Escola Tècnica Superior d'Enginyeries Industrial i Aeronàutica de Terrassa
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Titulació:

INGENIERÍA AERONÁUTICA

Alumne (nom i cognoms):

Manuel Lanuza Fabregat

Títol PFC:

Diseño del modelado y control de un Quad-Rotor

Director del PFC:

Fatiha Nejjari Akhi-Elarab

Convocatòria de lliurament del PFC

Cuatrimestre de otoño – Enero 2012

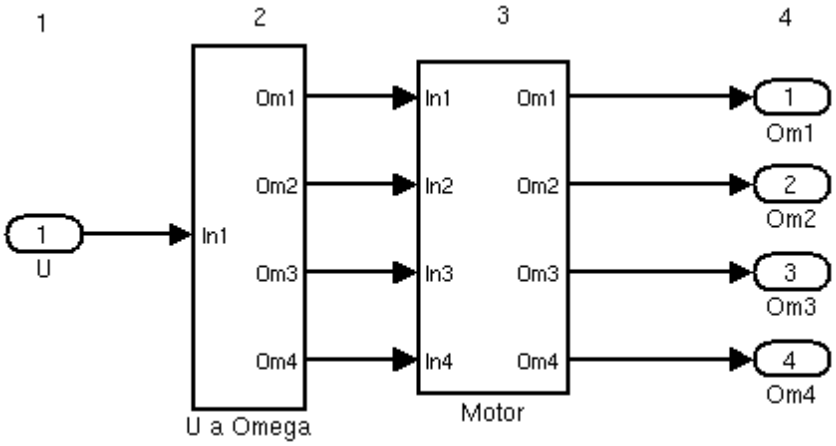
Contingut d'aquest volum: **-ANEXO-**

Índice general

1. Bloques Simulink del modelo completo

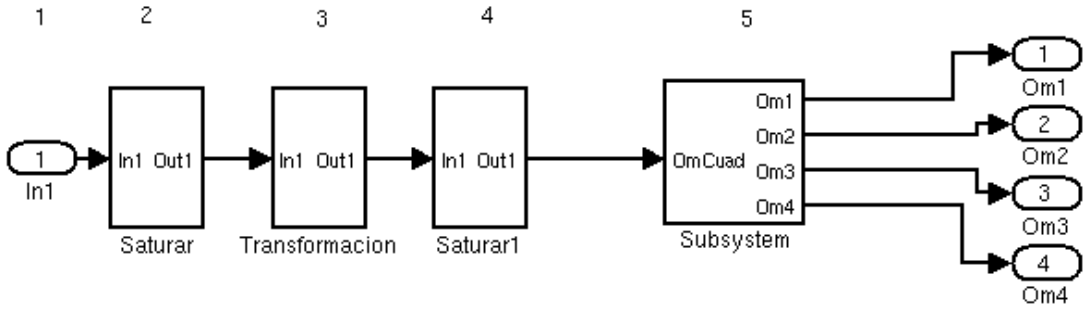
1.1 Bloque "Motor completo"

En este apartado se expondrán cada uno de los bloques internos que contienen los sistemas que forman el modelo completo del simulador. En primer lugar se tratará los bloques incluidos dentro del bloque del motor.



Dentro del bloque del motor nos encontramos cinco subsistemas diferentes:

1. Este primer bloque es la entrada al subsistema. En esta entrada se introducen los valores del vector de fuerzas de entrada U que se le requiere al sistema.
2. El bloque limitaciones se ampliará a continuación pero es el bloque en el que se aplican las distintas limitaciones fijadas por las fuerzas y momentos máximas que puede desarrollar el sistema además de transformar el vector de fuerzas U en velocidades angulares de cada uno de los rotores.
3. En este bloque se hacen pasar las velocidades angulares del motor por la función de transferencia de primer orden que simula el comportamiento del motor.
4. Es la salida del sistema que nos da las velocidades angulares que salen del bloque del motor para poder utilizarlas de entrada del sistema Quad-Rotor.

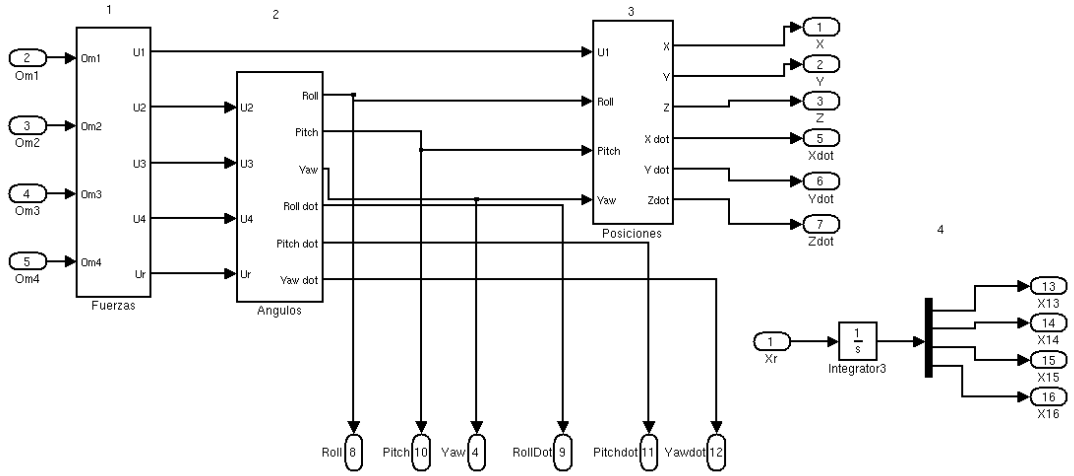


Dentro del subsistema “U a omega” encontramos los sistemas que se puede ver en la figura anterior.

1. Bloque de entrada del vector U.
2. En este bloque se limita las fuerzas que se le piden al sistema para ajustarlas a las máximas que nos puede dar con los motores trabajando al máximo.
3. Se transforman las fuerzas una vez limitadas en velocidades angulares al cuadrado.
4. Se limitan las velocidades angulares al cuadrado para que no salgan del rango entre 0-278 rad/s al cuadrado.
5. Se hace la raíz cuadrada de la señal y se separa en las cuatro velocidades angulares de los rotores.

1.2 Bloque “Quad-Rotor Dynamics”

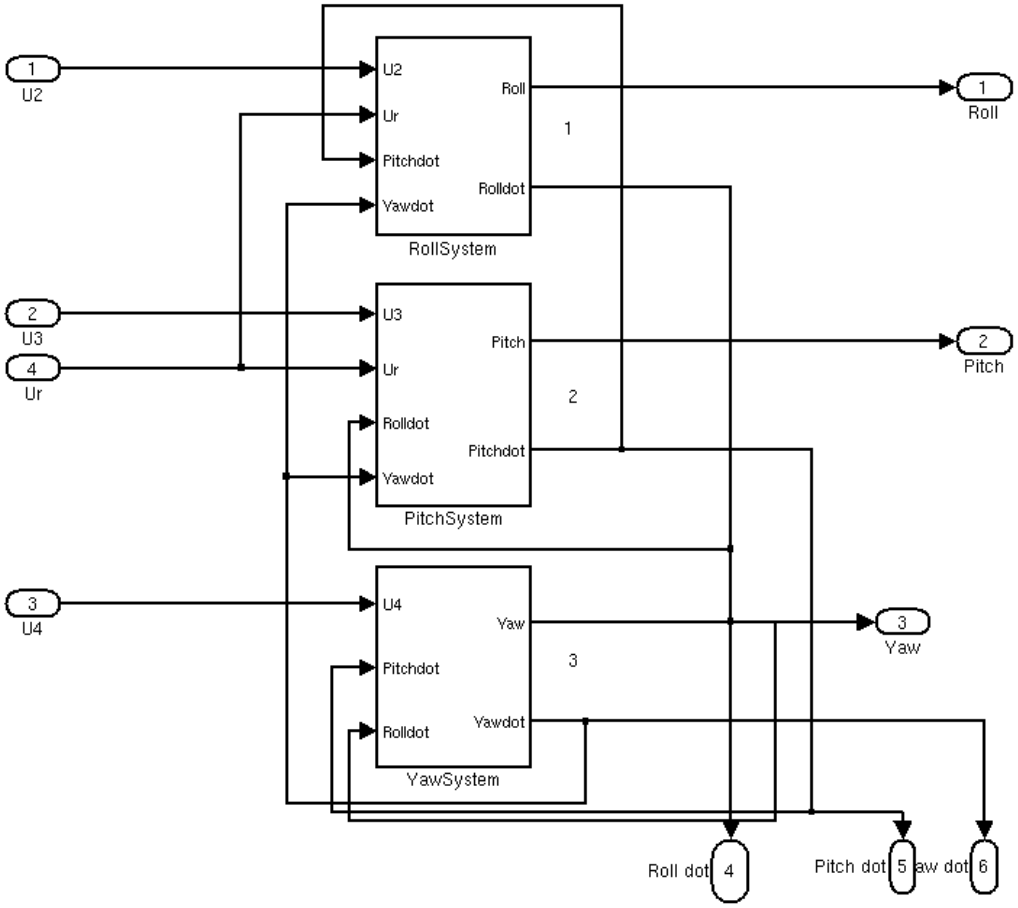
En este bloque se modela la dinámica al completo del sistema del Quad-Rotor. A continuación se mostrarán las imágenes de los subsistemas con una pequeña explicación como en el apartado anterior.



El bloque del sistema del Quad-Rotor además de los bloques de entrada que son las velocidades angulares y las salidas que son los estados del vector de estados encontramos cuatro subsistemas.

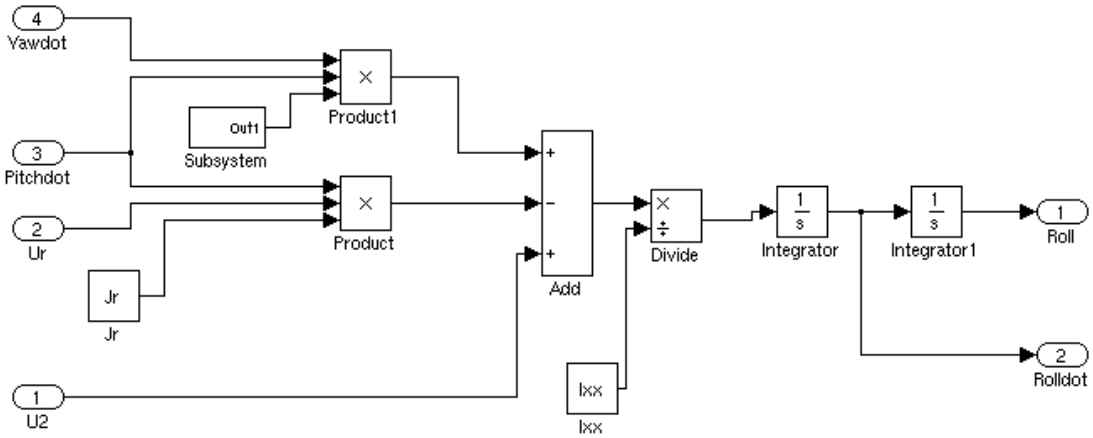
1. En este subsistema se transforman las velocidades angulares de cada uno de los rotores en la fuerza que supone para el sistema. Ya que los parámetros de entrada en las ecuaciones del movimiento son estas fuerzas.
2. En este subsistema se encuentran los distintos bloques que se ampliaran a continuación con las ecuaciones diferenciales que se definen para cada uno de los ángulos. Tiene como entradas U2, U3, U4 y el desequilibrio de velocidades angulares y como salidas los ángulos y las velocidades angulares.
3. Este subsistema es análogo al anterior pero con las posiciones y velocidades. En este caso se calcula después del bloque anterior porque además de U1 tiene como entrada el valor de los ángulos.
4. En este bloque se calcula el efecto integral en las simulaciones que lo contienen.

Dentro del bloque de ángulos encontramos:

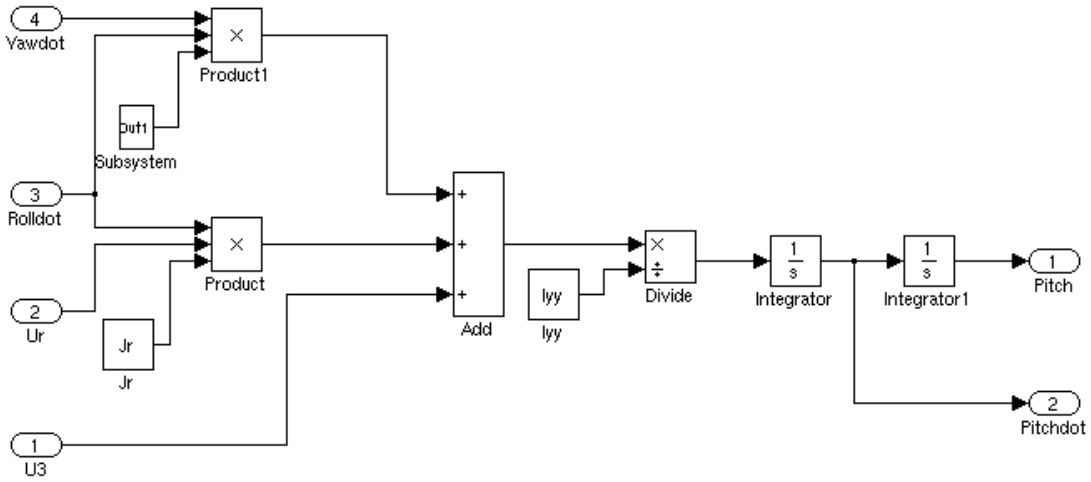


En este caso cada uno de los subsistemas contiene la ecuación diferencial expresada en forma de diagrama de bloques para da uno de los sistemas.

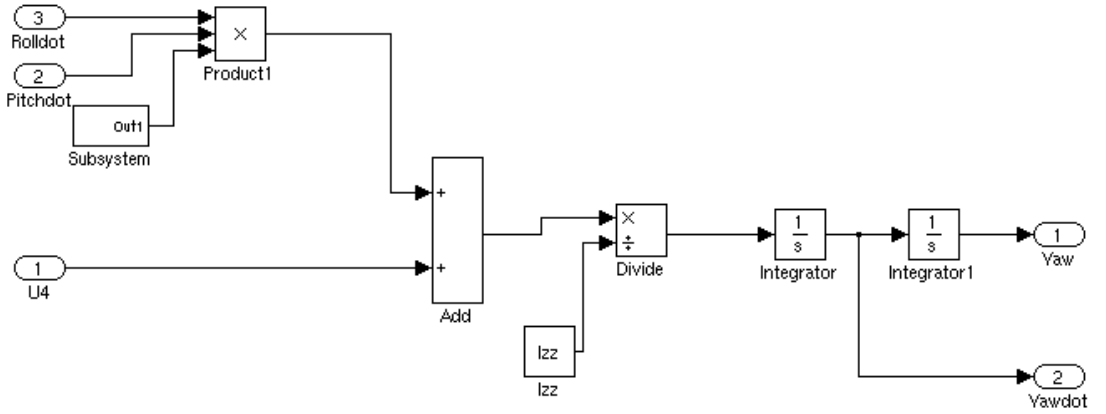
1. La ecuación diferencial para el ángulo de Roll:



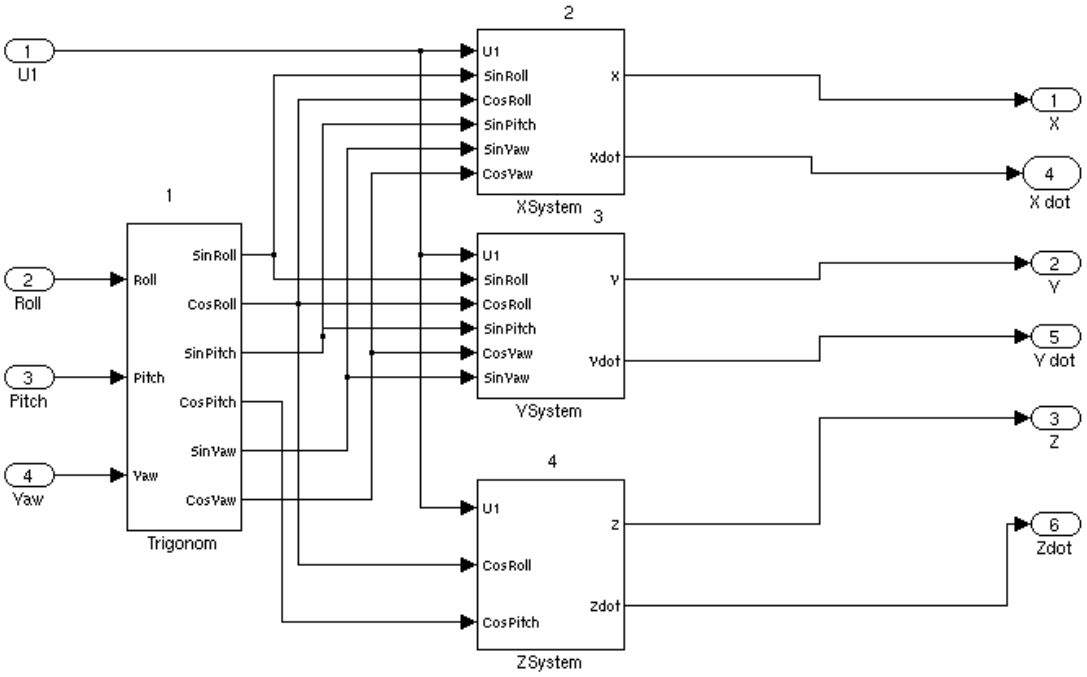
2. La ecuación diferencial para el ángulo de Pitch:



3. La ecuación diferencial para el ángulo de Yaw:

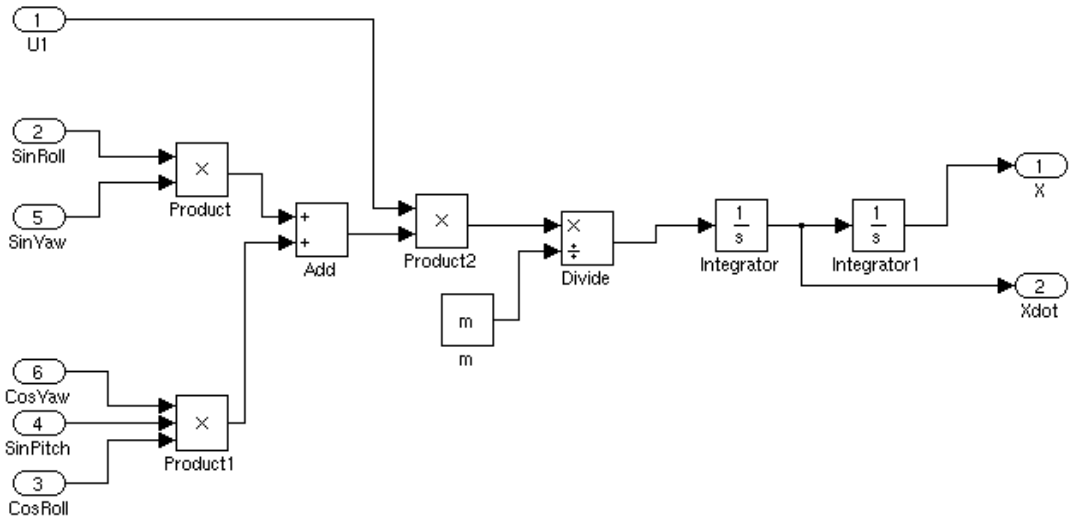


Y dentro del bloque de las posiciones encontramos:

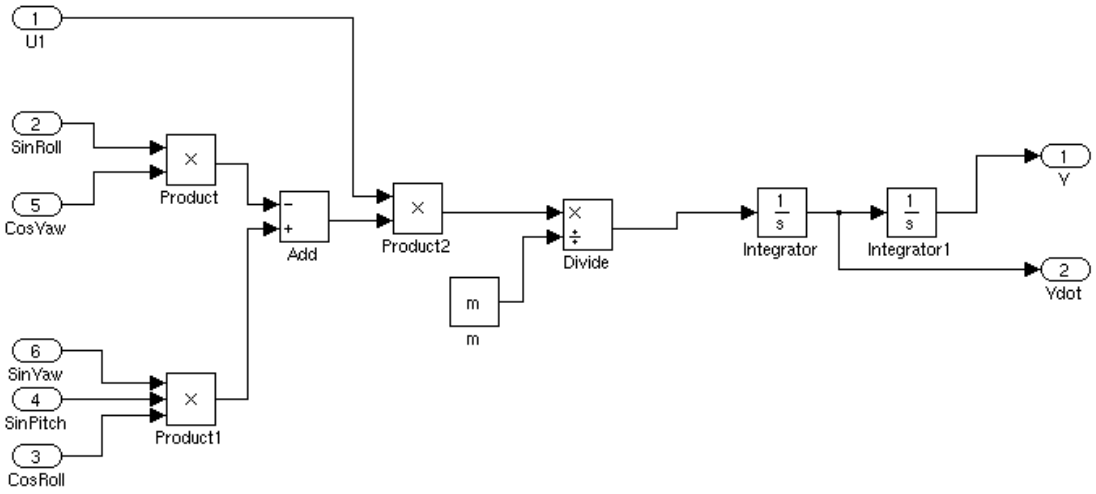


Donde se encuentran los cuatro subsistemas:

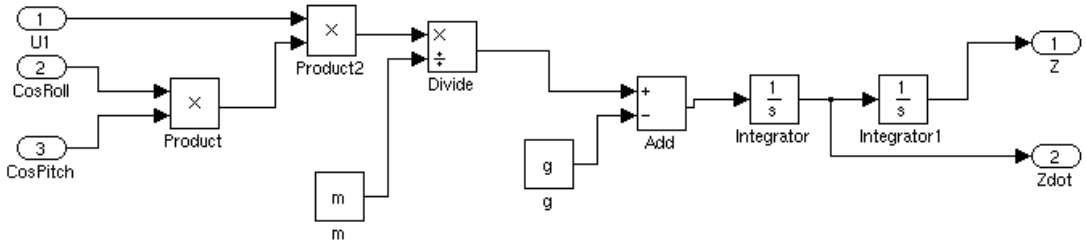
1. Se realizan los cálculos trigonométricos con los ángulos obtenidos para obtener el seno y el coseno de cada uno de ellos.
2. Bloque que contiene la ecuación diferencial en X:



3. Bloque que contiene la ecuación diferencial en Y:



4. Bloque que contiene la ecuación diferencial en Z:



2. Archivos .m programados para cada capítulo

En este apartado del anexo se pegará los códigos de los archivos .m de Matlab utilizados para las simulaciones definitivas de los distintos simuladores. El nombre de los apartados es el nombre que tiene cada uno de los programas.

2.1 Programa.m

En primer lugar se expone el código del programa que inicializa todas las variables y las matrices auxiliares:

```
% Valores de las constantes del sistema
g=9.8;
m=0.52;
Jr=8.66e-7;
Ixx=6.228e-3;
Iyy=6.225e-3;
Izz=1.121e-2;
l=0.235;
q=7.5e-7;
b=3.13e-5;
Ts=0.01;
OMmax=278;
OMmax2=OMmax*OMmax;
U1max=4*b*OMmax*OMmax;
U2max=l*b*OMmax*OMmax;
U3max=U2max;
U4max=2*q*OMmax*OMmax;

%Matrices auxiliares para pasar de U a Omega

Aux1=[b 0 0 0;
      0 l*b 0 0;
      0 0 l*b 0;
      0 0 0 q];

Aux2=[1 1 1 1;
      0 -1 0 1;
     -1 0 1 0;
     -1 1 -1 1];
inversa=inv(Aux2)*inv(Aux1);

Aux=Aux1*Aux2;


%Condiciones iniciales
```

```
x0 = [1 1 1 0.2];
```

2.2 LQR_QuadRotor.m

El siguiente código es el que se utiliza para el sistema Quad-Rotor de modelo completo controlado por LQR:

```
%Punto de equilibrio definido en vuelo a punto fijo
roll=pitch=yaw=z..=0
roll=0; rolldot=0; pitch=0; pitchdot=0; yaw=0; yawdot=0; U1=m*g;
Omega=0;

%Valores matriz A linealizada

A1=(-sin(roll)*sin(pitch)*sin(yaw)+cos(roll)*sin(yaw))*U1/m;
A2=(cos(roll)*cos(pitch)*cos(yaw))*U1/m;
A3=(-cos(roll)*sin(pitch)*sin(yaw)+sin(pitch)*cos(yaw))*U1/m;
A4=(-sin(roll)*sin(pitch)*sin(yaw)-cos(roll)*cos(yaw))*U1/m;
A5=(cos(roll)*cos(pitch)*sin(yaw))*U1/m;
A6=(cos(roll)*sin(pitch)*cos(yaw)+sin(roll)*sin(yaw))*U1/m;
A7=(-sin(roll)*cos(pitch))*U1/m;
A8=(-cos(roll)*sin(pitch))*U1/m;
A9=yawdot*((Iyy-Izz)/Ixx)-(Jr*Omega/Ixx);
A10=pitchdot*(Iyy-Izz)/Ixx;
A11=yawdot*((Izz-Ixx)/Iyy)-(Jr*Omega/Iyy);
A12=rolldot*(Izz-Ixx)/Iyy;
A13=pitchdot*(Ixx-Iyy)/Izz;
A14=rolldot*(Ixx-Iyy)/Izz;

A=[0 1 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 A1 0 A2 0 A3 0;
   0 0 0 1 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 A4 0 A5 0 A6 0;
   0 0 0 0 0 1 0 0 0 0 0 0;
   0 0 0 0 0 0 A7 0 A8 0 0 0;
   0 0 0 0 0 0 0 1 0 0 0 0;
   0 0 0 0 0 0 0 0 0 A9 0 A10;
   0 0 0 0 0 0 0 0 0 1 0 0;
   0 0 0 0 0 0 0 A11 0 0 0 A12;
   0 0 0 0 0 0 0 0 0 0 0 1;
   0 0 0 0 0 0 0 A13 0 A14 0 0];

%Matriz B linealizada

B=[0 0 0 0;
```

```
   (cos(roll)*sin(pitch)*cos(yaw)+sin(roll)*sin(yaw))/m 0 0 0;
   0 0 0 0;
   (cos(roll)*sin(pitch)*sin(yaw)-sin(roll)*cos(yaw))/m 0 0 0;
   0 0 0 0;
   (cos(roll)*cos(pitch))/m 0 0 0;
   0 0 0 0;
   0 1/Ixx 0 0;
   0 0 0 0;
   0 0 1/Iyy 0;
   0 0 0 0;
   0 0 0 1/Izz];

%Matrices C y D

C=eye(12);

D=zeros(12,4);

%Matrices Q y R

Q=[1 0 0 0 0 0 0 0 0 0 0 0;
   0 0.15 0 0 0 0 0 0 0 0 0 0;
   0 0 1 0 0 0 0 0 0 0 0 0;
   0 0 0 0.15 0 0 0 0 0 0 0 0;
   0 0 0 0 1 0 0 0 0 0 0 0;
   0 0 0 0 0 0.02 0 0 0 0 0 0;
   0 0 0 0 0 0 1 0 0 0 0 0;
   0 0 0 0 0 0 0 0.12 0 0 0 0;
   0 0 0 0 0 0 0 0 1 0 0 0;
   0 0 0 0 0 0 0 0 0 0.12 0 0;
   0 0 0 0 0 0 0 0 0 0 1 0;
   0 0 0 0 0 0 0 0 0 0 0 0.01];


 R=[0.00001 0 0 0;
    0 0.1 0 0;
    0 0 0.1 0;
    0 0 0 0.01];

%Calculo de la matriz K del LQR

[K]=lqr(A,B,Q,R);
```

2.3 LQR_integral.m

El siguiente es el programa que se ejecuta cuando para añadir el efecto integral. Es completamente análogo al programa anterior pero añadiendo más filas y columnas

en las matrices del sistema. Se muestra este pero de ahora en adelante simplemente se realizará de da por entendido que se realiza de manera análoga.

```
%Punto de equilibrio definido en vuelo a punto fijo
roll=pitch=yaw=z..=0
roll=0; rolldot=0; pitch=0; pitchdot=0; yaw=0; yawdot=0; U1=m*g;
Omega=0;

A1=(-sin(roll)*sin(pitch)*sin(yaw)+cos(roll)*sin(yaw))*U1/m;
A2=(cos(roll)*cos(pitch)*cos(yaw))*U1/m;
A3=(-cos(roll)*sin(pitch)*sin(yaw)+sin(pitch)*cos(yaw))*U1/m;
A4=(-sin(roll)*sin(pitch)*sin(yaw)-cos(roll)*cos(yaw))*U1/m;
A5=(cos(roll)*cos(pitch)*sin(yaw))*U1/m;
A6=(cos(roll)*sin(pitch)*cos(yaw)+sin(roll)*sin(yaw))*U1/m;
A7=(-sin(roll)*cos(pitch))*U1/m;
A8=(-cos(roll)*sin(pitch))*U1/m;
A9=yawdot*((Iyy-Izz)/Ixx)-(Jr*Omega/Ixx);
A10=pitchdot*(Iyy-Izz)/Ixx;
A11=yawdot*((Izz-Ixx)/Iyy)-(Jr*Omega/Iyy);
A12=rolldot*(Izz-Ixx)/Iyy;
A13=pitchdot*(Ixx-Iyy)/Izz;
A14=rolldot*(Ixx-Iyy)/Izz;

A=[0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 A1 0 A2 0 A3 0 0 0 0 0;
   0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 A4 0 A5 0 A6 0 0 0 0 0;
   0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 A7 0 A8 0 0 0 0 0 0 0;
   0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 0 0 0 A9 0 A10 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0;
   0 0 0 0 0 0 0 A11 0 0 0 A12 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0;
   0 0 0 0 0 0 0 A13 0 A14 0 0 0 0 0 0;
   1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0];


B=[0 0 0 0;
   (cos(roll)*sin(pitch)*cos(yaw)+sin(roll)*sin(yaw))/m 0 0 0;
   0 0 0 0;
   (cos(roll)*sin(pitch)*sin(yaw)-sin(roll)*cos(yaw))/m 0 0 0;
   0 0 0 0;
   (cos(roll)*cos(pitch))/m 0 0 0;
   0 0 0 0;
   0 1/Ixx 0 0;
   0 0 0 0;
   0 0 1/Iyy 0;
   0 0 0 0;
   0 0 0 1/Izz
   0 0 0 0
```

```
   0 0 0 0
   0 0 0 0
   0 0 0 0];

C=eye(16);

D=zeros(16,4);


Q=[1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
   0 0.5 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0.5 0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0; %Z
   0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 0 0.01 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0.01 0 0 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0 0 0.01 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0 0 0 0.05 0 0 0;
   0 0 0 0 0 0 0 0 0 0 0 0 0 0.09 0 0;
   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.01 0;  %Z
   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.05];


 R=[0.001 0 0 0;
    0 0.01 0 0;
    0 0 0.01 0;
    0 0 0 0.005];

[K]=lqr(A,B,Q,R);
```

2.4 LQR_QuadRotor_Obs.m

En este caso es el programa utilizado para los cálculos relacionados con el sistema con el observador integrado.

```
%Punto de equilibrio definido en vuelo a punto fijo
roll=pitch=yaw=z..=0
roll=0; rolldot=0; pitch=0; pitchdot=0; yaw=0; yawdot=0; U1=m*g;
Omega=0;

%Matriz A

A1=(-sin(roll)*sin(pitch)*sin(yaw)+cos(roll)*sin(yaw))*U1/m;
```

```
A2=(cos(roll)*cos(pitch)*cos(yaw))*U1/m;
A3=(-cos(roll)*sin(pitch)*sin(yaw)+sin(pitch)*cos(yaw))*U1/m;
A4=(-sin(roll)*sin(pitch)*sin(yaw)-cos(roll)*cos(yaw))*U1/m;
A5=(cos(roll)*cos(pitch)*sin(yaw))*U1/m;
A6=(cos(roll)*sin(pitch)*cos(yaw)+sin(roll)*sin(yaw))*U1/m;
A7=(-sin(roll)*cos(pitch))*U1/m;
A8=(-cos(roll)*sin(pitch))*U1/m;
A9=yawdot*((Iyy-Izz)/Ixx)-(Jr*Omega/Ixx);
A10=pitchdot*(Iyy-Izz)/Ixx;
A11=yawdot*((Izz-Ixx)/Iyy)-(Jr*Omega/Iyy);
A12=rolldot*(Izz-Ixx)/Iyy;
A13=pitchdot*(Ixx-Iyy)/Izz;
A14=rolldot*(Ixx-Iyy)/Izz;

A=[0 1 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 A1 0 A2 0 A3 0;
   0 0 0 1 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 A4 0 A5 0 A6 0;
   0 0 0 0 0 1 0 0 0 0 0 0;
   0 0 0 0 0 0 A7 0 A8 0 0 0;
   0 0 0 0 0 0 0 1 0 0 0 0;
   0 0 0 0 0 0 0 0 0 A9 0 A10;
   0 0 0 0 0 0 0 0 0 0 1 0;
   0 0 0 0 0 0 0 A11 0 0 0 A12;
   0 0 0 0 0 0 0 0 0 0 0 1;
   0 0 0 0 0 0 0 A13 0 A14 0 0];

%Matriz B

B=[0 0 0 0;
   (cos(roll)*sin(pitch)*cos(yaw)+sin(roll)*sin(yaw))/m 0 0 0;
   0 0 0 0;
   (cos(roll)*sin(pitch)*sin(yaw)-sin(roll)*cos(yaw))/m 0 0 0;
   0 0 0 0;
   (cos(roll)*cos(pitch))/m 0 0 0;
   0 0 0 0;
   0 1/Ixx 0 0;
   0 0 0 0;
   0 0 1/Iyy 0;
   0 0 0 0;
   0 0 0 1/Izz];

%Matriz C y D

C=[1 0 0 0 0 0 0 0 0 0 0 0;
   0 0 1 0 0 0 0 0 0 0 0 0;
   0 0 0 0 1 0 0 0 0 0 0 0;
   0 0 0 0 0 0 1 0 0 0 0 0;
   0 0 0 0 0 0 0 1 0 0 0 0;
   0 0 0 0 0 0 0 0 1 0 0 0;
   0 0 0 0 0 0 0 0 0 1 0 0;
   0 0 0 0 0 0 0 0 0 0 1 0;
   0 0 0 0 0 0 0 0 0 0 0 1];

D=zeros(12,4);
```

```
%Matrices Q y R del controlador

Q=[1 0 0 0 0 0 0 0 0 0 0 0;
   0 0.15 0 0 0 0 0 0 0 0 0 0;
   0 0 1 0 0 0 0 0 0 0 0 0;
   0 0 0 0.15 0 0 0 0 0 0 0 0;
   0 0 0 0 1 0 0 0 0 0 0 0;
   0 0 0 0 0 0.2 0 0 0 0 0 0;
   0 0 0 0 0 0 1 0 0 0 0 0;
   0 0 0 0 0 0 0 0.12 0 0 0 0;
   0 0 0 0 0 0 0 0 1 0 0 0;
   0 0 0 0 0 0 0 0 0 0.12 0 0;
   0 0 0 0 0 0 0 0 0 0 1 0;
   0 0 0 0 0 0 0 0 0 0 0 0.01];

 R=[0.1 0 0 0;
    0 0.1 0 0;
    0 0 0.1 0;
    0 0 0 0.1];

%Calculo matriz K del controlador

[K]=lqr(A,B,Q,R);

%Calculo matriz de Observabilidad

Obs=[C; C*A; C*A^2;C*A^3;C*A^4;C*A^5;C*A^6;
    C*A^7;C*A^8;C*A^9;C*A^10;C*A^11];

%ObtenciÛn rango de la matriz de observabilidad

Rank(Obs)



%DefiniciÛn de los polos

Polos=[-10 -120 -5 -110 -4.9 -106 -200 -100 -100 -110 -111 -11];

%Pole Placement

H=place(A', C', Polos)';

%Matriz de realimentaciÛn del observador

Func=A-H*C;
```

2.5 LQR_QuadRotor_Noise.m

Para finalizar se muestra el código para el sistema con ruido y el filtro de Kalman implementado.

```
%Punto de equilibrio definido en vuelo a punto fijo
roll=pitch=yaw=z..=0
roll=0; rolldot=0; pitch=0; pitchdot=0; yaw=0; yawdot=0; U1=m*g;
Omega=0;

%Matriz A

A1=(-sin(roll)*sin(pitch)*sin(yaw)+cos(roll)*sin(yaw))*U1/m;
A2=(cos(roll)*cos(pitch)*cos(yaw))*U1/m;
A3=(-cos(roll)*sin(pitch)*sin(yaw)+sin(pitch)*cos(yaw))*U1/m;
A4=(-sin(roll)*sin(pitch)*sin(yaw)-cos(roll)*cos(yaw))*U1/m;
A5=(cos(roll)*cos(pitch)*sin(yaw))*U1/m;
A6=(cos(roll)*sin(pitch)*cos(yaw)+sin(roll)*sin(yaw))*U1/m;
A7=(-sin(roll)*cos(pitch))*U1/m;
A8=(-cos(roll)*sin(pitch))*U1/m;
A9=yawdot*((Iyy-Izz)/Ixx)-(Jr*Omega/Ixx);
A10=pitchdot*(Iyy-Izz)/Ixx;
A11=yawdot*((Izz-Ixx)/Iyy)-(Jr*Omega/Iyy);
A12=rolldot*(Izz-Ixx)/Iyy;
A13=pitchdot*(Ixx-Iyy)/Izz;
A14=rolldot*(Ixx-Iyy)/Izz;

A=[0 1 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 A1 0 A2 0 A3 0;
   0 0 0 1 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 A4 0 A5 0 A6 0;
   0 0 0 0 0 1 0 0 0 0 0 0;
   0 0 0 0 0 0 A7 0 A8 0 0 0;
   0 0 0 0 0 0 0 1 0 0 0 0;
   0 0 0 0 0 0 0 0 0 A9 0 A10;
   0 0 0 0 0 0 0 0 0 1 0 0;
   0 0 0 0 0 0 0 A11 0 0 0 A12;
   0 0 0 0 0 0 0 0 0 0 0 1;
   0 0 0 0 0 0 0 A13 0 A14 0 0];

%Matriz B

B=[0 0 0 0;
   (cos(roll)*sin(pitch)*cos(yaw)+sin(roll)*sin(yaw))/m 0 0 0;
   0 0 0 0;
   (cos(roll)*sin(pitch)*sin(yaw)-sin(roll)*cos(yaw))/m 0 0 0;
   0 0 0 0;
   (cos(roll)*cos(pitch))/m 0 0 0;
   0 0 0 0;
   0 1/Ixx 0 0;
   0 0 0 0;
   0 0 1/Iyy 0;
   0 0 0 0;
```

```
   0 0 0 1/Izz];

%Matriz C, D y G del sistema con ruido

C=[1 0 0 0 0 0 0 0 0 0 0 0;
   0 0 1 0 0 0 0 0 0 0 0 0;
   0 0 0 0 1 0 0 0 0 0 0 0;
   0 0 0 0 0 0 1 0 0 0 0 0;
   0 0 0 0 0 0 0 1 0 0 0 0;
   0 0 0 0 0 0 0 0 1 0 0 0;
   0 0 0 0 0 0 0 0 0 1 0 0;
   0 0 0 0 0 0 0 0 0 0 1 0;
   0 0 0 0 0 0 0 0 0 0 0 1];

D=zeros(12,4);

G=[1 0 0 0 0 0 0 0 0 0 0 0;
   0 1 0 0 0 0 0 0 0 0 0 0;
   0 0 1 0 0 0 0 0 0 0 0 0;
   0 0 0 1 0 0 0 0 0 0 0 0;
   0 0 0 0 1 0 0 0 0 0 0 0;
   0 0 0 0 0 1 0 0 0 0 0 0;
   0 0 0 0 0 0 1 0 0 0 0 0;
   0 0 0 0 0 0 0 1 0 0 0 0;
   0 0 0 0 0 0 0 0 1 0 0 0;
   0 0 0 0 0 0 0 0 0 1 0 0;
   0 0 0 0 0 0 0 0 0 0 1 0;
   0 0 0 0 0 0 0 0 0 0 0 1];

%Matrices Q y R del controlador

Q=[1 0 0 0 0 0 0 0 0 0 0 0;
   0 5 0 0 0 0 0 0 0 0 0 0;
   0 0 1 0 0 0 0 0 0 0 0 0;
   0 0 0 0.1 0 0 0 0 0 0 0 0;
   0 0 0 0 1 0 0 0 0 0 0 0;
   0 0 0 0 0 0.8 0 0 0 0 0 0;
   0 0 0 0 0 0 1 0 0 0 0 0;
   0 0 0 0 0 0 0 2.1 0 0 0 0;
   0 0 0 0 0 0 0 0 1 0 0 0;
   0 0 0 0 0 0 0 0 0 0.2 0 0;
   0 0 0 0 0 0 0 0 0 0 1 0;
   0 0 0 0 0 0 0 0 0 0 0 0.2];

 R=[0.1 0 0 0;
    0 0.9 0 0;
    0 0 0.9 0;
    0 0 0 1];

%Matriz K del controlador

[K]=lqr(A,B,Q,R);

%Observabilidad
```

```
Obs=[C; C*A; C*A^2;C*A^3;C*A^4;C*A^5;C*A^6;
   C*A^7;C*A^8;C*A^9;C*A^10;C*A^11];

%Matriz de covarianza del ruido de planta

Noise=[0.0001 0 0 0 0 0 0 0 0 0 0 0;
       0 0.00015 0 0 0 0 0 0 0 0 0 0;
       0 0 0.00011 0 0 0 0 0 0 0 0 0;
       0 0 0 0.00016 0 0 0 0 0 0 0 0;
       0 0 0 0 0.00012 0 0 0 0 0 0 0;
       0 0 0 0 0 0.00013 0 0 0 0 0 0;
       0 0 0 0 0 0 0.000014 0 0 0 0 0;
       0 0 0 0 0 0 0 0.000017 0 0 0 0;
       0 0 0 0 0 0 0 0 0.000018 0 0 0;
       0 0 0 0 0 0 0 0 0 0.000019 0 0;
       0 0 0 0 0 0 0 0 0 0 0.000020 0;
       0 0 0 0 0 0 0 0 0 0 0 0.000021];

%Matriz de covarianza del ruido de medicion

Noise2=[0.00011 0 0 0 0 0 0 0 0 0 0 0;
        0 0.000111 0 0 0 0 0 0 0 0 0 0;
        0 0 0.000121 0 0 0 0 0 0 0 0 0;
        0 0 0 0.0000141 0 0 0 0 0 0 0 0;
        0 0 0 0 0.0000171 0 0 0 0 0 0 0;
        0 0 0 0 0 0.0000181 0 0 0 0 0 0;
        0 0 0 0 0 0 0.0000191 0 0 0 0 0;
        0 0 0 0 0 0 0 0.0000210 0 0 0 0;
        0 0 0 0 0 0 0 0 0.0000211 0 0 0;
        0 0 0 0 0 0 0 0 0 0.0000211 0 0;
        0 0 0 0 0 0 0 0 0 0 0.0000211 0;
        0 0 0 0 0 0 0 0 0 0 0 0.0000211];

%Matriz N

N=[0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0 0 0;
   0 0 0 0 0 0 0 0 0 0 0 0];

%Calculo de la matriz H del filtro de Kalman

[H] = lqe(A,G,C,Noise,Noise2,N);
```