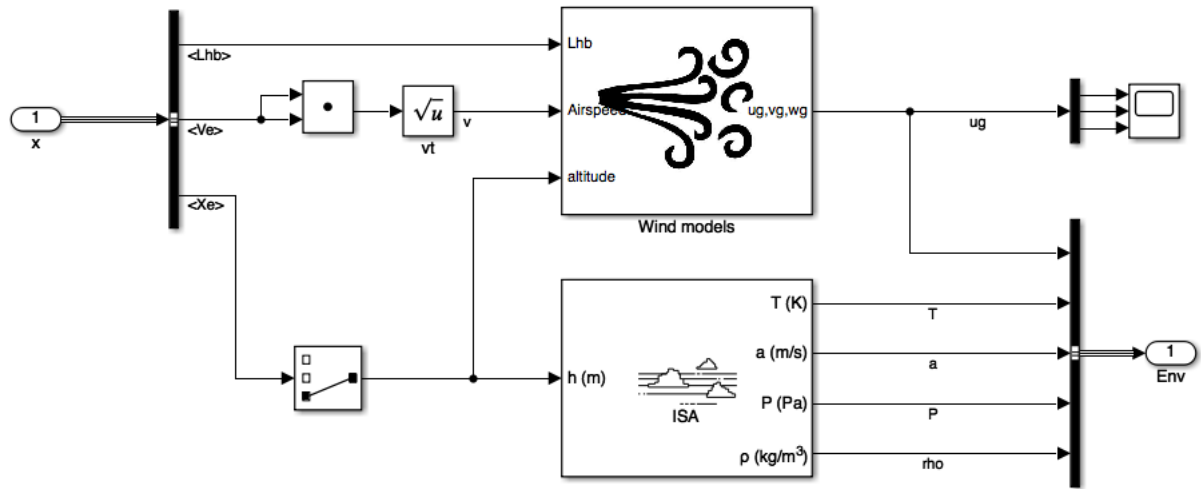


Figura C.5: Subsistema 'enviroment'

Se ha tomado como herencia del modelo [3], en él encontramos dos subsistemas:

- **'Wind models':** donde tenemos tres tipos de modelos de viento que podemos añadir a la simulación, un modelo de cortadura, un modelo de turbulencias continuas y un modelo de ráfagas discretas. Todos los modelos de viento se pueden anular con el slider que está en el modelo Simulink (ver figura C.2).
- **'ISA:'** Este bloque perteneciente al 'Aerospace Blockset', calcula temperatura, presión, densidad y velocidad del sonido según Atmósfera Estándar Internacional.

C.3. Subsistema 'Random input'

El subsistema 'Random input' solo estará activado durante la fase de linearización (ver 4.2.1 y 'Linearize.m' D.1). Sirve para que el control no sea linealmente dependiente del estado del avión y de esta forma el sistema sea observable. Para ello genera un vector de perturbación aleatorio que se suma al vector de control con una frecuencia determinada, normalmente menor al hercio (ver 'randU.m' D.5).

C.4. Subsistema 'workspace'

Este subsistema guarda en el workspace de Matlab `x_long`, `x_lat`, `u_long` y `u_lat`, es decir, las variables de estado y de control en cada momento de la simulación. Estos datos son utilizados por ejemplo para realizar la linearización de la planta (véase la sección 4.2.1 y el programa 'Linearize.m' D.1).

Este bloque se ejecuta con una frecuencia de 50 Hz.

C.5. Visores o scopes

Como se puede observar en la figura C.2 están dispuestos verticalmente una serie de visores o 'scopes' para poder observar las variables de la simulación.

C.6. Subsistema 'Controller'

Dentro de este sistema se encuentran las ganancias de realimentación de bucle cerrado del control, ya sea las del controlador clásico como las ganancias matriciales del espacio de los estados. Como se puede ver en la figura C.6, solo uno de los dos controladores realimenta el sistema, por lo que el usuario puede escoger cual de ellos está activo.

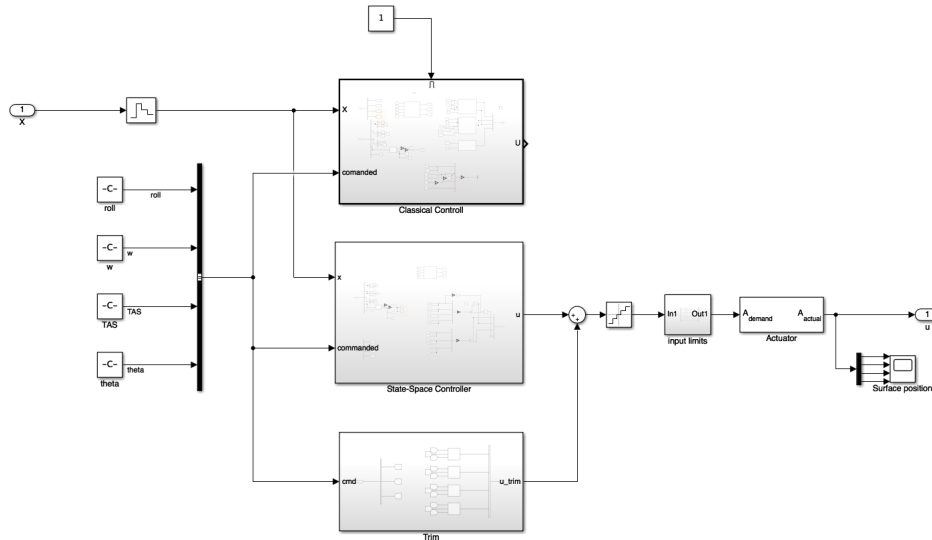


Figura C.6: Subsistema 'Controller'

También encontramos en este subsistema el bloque 'Trim', el cual dada una condición de vuelo deseada nos proporciona los inputs necesarios para mantener dicha condición en equilibrio. Ver sección 4.2.1 para más información de cómo se han calculado dichos valores. Como nota adicional mencionar que la función 'uAtTrim.m'D.4 realiza la misma labor que dicho bloque, sin embargo, debido al desarrollo del trabajo en el tiempo, primero se creó este bloque, y luego por necesidad de acceder a estos valores desde el código de Matlab se programó la función.

En la figura C.7 se puede ver el esquema interno del bloque 'Space-State controller'. En él principalmente se descompone el vector de estado en las variables longitudinales y laterales, y se multiplican por sendas ganancias longitudinales y laterales. Por último se compone el vector de mando. En este bloque podemos encontrar también un coordinador de viraje, en el cual se implementa mediante el comando de p, q y r en función del ángulo de balance y de asiento, la justificación a este coordinador se puede ver en 'Lewis' P342 [7]. También se encuentra este bloque ('Turn coordinator') en el controlador clásico.

Dentro del bloque 'Classical Controller' encontramos las ganancias del controlador clásico, calculando los input a cada mando por separado dependiendo cada uno de ciertas variables de estado. No se incluye una imagen de dicho esquema por no aportar información demasiado relevante al trabajo, sin embargo el modelo Simulink queda adjunto por lo que remitimos al lector a él para más información.

Por último, encontramos en el subsistema 'controller' un filtro de grado 0 y frecuencia de 50hz (es decir discretizamos las señales de entrada en el tiempo a dicha frecuencia), así como una discretización de la señal de salida y un modelo de actuadores. Esta modelización de los actuadores y el controlador es basta y rudimentaria, sin embargo recoge los principales problemas en la actuación real del UAV y es la actuación del controlador en tiempo discreto y la cuantización de la respuesta de los actuadores.

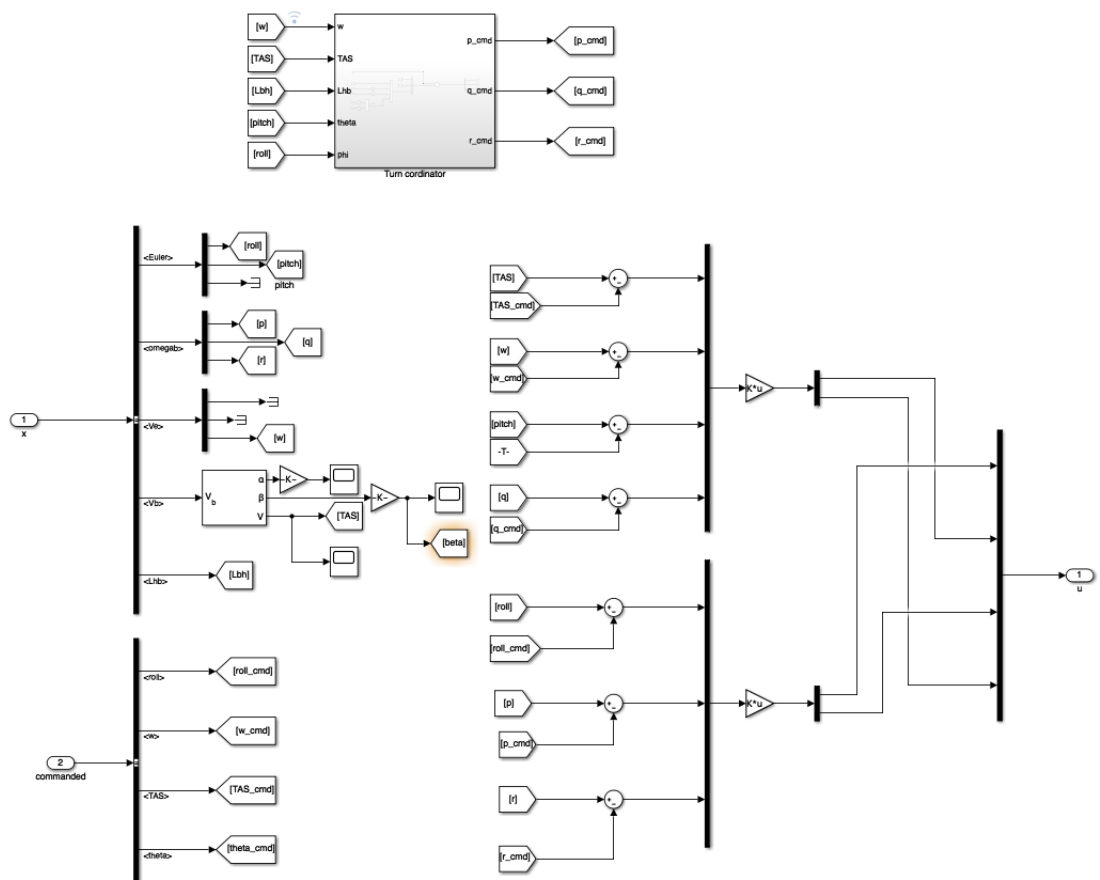


Figura C.7: Subsistema 'State-Space Controller'

Apéndice D

Archivos Matlab

D.1. Linearize.m

```
1 %=====
2 % TFG - Adaptative controll                                %
3 % 10/08/2019                                              %
4 %                                                         %
5 %                               LINEARIZE SYSTEM           %
6 %                                                         %
7 % Dom nguez Alegre, Carlos F.                             %
8 %=====
9 % Obtain a linear ss system from simulation values
10
11 clear XU_long x_long u_long
12 clear XU_lat x_lat u_lat
13
14 % INITIAL CONDITIONS
15 % must be chosen out of trim condition in order to capture de dynamic of
16 % the system
17 omega_0 = [0,0,0];
18 euler_0 = deg2rad([5,5,0]);
19 Vb_0 = [13      1    -0.15];
20 X_0 = [0 0 -100];
21
22 % DESIRED FLIGHT CONDITION
23 w_cmd = 0;
24 TAS_cmd = 12;
25 phi_cmd = 0;
26 theta_cmd = 0;
27
28
29 %-----
30 % RUN THE SIMULATION
31 % the random perturbation on imputs must be enhabled
32 sim('model_trim');
33
34
35 %-----
36 % LONGITUDINAL MODEL ESTIMATION
37
38 % TRIMING POINT FOR FLIGHT CONDITION
39 u_trim = control_at_trim(TRIM,TAS_cmd,w_cmd,phi_cmd);
40 x_trim_long = [TAS_cmd w_cmd 0 0];
41 u_trim_long = u_trim([2 4]);
42
43 % LONGITUDINAL VECTOR STATE AND DERIVATE
44 X_long = x_long.Data-x_trim_long;
45 U_long = u_long.Data-u_trim_long;
46 DX_long = diff(X_long)/0.02;
47 XU_long(:,1:4) = X_long(1:end-1,:);
48 XU_long(:,5:6) = U_long(1:end-1,:);
49
50 %Longitudinal model:
```

```

51 [A_long,B_long,cond,P_long,B_ls_long] = LSE(DX_long',XU_long',4,2);
52
53 %LQR lateral-directional weights and controller synthesis
54 Q_long = eye(4); Q_long = Q_long*10; Q_long(1,1)=0;Q_long(3,3)=0;
55 R_long = eye(2); R_long = R_long*800; R_long(2,2) = 100;
56 N_long = zeros(4,2);
57 [K_long,S,e] = lqr(A_long,B_long,Q_long,R_long,N_long)
58
59
60 %-----
61 % LATERAL-DIRECTIONAL MODEL ESTIMATION
62
63 u_trim_lat = u_trim([1 3]);
64 x_trim_lat = [phi_cmd,0,0,0];
65
66 %LATERAL VECTOR STATE AND DERIVATE
67 X_lat = x_lat.Data - x_trim_lat;
68 X_lat = X_lat(:,[1 3 4]);
69 U_lat = u_lat.Data - u_trim_lat;
70 DX_lat = diff(X_lat)/0.02;
71 XU_lat(:,1:3)=X_lat(1:end-1,:);
72 XU_lat(:,4:5)=U_lat(1:end-1,:);
73
74 %Lateral-directional model
75 [A_lat,B_lat,cond] = LSE(DX_lat',XU_lat',3,2)
76
77 %LQR lateral-directional weights and controller synthesis
78 Q_lat = eye(3);
79 Q_lat(1,1)=1000;
80 Q_lat(2,2)=10;
81 Q_lat(3,3)=10;
82 R_lat = eye(2); R_lat = R_lat*10;
83 N_lat = zeros(3,2);
84 [K_lat,S,e] = lqr(A_lat,B_lat,Q_lat,R_lat,N_lat)

```

D.2. LSE.m

```

1 function [A,Bc,condition,P,B] = LSE(DX,XU,Nx,Nu)
2 % Takes the data to be fitted into a linear model and returns the A and Bc
3 % system matrices
4
5 M = XU*XU';
6 P = M^(-1);
7 B = DX*XU';
8
9 AB = B*P;
10
11 A = AB(:,1:Nx);
12 Bc = AB(:,Nx+1:Nu+Nx);
13 condition = cond(M);
14
15 end

```

D.3. TrimUAV.m

```

1 %=====
2 % TFG - Adaptative controll %
3 % 07/08/2019 %
4 % %
5 % TRIM SYSTEM %
6 % %
7 % Domínguez Alegre, Carlos F. %
8 %=====
9 % Find trimming points on simulink model using classical controller instead
10 % of an optimization algorithm. This enables us to use the 6dof integrator
11 % from simulink with initial conditions, otherwise a model should be made
12 % without initial conditions and using directly a vector state x, this

```



```

13 % could be done getting rid of the 6 DOF module and minimizing the forces
14 % and moments.
15
16 % this simulation also proves the ability of the classical controller to
17 % return the plane to a stable state and find trimming points.
18
19 w = [1,0,-1,-2]; % vertical speeds
20 TAS = [8,10,12,14]; % absolute air speeds
21 phi = [-20,-15,-10,-5,0,5,10,15,20]; % bank angles
22 ncases = length(w)*length(TAS)*length(phi); % number of trim points
23 l = 0;
24
25 Vb_0 = [11 0.5 0.5]; %IC should be choosen wisely to avoid simulink lock out
26
27 f = waitbar(l/ncases,'progress: '); % Progress bar...
28
29 for i=1:length(w) % Loop in vertical speeds
30     for j=1:length(TAS) % Loop in TAS
31         for k=1:length(phi) % Loop in bank angle
32
33             %Find trimming point around this state:
34             sim('model_trim');
35             % Store input to system at the end of simulation
36             u_trim(i,j,k,:) = u_out.Data(end,:);
37             l = l+1;
38             waitbar(l/ncases,f,"progress: "+num2str(l) + " of "+num2str(ncases))
39             clear u_out x_out tout logouts
40             end %phi
41         end %TAS
42     end %w
43
44 % SAVE DATA TO WORKSPACE
45 save u_trim;
46 w_trim = w;
47 TRIM.TAS = TAS;
48 TRIM.phi = phi;
49
50 for i=1:length(w)
51     for j=1:length(TAS)
52         for k=1:length(phi)
53             m = length(w)+1-i;
54             TRIM.u_a(i,j,k) = u_trim(m,j,k,1);
55             TRIM.u_e(i,j,k) = u_trim(m,j,k,2);
56             TRIM.u_r(i,j,k) = u_trim(m,j,k,3);
57             TRIM.u_p(i,j,k) = u_trim(m,j,k,4);
58             TRIM.w(i) = w(m);
59         end
60     end
61 end
62
63 %=====
64 % PLOT GRAPHS
65
66 % u_p as a function of w and TAS at 0 bank
67
68 for i=1:length(w)
69     for j=1:length(TAS)
70         u_p_wTAS(i,j)=u_trim(i,j,5,4);
71     end
72 end
73 surf(w,TAS,u_p_wTAS)
74 xlabel('vertical speed (m/s)')
75 ylabel('TAS (m/s)')
76 zlabel('Throttle')
77 title('Throttle vs TAS and w (h=0)')
78
79 % u_p as a function of TAS and bank at w=0
80
81 for j=1:length(TAS)
82     for k=1:length(phi)
83         u_p_TASphi(j,k) = u_trim(2,j,k,4);
84     end

```

```

85 end
86
87 surf(phi,TAS,u_p_TASphi)
88
89
90
91 % elevator as a function of bank at w = 0 and TAS
92 for j=1:length(TAS)
93     for k=1:length(phi)
94         u_e_TASphi(j,k) = u_trim(2,j,k,2);
95     end
96 end
97 surf(phi,TAS,u_e_TASphi)
98 xlabel('\phi - bank angle (deg)')
99 ylabel('TAS (m/s)')
100 zlabel('u_e')
101 title('elevator vs TAS and \phi at w=0 (h=0)')
102
103 figure
104 plot(TAS,u_e_TASphi(:,1),'-x')
105 xlabel('TAS (m/s)')
106 ylabel('u_e')
107 title('u_e vs TAS for a level flight (w=0, h=0)')
108 grid on
109 hold on
110 plot(TAS,u_e_TASphi(:,2),'-x')
111 plot(TAS,u_e_TASphi(:,5),'-x')
112 legend("\phi=20", "\phi=15", "\phi=0")
113
114
115 % u_a vs TAS OJO! explicaci n posible -> cl_beta aumenta al aumentar alpha
116 % debido al efecto del tim n de cola
117 % Efectivamente, es debido a que el avi n realizaba un skid -> angulo de
118 % inclinaci n positivo y beta negativo a altos ngulos de ataque debido a
119 % la simplificaci n de cos(theta)?1 del controlador de viraje coordinado,
120 % se soluciona a adiando el t rminino cos(theta) tanto a r como a q,
121 % posiblemente tambi n sea necesario a adir sen(theta) en p. Ocurre
122 % solamente para ngulos de asiento grandes (20 grados en este caso)
123
124
125 for j=1:length(TAS)
126     for k=1:length(phi)
127         u_a_TASphi(j,k) = u_trim(4,j,k,1);
128     end
129 end
130 figure
131 hold on
132 grid on
133 plot(TAS,u_a_TASphi(:,1))
134 plot(TAS,u_a_TASphi(:,3))
135 plot(TAS,u_a_TASphi(:,5))
136 plot(TAS,u_a_TASphi(:,7))
137 plot(TAS,u_a_TASphi(:,9))
138
139 xlabel('TAS (m/s)')
140 ylabel('u_a')
141 legend('\phi=-20', '\phi=-10', '\phi=0', '\phi=10', '\phi=20')
142
143 title('Ailerons vs TAS for different bank angles, (w=-2 m/s h=0)')

```

D.4. uAtTrim.m

```

1 function [u_trim] = uAtTrim(TRIM,TAS,w,phi)
2 % Returns trim controls for a given flight condition
3 u_trim(1) = interp3(TRIM.TAS,TRIM.w,TRIM.phi,TRIM.u_a,TAS,w,phi);
4 u_trim(2) = interp3(TRIM.TAS,TRIM.w,TRIM.phi,TRIM.u_e,TAS,w,phi);
5 u_trim(3) = interp3(TRIM.TAS,TRIM.w,TRIM.phi,TRIM.u_r,TAS,w,phi);
6 u_trim(4) = interp3(TRIM.TAS,TRIM.w,TRIM.phi,TRIM.u_p,TAS,w,phi);
7 end

```

loadDATCOMmodel.m

```
1 %=====
2 % TFG - Control Adaptativo %
3 % 07/08/2019 %
4 % %
5 % LOAD SIM VARIABLES %
6 % %
7 % Dom nguez Alegre, Carlos F. %
8 %=====
9 %Loads all the derivatives obtained from the datcom model to the UAV struct
10 %in order for the symulink model to use it.
11
12 clear all
13 close all
14
15 raw = datcomimport("CFDA_UAV.out");
16 UAV = raw{1,1};
17
18 %cleaning data
19 n = UAV.nalpha;
20 UAV.cyb = UAV.cyb(1)*ones(1,n);
21 UAV.cnb = UAV.cnb(1)*ones(1,n);
22 UAV.clq = UAV.clq(1)*ones(1,n);
23 UAV.cmq = UAV.cmq(1)*ones(1,n);
24
25
26 %Add data from the second case (symetric deflection of flaps -> elevator)
27 UAV.dcl_sym = raw{1,2}.dcl_sym;
28 UAV.dcm_sym = raw{1,2}.dcm_sym;
29 UAV.dcdi_sym = raw{1,2}.dcdi_sym;
30 UAV.delta = raw{1,2}.delta;
31
32
33
34 % Add rudder information
35 UAV.srudder = 0.01;
36 UAV.lrudder =0.835;
37 UAV.hrudder =0.08;
38 UAV.delta_rud_max = 10;
39
40 UAV.srudder_a = UAV.srudder/UAV.sref;
41 UAV.lrudder_a = UAV.lrudder/UAV.cbar;
42
43 alpha = deg2rad(UAV.alpha);
44 h = UAV.hrudder;
45 l = UAV.lrudder;
46
47 UAV.hrudder_a = -(h*cos(alpha)-l*sin(alpha))/UAV.blref;
48
49 clear raw
50 clear n
51
52 %-----
53 % LOAD MASS PROPERTIES
54
55 UAV.Icg = load("UAV_Icg.txt");
56 UAV.mass = load("UAV_mass.txt");
57 UAV.xcg = load("UAV_Xcg.txt");
58
59
60 %-----
61 % ADD ENGINE PROPERTIES
62
63 UAV.power = 150; %Nominal power 150 watts, efficiency 0,7 aprox
64 UAV.eng_eff = 0.7;
65 UAV.rpm_max = 10000;
```

D.5. randU.m

```
1 function y = rand_u
2 y = (rand(4,1) - 0.5*ones(4,1)).*[0.1;0.1;0.1;0.2];
```

D.6. Jfunction.m

```
1 function J = Jfunction(Vx,Vu,Q,R)
2     J = 0;
3     for i=1:length(Vx)
4         x = Vx(i,:);
5         u = Vu(i,:);
6         J = J + x'*Q*x + u'*R*u;
7     end
8 end
```

D.7. RLSESIM.m

```
1 %=====
2 % TFG - Adaptative controll                                     %
3 % 09/08/2019                                                    %
4 %                                                                %
5 %                      RLSE-SELF-TUNING                          %
6 %                                                                %
7 % Dom nguez Alegre, Carlos F.                                    %
8 %=====
9
10 A = A_long;
11 B = B_long;
12 R = 1;
13 P = eye(6)*100;
14 %P = P_long;
15
16 clear X_long U_long dx_long xu_long ref mynorm
17
18 X_long = x_long.Data-x_trim_long;
19 U_long = u_long.Data-u_trim_long;
20
21 dx_long = diff(X_long)/0.02;
22 xu_long(:,1:4) = X_long(1:end-1,:);
23 xu_long(:,5:6) = U_long(1:end-1,:);
24
25
26 for i=1:length(dx_long)
27     y = dx_long(i,:);
28     x = xu_long(i,:);
29     err = y' - [A B]*x';
30     x = xu_long(i,:);
31     [A,B,P] = RLSE_klm(A,B,err',x',P,4,2,R);
32     ref(i) = B(end,1);
33 end
34
35 % Plot B(4,1) over time
36 plot(ref)
37
38
39 % Calculate new gain
40 [K_long,S,e] = lqr(A,B,Q_long,R_long,N_long)
```

D.8. RLSEklm.m

```
1 function [A,B,P] = RLSE_klm(A,B,DX_err,XU,P,Nx,Nu,R)
2     % Re calculate A and B matrices from the new data point
3     I = eye(Nx+Nu);
4
5     AB = [A B];
6     AB = AB';
7
8     K = P*XU/(XU'*P*XU + R);
9     P = (I-K*XU')*P;
10
11     AB = AB + K*(DX_err);
12
13     AB = AB';
14     A = AB(:,1:Nx);
15     B = AB(:,Nx+1:end);
16 end
```

D.9. SPSA.m

```
1 %=====
2 % TFG - Adaptative controll                                     %
3 % 19/08/2019                                                    %
4 %                                                                %
5 %                      SPSA- SELF TUNING LQR                    %
6 %                                                                %
7 % Dom nguez Alegre, Carlos F.                                   %
8 %=====
9
10
11 Q = Q_long;
12 R = R_long;
13 Q_hat = Q;
14 R_hat = R;
15
16 theta = [1 1];
17
18
19 a1 = 0.1;
20 a2 = 50;
21 alpha = 1;
22 c1 = 0.2;
23 gamma = 0.4;
24
25 Nexperiments = 10;
26 ng = 2;
27
28 floop = true;
29
30 f = waitbar(1/ncases,'progress: '); % Progress bar...
31
32 for i=1:Nexperiments
33     %a = a1/(i + 1 + a2)^alpha
34     c = c1/(i + 1)^gamma
35     g = 0;
36     for j=1:ng
37         delta = Bernoulli(2);
38         theta_p = theta + c*delta;
39         theta_m = theta - c*delta;
40
41         %Cost evaluation +
42         [Q,R] = QRTheta(theta_p);
43         [K_long,S,e] = lqr(A_long,B_long,Q,R,N_long);
44         sim('model_lqr_long');
45         u = u_long.Data;
46         x = x_long.Data;
47         J_p = J_function(x,u,Q_long,R_long);
48
```

```

49     %Cost evaluation -
50     [Q,R] = QRTheta(theta_m);
51     [K_long,S,e] = lqr(A_long,B_long,Q,R,N_long);
52     sim('model_lqr_long');
53     u = u_long.Data;
54     x = x_long.Data;
55
56     J_m = J_function(x,u,Q_long,R_long);
57
58     g = g + (J_p-J_m)./(2*c*delta)/ng
59
60     waitbar((i*ng + j -1)/(ng*Nexperiments),f,"Experiment: " + num2str(i) + "/" +
num2str(Nexperiments) + ...
61         "\n iteration: " + num2str(j) + "/" + num2str(ng))
62 end
63
64
65 if floop
66     a1 = a2*0.2/max(abs(g));
67     floop = false;
68 end
69 a = a1/(i + 1 + a2)^alpha;
70 a*g
71 theta = theta - a*g
72 mytheta(i,:) = theta;
73 err(i) = (J_p+J_m)/2;
74
75
76
77 end
78
79
80
81
82 % FUNCTIONS
83 function J = J_function(Vx,Vu,Q,R)
84     J = 0;
85     for i=1:length(Vx)
86         x = Vx(i,:)';
87         u = Vu(i,:)';
88         J = J+ x'*Q*x + u'*R*u;
89     end
90 end
91
92
93 function B = Bernoulli(n)
94 %Returns an n vector with a bernoulli distribution
95     for i=1:n
96         B(i) = binornd(1,0.5)*2-1;
97     end
98 end
99
100
101 function [Q,R] = QRTheta(theta)
102     Q(1,1) = 10;
103     Q(2,2) = 10*theta(1);
104     Q(3,3) = 5;
105     Q(4,4) = 5*theta(2);
106     R = zeros(2);
107     R(1,1) = 1000;
108     R(2,2) = 100;
109
110
111 end

```

D.10. KlongRLSE.m

```
1 classdef KlongRLSE < matlab.System & matlab.system.mixin.Propagates
2     % K_long synthesis from A and B
3
4     properties
5
6     end
7
8     properties(DiscreteState)
9
10    end
11
12    % Pre-computed constants
13    properties(Access = private)
14
15    end
16
17    methods(Access = protected)
18
19
20        % Propagation Methods
21        function n = getOutputSizeImpl(obj)
22            n = [2 4];
23        end
24
25
26        function r = isOutputFixedSizeImpl(obj)
27            r = true;
28        end
29
30
31        function r = getOutputDataTypeImpl(obj)
32            r = 'double';
33        end
34
35        function r = isOutputComplexImpl(obj)
36            r = false;
37        end
38
39
40        function setupImpl(obj)
41            % Perform one-time calculations, such as computing constants
42        end
43
44        function K_long = stepImpl(obj,A,B)
45            % Implement algorithm. Calculate y as a function of input u and
46            % discrete states.
47            Q = diag([10 10 5 10]);
48            R = diag([1000 100]);
49            N = zeros(4,2);
50
51            K_long = lqr(A,B,Q,R,N);
52        end
53
54        function resetImpl(obj)
55            % Initialize / reset discrete-state properties
56        end
57    end
58 end
```

Bibliografía

- [1] *USAF stability and control datcom*. Flight Control Division, Air Force Flight Dynamics Laboratory, 1978.
- [2] Bernoulli distribution. "https://en.wikipedia.org/wiki/Bernoulli_distribution", consulta jul. 2019.
- [3] Lightweight airplane design - matlab simulink - mathworks españa. "<https://es.mathworks.com/help/aeroblks/lightweight-airplane-design.html>", consulta jun. 2019.
- [4] Simultaneous perturbation stochastic approximation. "<https://www.jhuapl.edu/SPSA/>", consulta jul. 2019.
- [5] Spall, J. C. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE TRANSACTIONS ON AUTOMATIC CONTROL*, VOL. 37, NO.3, 1992.
- [6] Trimpe, S.; Doessegger, S. and D'Andrea, R. A self-tuning lqr approach demonstrated on an inverted pendulum. *IFAC Proceedings Volumes*, 47(3):11281–11287, 2014.
- [7] Stevens, B. ; Lewis, F. and Johnson, N. *Aircraft control and simulation: dynamics, controls design, and autonomous systems*. John Wiley & Sons, 2016.
- [8] Ashiq, M. State-space rls. *ICASSP*, 2003.
- [9] Maximize Market Research. Uav drones market – global industry analysis and forecast (2017-2026). "<https://www.maximizemarketresearch.com/market-report/uav-drones-market/2632/#details>", 2017.
- [10] Wellstead, P.E. and Zarrop, M.B. *Self-tuning systems: control and signal processing*. Wiley, 1995.
- [11] Ye, H. and Yutian, L. Online recursive closed-loop state space model identification for damping control. *2010 Asia-Pacific Power and Energy Engineering Conference*, 2010.