Anexo

Aquí se incluye la programación en Modelica de todas las partes y módulos que se han descrito durante el trabajo.

Modelo del cuadricóptero

```
model Quadcopter
  //
  //INPUTS & OUTPUTS
  Modelica.Blocks.Interfaces.RealInput InV1 annotation(
    Placement(visible = true, transformation(origin = {-
120, 74}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {-120, 74}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput InV2 annotation(
    Placement(visible = true, transformation(origin = {-
120, 24}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {-120, 24}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput InV3 annotation(
    Placement(visible = true, transformation(origin = {-
120, -20}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {-120, -20}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput InV4 annotation(
    Placement(visible = true, transformation(origin = {-
120, -66}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {-120, -66}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput OutW[4] annotation(
    Placement(visible = true, transformation(origin = {114,
-80}, extent = {{-14, -14}, {14, 14}}, rotation = 0),
iconTransformation(origin = {114, -80}, extent = {{-14, -
14}, {14, 14}}, rotation = 0)));
  Modelica.Electrical.Analog.Sources.SignalVoltage V1
annotation(
    Placement(visible = true, transformation(origin = {-64,
74}, extent = {{-18, -18}, {18, 18}}, rotation = 90)));
  Modelica.Electrical.Analog.Sources.SignalVoltage V2
annotation(
    Placement(visible = true, transformation(origin = {-64,
24}, extent = {{-18, -18}, {18, 18}}, rotation = 90)));
  Modelica.Electrical.Analog.Sources.SignalVoltage V3
annotation(
    Placement(visible = true, transformation(origin = {-64,
-20}, extent = {{-18, -18}, {18, 18}}, rotation = 90)));
```

```
  Modelica.Electrical.Analog.Sources.SignalVoltage V4
annotation(
    Placement(visible = true, transformation(origin = {-64,
-66}, extent = {{-18, -18}, {18, 18}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealOutput OutPos[3]
annotation(
    Placement(visible = true, transformation(origin = {115,
65}, extent = {{-15, -15}, {15, 15}}, rotation = 0),
iconTransformation(origin = {115, 65}, extent = {{-15, -
15}, {15, 15}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput OutAcc[3]
annotation(
    Placement(visible = true, transformation(origin = {114,
28}, extent = {{-14, -14}, {14, 14}}, rotation = 0),
iconTransformation(origin = {114, 28}, extent = {{-14, -
14}, {14, 14}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput OutVel[3]
annotation(
    Placement(visible = true, transformation(origin = {114,
46}, extent = {{-14, -14}, {14, 14}}, rotation = 0),
iconTransformation(origin = {114, 46}, extent = {{-14, -
14}, {14, 14}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput OutOri[3]
annotation(
    Placement(visible = true, transformation(origin = {115,
-13}, extent = {{-15, -15}, {15, 15}}, rotation = 0),
iconTransformation(origin = {115, -13}, extent = {{-15, -
15}, {15, 15}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput OutAngVel[3]
annotation(
    Placement(visible = true, transformation(origin = {115,
-33}, extent = {{-15, -15}, {15, 15}}, rotation = 0),
iconTransformation(origin = {115, -33}, extent = {{-15, -
15}, {15, 15}}, rotation = 0)));
  Modelica.Electrical.Analog.Basic.Ground G1 annotation(
    Placement(visible = true, transformation(origin = {-22,
76}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Electrical.Analog.Basic.Ground G2 annotation(
    Placement(visible = true, transformation(origin = {-22,
32}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Electrical.Analog.Basic.Ground G3 annotation(
    Placement(visible = true, transformation(origin = {-20,
-16}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Electrical.Analog.Basic.Ground G4 annotation(
    Placement(visible = true, transformation(origin = {-20,
-62}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  //
  //IMPORTS
  import SI = Modelica.SIunits;
  import Math = Modelica.Math;
```

```
  //
  //TYPES
  type ThrustConstant = Real(unit = "N.s2/rad2");
  type DragConstant = Real(unit = "N.m.s2/rad2");
  type MechanicalConstant = Real(unit = "N.m/A");
  type ElectricalConstant = Real(unit = "V.s/rad");
  type SquaredAngularVelocity = Real(unit = "rad2/s2");
  type Angle = Real(unit = "rad");
  //
  //PARAMETERS (all units in SI)
  parameter SI.Mass Mc = 0.8211 "Plate mass / Central
mass";
  parameter SI.Mass Mm = 0.1247 "Motor mass";
  parameter SI.Mass Mt = Mc + 4 * Mm "Total mass: 1.3199";
  parameter SI.Mass Mr = 0.05161 "Rotor mass";
  parameter SI.Length Lr = 0.1 "Rotor length";
  parameter SI.Length La = 0.211 "Arm length";
  parameter SI.Length Rp = 0.07187 "Plate (centre) radius";
  parameter SI.Inertia Ir = Mr * Lr ^ 2 / 12 "Moment of
inertia of a rotor: 4.300833*10^-7";
  parameter SI.Inertia Ix = 2 * Mc * Rp ^ 2 / 5 + 2 * Mm *
La ^ 2 "Moment of inertia of the quadcopter around the X
axis: 0.0128";
  parameter SI.Inertia Iy = Ix "Moment of inertia of the
quadcopter around the Y axis";
  parameter SI.Inertia Iz = 2 * Mc * Rp ^ 2 / 5 + 4 * Mm *
La ^ 2 "Moment of inertia of the quadcopter around the Z
axis: 0.0239036";
  parameter SI.Inertia J[3, 3] = [Ix, 0, 0; 0, Iy, 0; 0, 0,
Iz] "Inertia tensor of the quadcopter";
  parameter SI.Inertia invJ[3, 3] = [1 / Ix, 0, 0; 0, 1 /
Iy, 0; 0, 0, 1 / Iz] "Inverse of the inertia tensor";
  parameter SI.Acceleration g = 9.81 "Gravity acceleration
on Earth";
  parameter ThrustConstant TC = 9.9865 * 10 ^ (-6) "Thrust
constant";
  parameter DragConstant DC = 1.5978 * 10 ^ (-7) "Drag
constant";
  parameter MechanicalConstant Km = 0.01 "Mechanical
constant of each motor";
  parameter ElectricalConstant Ke = 0.01 "Electrical
constant of each motor";
  parameter SI.Resistance Res = 0.1107 "Armature resistance
of each motor";
  parameter SI.Voltage Vmax = 11.1 "Maximum armature
voltage for each motor";
  parameter Real A[3, 4] = [TC * La, 0, -TC * La, 0; 0, -TC
* La, 0, TC * La; -DC, DC, -DC, DC] "Auxiliar matrix of
thrust and drag constants";
  //
```

```
  //Initial values
  parameter SI.Length x0 = 0 "Initial value for the
position on the X axis";
  parameter SI.Length y0 = 0 "Initial value for the
position on the Y axis";
  parameter SI.Length z0 = 0 "Initial value for the
position on the Z axis";
  parameter Angle phi0 = 0 "Initial value for the
orientation in X axis";
  parameter Angle theta0 = 0 "Initial value for the
orientation in Y axis";
  parameter Angle psi0 = 0 "Initial value for the
orientation in Z axis";
  parameter SI.AngularVelocity w10 = 569.3356 "Value of the
angular velocity of rotor 1 for hovering";
  parameter SI.AngularVelocity w20 = 569.3356 "Value of the
angular velocity of rotor 2 for hovering";
  parameter SI.AngularVelocity w30 = 569.3356 "Value of the
angular velocity of rotor 3 for hovering";
  parameter SI.AngularVelocity w40 = 569.3356 "Value of the
angular velocity of rotor 4 for hovering";
  //
  //VARIABLES
  //Quadrotor kinematics
  SI.Length x[3, 1] "Position vector in the world
reference";
  SI.Velocity derX[3, 1] "Velocities vector in the world
reference";
  Angle phi, theta, psi "Orientation in X, Y and Z axis
(resp.) in the world reference";
  Angle OriVec[3, 1] = [phi; theta; psi] "Orientation
vector in the world reference";
  SI.Velocity v[3, 1] "Velocities vector in the robot
reference";
  SI.AngularVelocity P, Q, R "Angular velocity around the
X, Y and Z axis (resp.) in the robot reference";
  SI.AngularVelocity AngVel[3, 1] = [P; Q; R] "Angular
velocity vector in the robot reference";
  SI.AngularVelocity S[3, 3] = [0, -R, Q; R, 0, -P; -Q, P,
0] "Auxiliar matrix";
  //
  //Motors kinematics
  SI.AngularVelocity w1, w2, w3, w4 "Angular velocity of
rotor 1, 2, 3 and 4 (resp.)";
  SI.AngularVelocity Omega[1, 4] = [w1, w2, w3, w4] "Vector
of the angular velocities of the rotors";
  SquaredAngularVelocity SqOmega[1, 4] = [w1 ^ 2, w2 ^ 2,
w3 ^ 2, w4 ^ 2] "Vector of the squared angular velocities";
  //
  //Rotation matrixes
```

```
  Real Rot[3, 3] = [cos(psi) * cos(theta), cos(psi) *
sin(theta) * sin(phi) - sin(psi) * cos(phi), cos(psi) *
sin(theta) * cos(phi) + sin(psi) * sin(phi); sin(psi) *
cos(theta), sin(psi) * sin(theta) * sin(phi) + cos(psi) *
cos(phi), sin(psi) * sin(theta) * cos(phi) - cos(psi) *
sin(phi); -sin(theta), cos(theta) * sin(phi), cos(theta) *
cos(phi)] "Rotation matrix from v to der(x)";
  Real RotA[3, 3] = [1, sin(phi) * tan(theta), cos(phi) *
tan(theta); 0, cos(phi), -sin(phi); 0, sin(phi) /
cos(theta), cos(phi) / cos(theta)] "Rotation matrix from
AngVel to der(OriVec)";
  //
  //Quadrotor dynamics
  SI.AngularVelocity Gyro = (-w1) + w2 - w3 + w4
"Gyroscopic term";
  SI.Force Thrust = TC * (w1 ^ 2 + w2 ^ 2 + w3 ^ 2 + w4 ^
2) "Thrust in the positive direction of the local  Z axis";
  SI.Torque Tau[3, 1] = A * transpose(SqOmega) "Vector of
torques driven on the quadcopter";
  SI.Torque TauL[4, 1] = DC * transpose(SqOmega)
"Aerodynamic load";
  //
  //Input assignation
  SI.Voltage Vol[4, 1] = [V1.p.v * 0.111; V2.p.v * 0.111;
V3.p.v * 0.111; V4.p.v * 0.111];
initial equation
  x = [x0; y0; z0];
  OriVec = [phi0; theta0; psi0];
  v = [0; 0; 0];
  AngVel = [0; 0; 0];
  w1 = w10;
  w2 = w20;
  w3 = w30;
  w4 = w40;
equation
  connect(InV4, V4.v) annotation(
    Line(points = {{-120, -66}, {-78, -66}, {-78, -66}, {-
76, -66}}, color = {0, 0, 127}));
  connect(InV2, V2.v) annotation(
    Line(points = {{-120, 24}, {-78, 24}, {-78, 24}, {-76,
24}}, color = {0, 0, 127}));
  connect(InV3, V3.v) annotation(
    Line(points = {{-118, -20}, {-76, -20}}, color = {0, 0,
127}));
  connect(InV1, V1.v) annotation(
    Line(points = {{-118, 74}, {-76, 74}}, color = {0, 0,
127}));
//Connections
  connect(V4.n, G4.p) annotation(
```

```
    Line(points = {{-64, -48}, {-22, -48}, {-22, -52}, {-
20, -52}}, color = {0, 0, 255}));
  connect(V3.n, G3.p) annotation(
    Line(points = {{-64, -2}, {-20, -2}, {-20, -6}, {-20, -
6}, {-20, -6}}, color = {0, 0, 255}));
  connect(V2.n, G2.p) annotation(
    Line(points = {{-64, 42}, {-43, 42}, {-43, 40}, {-22,
40}}, color = {0, 0, 255}));
  connect(V1.n, G1.p) annotation(
    Line(points = {{-64, 92}, {-22, 92}, {-22, 88}}, color
= {0, 0, 255}));
//Equations
  derX = der(x);
  der(x) = Rot * v;
  der(OriVec) = RotA * AngVel;
  der(v) = [0; 0; Thrust / Mt] + g * [sin(theta); -sin(phi)
* cos(theta); -cos(phi) * cos(theta)] - S * v;
  der(AngVel) = invJ * (Tau - S * J * AngVel - S * [0; 0;
Ir * Gyro]);
  Ir * der(w1) + Km * Ke / Res * w1 = Km / Res * Vol[1, 1]
- TauL[1, 1];
  Ir * der(w2) + Km * Ke / Res * w2 = Km / Res * Vol[2, 1]
- TauL[2, 1];
  Ir * der(w3) + Km * Ke / Res * w3 = Km / Res * Vol[3, 1]
- TauL[3, 1];
  Ir * der(w4) + Km * Ke / Res * w4 = Km / Res * Vol[4, 1]
- TauL[4, 1];
//Output assignation
  OutPos = {x[1,1], x[2,1], x[3,1]};
  OutVel = {der(x[1, 1]), der(x[2, 1]), der(x[3, 1])};
  OutAcc = {der(derX[1, 1]), der(derX[2, 1]), der(derX[3,
1])};
  OutOri = {phi, theta, psi};
  OutAngVel = {der(phi), der(theta), der(psi)};
  OutW = {w1, w2, w3, w4};
//
  annotation(
    uses(Modelica(version = "3.2.2")));
end Quadcopter;
```

Control de velocidad de los rotores por emulación

```
model MotorControl
  Modelica.Blocks.Interfaces.RealOutput u annotation(
    Placement(visible = true, transformation(origin = {116,
1.77636e-15}, extent = {{-16, -16}, {16, 16}}, rotation =
0), iconTransformation(origin = {116, 1.77636e-15}, extent
= {{-16, -16}, {16, 16}}, rotation = 0)));
```

```
  Modelica.Blocks.Interfaces.RealInput e annotation(
    Placement(visible = true, transformation(origin = {-
120, 0}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {-120, 0}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  parameter Real T = 0.003;
  Real u_k1(start = 56.4567), u_k2(start = 56.4567),
e_k(start = 0), e_k1(start = 0), e_k2(start = 0);
algorithm
  when sample(0, T) then
    e_k := e;
    u := 1.7408 * u_k1 - 0.7408 * u_k2 + 0.02936 * e_k1 -
0.02724 * e_k2;
    if u > 100 then
      u := 100;
    end if;
    if u < 0 then
      u := 0;
    end if;
    e_k2 := e_k1;
    e_k1 := e_k;
    u_k2 := u_k1;
    u_k1 := u;
  end when;
end MotorControl;
```

Control de velocidad de los rotores por deadbeat

```
model MotorDeadbeat
  Modelica.Blocks.Interfaces.RealOutput u annotation(
    Placement(visible = true, transformation(origin = {116,
1.77636e-15}, extent = {{-16, -16}, {16, 16}}, rotation =
0), iconTransformation(origin = {116, 1.77636e-15}, extent
= {{-16, -16}, {16, 16}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput e annotation(
    Placement(visible = true, transformation(origin = {-
120, 0}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {-120, 0}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  parameter Real T = 0.003;
  Real u_k1(start = 56.4567), e_k(start = 0), e_k1(start =
0);
algorithm
  when sample(0, T) then
    e_k := e;
    u := u_k1 + 1.36 * e_k - 1.25596 * e_k1;
    if u > 100 then
      u := 100;
```

```
    end if;
    if u < 0 then
      u := 0;
    end if;
    e_k1 := e_k;
    u_k1 := u;
  end when;
end MotorDeadbeat;
```

Módulo de conversión de T y pares a velocidades angulares

```
model TtoW
  Modelica.Blocks.Interfaces.RealInput T annotation(
    Placement(visible = true, transformation(origin = {-
120, 80}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {-120, 80}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput TauPhi annotation(
    Placement(visible = true, transformation(origin = {-
120, 0}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {-120, 0}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput TauTheta annotation(
    Placement(visible = true, transformation(origin = {-
120, -40}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {-120, -40}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput TauPsi annotation(
    Placement(visible = true, transformation(origin = {-
120, -80}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {-120, -80}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput W1 annotation(
    Placement(visible = true, transformation(origin = {120,
80}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {120, 80}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput W2 annotation(
    Placement(visible = true, transformation(origin = {120,
26}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {120, 26}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput W3 annotation(
    Placement(visible = true, transformation(origin = {120,
-26}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {120, -26}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput W4 annotation(
```

```
    Placement(visible = true, transformation(origin = {120,
-80}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {120, -80}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  import Math = Modelica.Math;
  parameter Real La = 0.211 "Arm length";
  parameter Real K = 100135.1825 "Inverse of the Thrust
constant (9.9865 * 10 ^ (-6))";
  parameter Real B = 6258605.583 "Inverse of the Drag
constant (1.5978 * 10 ^ (-7))";
  Real R1, R2, R3, R4, Flag(start = 0);
equation
  R1 = T * K / 4 + TauPhi * K / (2 * La) - TauPsi * B / 4;
  R2 = T * K / 4 - TauTheta * K / (2 * La) + TauPsi * B /
4;
  R3 = T * K / 4 - TauPhi * K / (2 * La) - TauPsi * B / 4;
  R4 = T * K / 4 + TauTheta * K / (2 * La) + TauPsi * B /
4;
  W1 = if R1 < 0 then 0 else sqrt(R1);
  W2 = if R2 < 0 then 0 else sqrt(R2);
  W3 = if R3 < 0 then 0 else sqrt(R3);
  W4 = if R4 < 0 then 0 else sqrt(R4);
  Flag = if R1 < 0 or R2 < 0 or R3 < 0 or R4 < 0 then 1
else 0;
annotation(
    uses(Modelica(version = "3.2.2")));
end TtoW;
```

Control de orientación con realimentación taquimétrica

```
model OriControlP_D
  Modelica.Blocks.Interfaces.RealInput Phid annotation(
    Placement(visible = true, transformation(origin = {-
120, 60}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {-120, 60}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput Thetad annotation(
    Placement(visible = true, transformation(origin = {-
120, 0}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {-120, 0}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput Psid annotation(
    Placement(visible = true, transformation(origin = {-
120, -60}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {-120, -60}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput Phi annotation(
```

```
    Placement(visible = true, transformation(origin = {-80,
-120}, extent = {{-20, -20}, {20, 20}}, rotation = 90),
iconTransformation(origin = {-80, -120}, extent = {{-20, -
20}, {20, 20}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput Theta annotation(
    Placement(visible = true, transformation(origin = {-50,
-120}, extent = {{-20, -20}, {20, 20}}, rotation = 90),
iconTransformation(origin = {-50, -120}, extent = {{-20, -
20}, {20, 20}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput Psi annotation(
    Placement(visible = true, transformation(origin = {-20,
-120}, extent = {{-20, -20}, {20, 20}}, rotation = 90),
iconTransformation(origin = {-20, -120}, extent = {{-20, -
20}, {20, 20}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput derPhi annotation(
    Placement(visible = true, transformation(origin = {20,
-120}, extent = {{-20, -20}, {20, 20}}, rotation = 90),
iconTransformation(origin = {20, -120}, extent = {{-20, -
20}, {20, 20}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput derTheta annotation(
    Placement(visible = true, transformation(origin = {50,
-120}, extent = {{-20, -20}, {20, 20}}, rotation = 90),
iconTransformation(origin = {50, -120}, extent = {{-20, -
20}, {20, 20}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput derPsi annotation(
    Placement(visible = true, transformation(origin = {80,
-120}, extent = {{-20, -20}, {20, 20}}, rotation = 90),
iconTransformation(origin = {80, -120}, extent = {{-20, -
20}, {20, 20}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealOutput TauPhi annotation(
    Placement(visible = true, transformation(origin = {120,
40}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {120, 40}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput TauTheta
annotation(
    Placement(visible = true, transformation(origin = {120,
3.55271e-15}, extent = {{-20, -20}, {20, 20}}, rotation =
0), iconTransformation(origin = {120, 3.55271e-15}, extent
= {{-20, -20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput TauPsi annotation(
    Placement(visible = true, transformation(origin = {120,
-40}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {120, -40}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
//REALIMENTACIÓN TAQUIMÉTRICA
  parameter Real T = 0.05, Kp = 0.2888, Kd = 0.1216, Kpz =
0.5392, Kdz = 0.227;
algorithm
  when sample(0, T) then
```

```
    TauPhi := Kp*(Phid - Phi) - Kd*derPhi;
    TauTheta := Kp*(Thetad - Theta) - Kd*derTheta;
    TauPsi := Kpz*(Psid - Psi) - Kdz*derPsi;
  end when;
```

Control de orientación con deadbeat

```
model OriControlDeadBeat
  Modelica.Blocks.Interfaces.RealInput Phid annotation(
    Placement(visible = true, transformation(origin = {-
120, 60}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {-120, 60}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput Thetad annotation(
    Placement(visible = true, transformation(origin = {-
120, 0}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {-120, 0}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput Psid annotation(
    Placement(visible = true, transformation(origin = {-
120, -60}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {-120, -60}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput Phi annotation(
    Placement(visible = true, transformation(origin = {-80,
-120}, extent = {{-20, -20}, {20, 20}}, rotation = 90),
iconTransformation(origin = {-80, -120}, extent = {{-20, -
20}, {20, 20}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput Theta annotation(
    Placement(visible = true, transformation(origin = {-50,
-120}, extent = {{-20, -20}, {20, 20}}, rotation = 90),
iconTransformation(origin = {-50, -120}, extent = {{-20, -
20}, {20, 20}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput Psi annotation(
    Placement(visible = true, transformation(origin = {-20,
-120}, extent = {{-20, -20}, {20, 20}}, rotation = 90),
iconTransformation(origin = {-20, -120}, extent = {{-20, -
20}, {20, 20}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput derPhi annotation(
    Placement(visible = true, transformation(origin = {20,
-120}, extent = {{-20, -20}, {20, 20}}, rotation = 90),
iconTransformation(origin = {20, -120}, extent = {{-20, -
20}, {20, 20}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput derTheta annotation(
    Placement(visible = true, transformation(origin = {50,
-120}, extent = {{-20, -20}, {20, 20}}, rotation = 90),
```

```
iconTransformation(origin = {50, -120}, extent = {{-20, -
20}, {20, 20}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput derPsi annotation(
    Placement(visible = true, transformation(origin = {80,
-120}, extent = {{-20, -20}, {20, 20}}, rotation = 90),
iconTransformation(origin = {80, -120}, extent = {{-20, -
20}, {20, 20}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealOutput TauPhi annotation(
    Placement(visible = true, transformation(origin = {120,
40}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {120, 40}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput TauTheta
annotation(
    Placement(visible = true, transformation(origin = {120,
3.55271e-15}, extent = {{-20, -20}, {20, 20}}, rotation =
0), iconTransformation(origin = {120, 3.55271e-15}, extent
= {{-20, -20}, {20, 20}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput TauPsi annotation(
    Placement(visible = true, transformation(origin = {120,
-40}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {120, -40}, extent = {{-20, -
20}, {20, 20}}, rotation = 0)));
  //DEADBEAT CONTROL
  parameter Real T = 0.05;
  Real ux, u_k1x(start = 0), e_kx(start = 0), e_k1x(start =
0);
  Real uy, u_k1y(start = 0), e_ky(start = 0), e_k1y(start =
0);
  Real uz, u_k1z(start = 0), e_kz(start = 0), e_k1z(start =
0);
algorithm
  when sample(0, T) then
  //X
    e_kx := Phid - Phi;
    ux := -u_k1x + 10.24003 * (e_kx - e_k1x);
    e_k1x := e_kx;
    u_k1x := ux;
    TauPhi := ux;
  //Y
    e_ky := Thetad - Theta;
    uy := -u_k1y + 10.24003 * (e_ky - e_k1y);
    e_k1y := e_ky;
    u_k1y := uy;
    TauTheta := uy;
  //Z
    e_kz := Psid - Psi;
    uz := -u_k1z + 19.12046 * (e_kz - e_k1z);
    e_k1z := e_kz;
    u_k1z := uz;
```

```
    TauPsi := uz;
  end when;
```

Control de posición

```
model PosControl
  Modelica.Blocks.Interfaces.RealInput xd annotation(
    Placement(visible = true, transformation(origin = {-
113, 93}, extent = {{-13, -13}, {13, 13}}, rotation = 0),
iconTransformation(origin = {-110, 90}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput yd annotation(
    Placement(visible = true, transformation(origin = {-
113, 33}, extent = {{-13, -13}, {13, 13}}, rotation = 0),
iconTransformation(origin = {-110, 30}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput zd annotation(
    Placement(visible = true, transformation(origin = {-
113, -27}, extent = {{-13, -13}, {13, 13}}, rotation = 0),
iconTransformation(origin = {-110, -30}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput derXd annotation(
    Placement(visible = true, transformation(origin = {-
113, 73}, extent = {{-13, -13}, {13, 13}}, rotation = 0),
iconTransformation(origin = {-110, 70}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput derYd annotation(
    Placement(visible = true, transformation(origin = {-
113, 13}, extent = {{-13, -13}, {13, 13}}, rotation = 0),
iconTransformation(origin = {-110, 10}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput derZd annotation(
    Placement(visible = true, transformation(origin = {-
113, -47}, extent = {{-13, -13}, {13, 13}}, rotation = 0),
iconTransformation(origin = {-110, -50}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput derderXd annotation(
    Placement(visible = true, transformation(origin = {-
113, 53}, extent = {{-13, -13}, {13, 13}}, rotation = 0),
iconTransformation(origin = {-110, 50}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput derderYd annotation(
    Placement(visible = true, transformation(origin = {-
113, -7}, extent = {{-13, -13}, {13, 13}}, rotation = 0),
iconTransformation(origin = {-110, -10}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput derderZd annotation(
```

```
    Placement(visible = true, transformation(origin = {-
113, -67}, extent = {{-13, -13}, {13, 13}}, rotation = 0),
iconTransformation(origin = {-110, -70}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput Psid annotation(
    Placement(visible = true, transformation(origin = {-
113, -87}, extent = {{-13, -13}, {13, 13}}, rotation = 0),
iconTransformation(origin = {-110, -90}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput x annotation(
    Placement(visible = true, transformation(origin = {-93,
-113}, extent = {{-13, -13}, {13, 13}}, rotation = 90),
iconTransformation(origin = {-90, -110}, extent = {{-10, -
10}, {10, 10}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput y annotation(
    Placement(visible = true, transformation(origin = {-73,
-113}, extent = {{-13, -13}, {13, 13}}, rotation = 90),
iconTransformation(origin = {-70, -110}, extent = {{-10, -
10}, {10, 10}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput z annotation(
    Placement(visible = true, transformation(origin = {-53,
-113}, extent = {{-13, -13}, {13, 13}}, rotation = 90),
iconTransformation(origin = {-50, -110}, extent = {{-10, -
10}, {10, 10}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput derZ annotation(
    Placement(visible = true, transformation(origin = {7, -
113}, extent = {{-13, -13}, {13, 13}}, rotation = 90),
iconTransformation(origin = {10, -110}, extent = {{-10, -
10}, {10, 10}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput derderX annotation(
    Placement(visible = true, transformation(origin = {27,
-113}, extent = {{-13, -13}, {13, 13}}, rotation = 90),
iconTransformation(origin = {30, -110}, extent = {{-10, -
10}, {10, 10}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput derderY annotation(
    Placement(visible = true, transformation(origin = {47,
-113}, extent = {{-13, -13}, {13, 13}}, rotation = 90),
iconTransformation(origin = {50, -110}, extent = {{-10, -
10}, {10, 10}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput derderZ annotation(
    Placement(visible = true, transformation(origin = {67,
-113}, extent = {{-13, -13}, {13, 13}}, rotation = 90),
iconTransformation(origin = {70, -110}, extent = {{-10, -
10}, {10, 10}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput Psi annotation(
    Placement(visible = true, transformation(origin = {87,
-113}, extent = {{-13, -13}, {13, 13}}, rotation = 90),
iconTransformation(origin = {90, -110}, extent = {{-10, -
10}, {10, 10}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput derX annotation(
```

```
    Placement(visible = true, transformation(origin = {-33,
-113}, extent = {{-13, -13}, {13, 13}}, rotation = 90),
iconTransformation(origin = {-30, -110}, extent = {{-10, -
10}, {10, 10}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealInput derY annotation(
    Placement(visible = true, transformation(origin = {-13,
-113}, extent = {{-13, -13}, {13, 13}}, rotation = 90),
iconTransformation(origin = {-10, -110}, extent = {{-10, -
10}, {10, 10}}, rotation = 90)));
  Modelica.Blocks.Interfaces.RealOutput Phi annotation(
    Placement(visible = true, transformation(origin = {120,
-20}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {115, -25}, extent = {{-15, -
15}, {15, 15}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput Theta annotation(
    Placement(visible = true, transformation(origin = {120,
-50}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {116, -56}, extent = {{-16, -
16}, {16, 16}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput Thrust annotation(
    Placement(visible = true, transformation(origin = {120,
60}, extent = {{-20, -20}, {20, 20}}, rotation = 0),
iconTransformation(origin = {115, 55}, extent = {{-15, -
15}, {15, 15}}, rotation = 0)));
  import Math = Modelica.Math;
  parameter Real g = 9.81, m = 1.3199, T = 0.05;
  parameter Real Kpx = 1, Kdx = 12, Kddx = 2;
  parameter Real Kpy = 40, Kdy = 40, Kddy = 10;
  parameter Real Kpz = 20, Kdz = 10, Kddz = 2;

  Real dx, dy, dz;
algorithm
  when sample(0,T) then
    dx := Kpx * (xd - x) + Kdx * (derXd - derX) + Kddx *
(derderXd - derderX);
    dy := Kpy * (yd - y) + Kdy * (derYd - derY) + Kddy *
(derderYd - derderY);
    dz := Kpz * (zd - z) + Kdz * (derZd - derZ) + Kddz *
(derderZd - derderZ);
    Phi := asin((dx*sin(Psid) - dy*cos(Psid))/(dx^2 + dy^2
+ (dz + g)^2));
    Theta := atan((dx*cos(Psid) + dy*sin(Psid))/(dz + g));
    Thrust := m*(dx*(sin(Theta)*cos(Psid)*cos(Phi) +
sin(Psid)*sin(Phi)) +
                 dy*(sin(Theta)*sin(Psid)*cos(Phi) -
cos(Psid)*sin(Phi)) +
                 (dz + g)*cos(Theta)*cos(Phi));
  end when;
  annotation(
    uses(Modelica(version = "3.2.2")));
```

```
end PosControl;
```

Generador de trayectorias

```
model TrajGen3
  Modelica.Blocks.Interfaces.RealInput xd annotation(
    Placement(visible = true, transformation(origin = {-
124, 50}, extent = {{-24, -24}, {24, 24}}, rotation = 0),
iconTransformation(origin = {-124, 50}, extent = {{-24, -
24}, {24, 24}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealInput yd annotation(
    Placement(visible = true, transformation(origin = {-
124, -3.55271e-15}, extent = {{-24, -24}, {24, 24}},
rotation = 0), iconTransformation(origin = {-124, -
3.55271e-15}, extent = {{-24, -24}, {24, 24}}, rotation =
0)));
  Modelica.Blocks.Interfaces.RealInput zd annotation(
    Placement(visible = true, transformation(origin = {-
124, -50}, extent = {{-24, -24}, {24, 24}}, rotation = 0),
iconTransformation(origin = {-124, -50}, extent = {{-24, -
24}, {24, 24}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput y annotation(
    Placement(visible = true, transformation(origin = {110,
14}, extent = {{-10, -10}, {10, 10}}, rotation = 0),
iconTransformation(origin = {110, 14}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput vy annotation(
    Placement(visible = true, transformation(origin = {110,
0}, extent = {{-10, -10}, {10, 10}}, rotation = 0),
iconTransformation(origin = {110, 0}, extent = {{-10, -10},
{10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput accy annotation(
    Placement(visible = true, transformation(origin = {110,
-14}, extent = {{-10, -10}, {10, 10}}, rotation = 0),
iconTransformation(origin = {110, -14}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput z annotation(
    Placement(visible = true, transformation(origin = {110,
-50}, extent = {{-10, -10}, {10, 10}}, rotation = 0),
iconTransformation(origin = {110, -50}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput x annotation(
    Placement(visible = true, transformation(origin = {110,
84}, extent = {{-10, -10}, {10, 10}}, rotation = 0),
iconTransformation(origin = {110, 84}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput vx annotation(
```

```
    Placement(visible = true, transformation(origin = {110,
70}, extent = {{-10, -10}, {10, 10}}, rotation = 0),
iconTransformation(origin = {110, 70}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput accx annotation(
    Placement(visible = true, transformation(origin = {110,
56}, extent = {{-10, -10}, {10, 10}}, rotation = 0),
iconTransformation(origin = {110, 56}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput vz annotation(
    Placement(visible = true, transformation(origin = {110,
-64}, extent = {{-10, -10}, {10, 10}}, rotation = 0),
iconTransformation(origin = {110, -64}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Interfaces.RealOutput accz annotation(
    Placement(visible = true, transformation(origin = {110,
-78}, extent = {{-10, -10}, {10, 10}}, rotation = 0),
iconTransformation(origin = {110, -78}, extent = {{-10, -
10}, {10, 10}}, rotation = 0)));

  import Math = Modelica.Math;
  parameter Real PI = 3.1415926535;
  parameter Real b = 0.25, Vmax = 0.3;

  Real ax, ay, az, c, VmaxX, VmaxY, VmaxZ;
  Real Jouncex, Jerkx, Accx, Velx, Posx;
  Real Jouncey, Jerky, Accy, Vely, Posy;
  Real Jouncez, Jerkz, Accz, Velz, Posz;

equation
  if (abs(xd) > abs(yd)) and (abs(xd) > abs(zd)) then
    c = (abs(xd) - Vmax) / Vmax;
    VmaxX = xd/(c+1);
    VmaxY = yd/(c+1);
    VmaxZ = zd/(c+1);
  elseif abs(yd) > abs(zd) then
    c = (abs(yd) - Vmax) / Vmax;
    VmaxX = xd/(c+1);
    VmaxY = yd/(c+1);
    VmaxZ = zd/(c+1);
  else
    c = (abs(zd) - Vmax) / Vmax;
    VmaxX = xd/(c+1);
    VmaxY = yd/(c+1);
    VmaxZ = zd/(c+1);
  end if;

  ax = VmaxX/0.02097;
  ay = VmaxY/0.02097;
  az = VmaxZ/0.02097;
```

```
  if time <= b then
    Jouncex = ax * sin(1 / b * PI * time);
    Jouncey = ay * sin(1 / b * PI * time);
    Jouncez = az * sin(1 / b * PI * time);
  elseif time > b and time <= 3 * b then
    Jouncex = -ax * sin(1 / (2 * b) * PI * time - PI / 2);
    Jouncey = -ay * sin(1 / (2 * b) * PI * time - PI / 2);
    Jouncez = -az * sin(1 / (2 * b) * PI * time - PI / 2);
  elseif time > 3 * b and time < 4 * b then
    Jouncex = ax * sin(1 / b * PI * time - 3 * PI);
    Jouncey = ay * sin(1 / b * PI * time - 3 * PI);
    Jouncez = az * sin(1 / b * PI * time - 3 * PI);
  elseif time >= 4 * b and time <= 4 * b + c then
    Jouncex = 0;
    Jouncey = 0;
    Jouncez = 0;
  elseif time > 4 * b + c and time <= 4 * b + c + b then
    Jouncex = -ax * sin(1 / b * PI * (time - (4 * b + c)));
    Jouncey = -ay * sin(1 / b * PI * (time - (4 * b + c)));
    Jouncez = -az * sin(1 / b * PI * (time - (4 * b + c)));
  elseif time > 4 * b + c + b and time <= 4 * b + c + 3 * b
then
    Jouncex = ax * sin(1 / (2 * b) * PI * (time - (4 * b +
c)) - PI / 2);
    Jouncey = ay * sin(1 / (2 * b) * PI * (time - (4 * b +
c)) - PI / 2);
    Jouncez = az * sin(1 / (2 * b) * PI * (time - (4 * b +
c)) - PI / 2);
  elseif time > 4 * b + c + 3 * b and time <= 4 * b + c + 4
* b then
    Jouncex = -ax * sin(1 / b * PI * (time - (4 * b + c)) -
3 * PI);
    Jouncey = -ay * sin(1 / b * PI * (time - (4 * b + c)) -
3 * PI);
    Jouncez = -az * sin(1 / b * PI * (time - (4 * b + c)) -
3 * PI);
  else
    Jouncex = 0;
    Jouncey = 0;
    Jouncez = 0;
  end if;

  der(Jerkx) = if (time > 4*b) and (time < 4*b + c) then 0
else Jouncex;
  der(Accx) = if (time > 4*b) and (time < 4*b + c) then 0
else Jerkx;
  der(Velx) = if (time > 4*b) and (time < 4*b + c) then 0
else Accx;
  der(Posx) = Velx;
```

```
  der(Jerky) = if (time > 4*b) and (time < 4*b + c) then 0
else Jouncey;
  der(Accy) = if (time > 4*b) and (time < 4*b + c) then 0
else Jerky;
  der(Vely) = if (time > 4*b) and (time < 4*b + c) then 0
else Accy;
  der(Posy) = Vely;

  der(Jerkz) = if (time > 4*b) and (time < 4*b + c) then 0
else Jouncez;
  der(Accz) = if (time > 4*b) and (time < 4*b + c) then 0
else Jerkz;
  der(Velz) = if (time > 4*b) and (time < 4*b + c) then 0
else Accz;
  der(Posz) = Velz;

  x = if time > 8 * b + c then pre(x) else Posx;
  vx = if time > 8 * b + c then 0 else Velx;
  accx = if time > 8 * b + c then 0 else Accx;

  y = if time > 8 * b + c then pre(y) else Posy;
  vy = if time > 8 * b + c then 0 else Vely;
  accy = if time > 8 * b + c then 0 else Accy;

  z = if time > 8 * b + c then pre(z) else Posz;
  vz = if time > 8 * b + c then 0 else Velz;
  accz = if time > 8 * b + c then 0 else Accz;
  annotation(
    uses(Modelica(version = "3.2.2")));
end TrajGen3;
```

Esquema de control global

```
model Control
  Quadcopter Q1 annotation(
    Placement(visible = true, transformation(origin = {267,
35}, extent = {{-33, -33}, {33, 33}}, rotation = 0)));
  Modelica.Blocks.Math.Feedback FeedW1 annotation(
    Placement(visible = true, transformation(origin = {110,
60}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Math.Feedback FeedW2 annotation(
    Placement(visible = true, transformation(origin = {126,
42}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Math.Feedback FeedW3 annotation(
    Placement(visible = true, transformation(origin = {138,
28}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Math.Feedback FeedW4 annotation(
    Placement(visible = true, transformation(origin = {152,
12}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
```

```
  PosControl PosC annotation(
    Placement(visible = true, transformation(origin = {-
155, 35}, extent = {{-43, -43}, {43, 43}}, rotation = 0)));
  Modelica.Blocks.Sources.Constant X(k = 3) annotation(
    Placement(visible = true, transformation(origin = {-
366, 68}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Sources.Constant Y(k = 2)  annotation(
    Placement(visible = true, transformation(origin = {-
366, 38}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Sources.Constant Z(k = 1) annotation(
    Placement(visible = true, transformation(origin = {-
366, 10}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Sources.Constant Psi(k = 0)  annotation(
    Placement(visible = true, transformation(origin = {-
252, -30}, extent = {{-10, -10}, {10, 10}}, rotation =
0)));
  TrajGen3 TGen annotation(
    Placement(visible = true, transformation(origin = {-
283, 39}, extent = {{-41, -41}, {41, 41}}, rotation = 0)));
  // 0.003 "Sample period for motors control";
  // 0.05 "Sample period for orientation and position
control";
  OriControlP_D OriC annotation(
    Placement(visible = true, transformation(origin = {-43,
23}, extent = {{-21, -21}, {21, 21}}, rotation = 0)));
  TtoW ttoW1 annotation(
    Placement(visible = true, transformation(origin = {51,
35}, extent = {{-31, -31}, {31, 31}}, rotation = 0)));
  MotorControl R1 annotation(
    Placement(visible = true, transformation(origin = {150,
60}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  MotorControl R2 annotation(
    Placement(visible = true, transformation(origin = {198,
42}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  MotorControl R3 annotation(
    Placement(visible = true, transformation(origin = {170,
28}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  MotorControl R4 annotation(
    Placement(visible = true, transformation(origin = {204,
12}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
equation
  connect(R4.u, Q1.InV4) annotation(
    Line(points = {{216, 12}, {222, 12}, {222, 14}, {228,
14}}, color = {0, 0, 127}));
  connect(FeedW4.y, R4.e) annotation(
    Line(points = {{162, 12}, {192, 12}, {192, 12}, {192,
12}}, color = {0, 0, 127}));
  connect(R3.u, Q1.InV3) annotation(
    Line(points = {{182, 28}, {222, 28}, {222, 28}, {228,
28}}, color = {0, 0, 127}));
```

```
  connect(FeedW3.y, R3.e) annotation(
    Line(points = {{148, 28}, {156, 28}, {156, 28}, {158,
28}}, color = {0, 0, 127}));
  connect(R2.u, Q1.InV2) annotation(
    Line(points = {{210, 42}, {224, 42}, {224, 42}, {228,
42}}, color = {0, 0, 127}));
  connect(FeedW2.y, R2.e) annotation(
    Line(points = {{136, 42}, {184, 42}, {184, 42}, {186,
42}}, color = {0, 0, 127}));
  connect(R1.u, Q1.InV1) annotation(
    Line(points = {{162, 60}, {220, 60}, {220, 60}, {228,
60}}, color = {0, 0, 127}));
  connect(FeedW1.y, R1.e) annotation(
    Line(points = {{120, 60}, {138, 60}, {138, 60}, {138,
60}}, color = {0, 0, 127}));
  connect(Psi.y, OriC.Psid) annotation(
    Line(points = {{-240, -30}, {-92, -30}, {-92, 10}, {-
68, 10}, {-68, 10}}, color = {0, 0, 127}));
  connect(Q1.OutOri[3], PosC.Psi) annotation(
    Line(points = {{304, 30}, {340, 30}, {340, -24}, {-116,
-24}, {-116, -12}, {-116, -12}}, color = {0, 0, 127},
thickness = 0.5));
  connect(Psi.y, PosC.Psid) annotation(
    Line(points = {{-241, -30}, {-229, -30}, {-229, -2}, {-
203, -2}}, color = {0, 0, 127}));
  connect(Q1.OutPos[3], PosC.z) annotation(
    Line(points = {{304.95, 58.45}, {364.95, 58.45},
{364.95, -43.55}, {-177.05, -43.55}, {-177.05, -9.54999},
{-175.05, -9.54999}}, color = {0, 0, 127}, thickness =
0.5));
  connect(Q1.OutPos[2], PosC.y) annotation(
    Line(points = {{304.95, 58.45}, {364.95, 58.45},
{364.95, -43.55}, {-185.05, -43.55}, {-185.05, -9.54999},
{-185.05, -9.54999}}, color = {0, 0, 127}, thickness =
0.5));
  connect(Q1.OutPos[1], PosC.x) annotation(
    Line(points = {{304.95, 58.45}, {364.95, 58.45},
{364.95, -43.55}, {-193.05, -43.55}, {-193.05, -9.54999},
{-193.05, -9.54999}}, color = {0, 0, 127}, thickness =
0.5));
  connect(Q1.OutVel[3], PosC.derZ) annotation(
    Line(points = {{304.62, 52.18}, {358.62, 52.18},
{358.62, -37.82}, {-149.38, -37.82}, {-149.38, -9.82}, {-
149.38, -9.82}}, color = {0, 0, 127}, thickness = 0.5));
  connect(Q1.OutVel[2], PosC.derY) annotation(
    Line(points = {{304.62, 52.18}, {358.62, 52.18},
{358.62, -37.82}, {-159.38, -37.82}, {-159.38, -9.82}, {-
159.38, -9.82}}, color = {0, 0, 127}, thickness = 0.5));
  connect(Q1.OutVel[1], PosC.derX) annotation(
```

```
    Line(points = {{304.62, 52.18}, {358.62, 52.18},
{358.62, -37.82}, {-167.38, -37.82}, {-167.38, -9.82}, {-
167.38, -9.82}}, color = {0, 0, 127}, thickness = 0.5));
  connect(Q1.OutAcc[3], PosC.derderZ) annotation(
    Line(points = {{304.62, 46.24}, {350.62, 46.24},
{350.62, -29.76}, {-125.38, -29.76}, {-125.38, -9.76001},
{-123.38, -9.76001}}, color = {0, 0, 127}, thickness =
0.5));
  connect(Q1.OutAcc[2], PosC.derderY) annotation(
    Line(points = {{304.62, 46.24}, {350.62, 46.24},
{350.62, -29.76}, {-133.38, -29.76}, {-133.38, -9.76001},
{-133.38, -9.76001}}, color = {0, 0, 127}, thickness =
0.5));
  connect(Q1.OutAcc[1], PosC.derderX) annotation(
    Line(points = {{304.62, 46.24}, {350.62, 46.24},
{350.62, -29.76}, {-141.38, -29.76}, {-141.38, -9.76001},
{-141.38, -9.76001}}, color = {0, 0, 127}, thickness =
0.5));
  connect(Q1.OutOri[3], OriC.Psi) annotation(
    Line(points = {{304.95, 32.71}, {340.95, 32.71},
{340.95, -19.29}, {-47.05, -19.29}, {-47.05, 0.710002}, {-
47.05, 0.710002}}, color = {0, 0, 127}, thickness = 0.5));
  connect(Q1.OutOri[2], OriC.Theta) annotation(
    Line(points = {{304.95, 32.71}, {340.95, 32.71},
{340.95, -19.29}, {-53.05, -19.29}, {-53.05, 0.710002}, {-
53.05, 0.710002}}, color = {0, 0, 127}, thickness = 0.5));
  connect(Q1.OutOri[1], OriC.Phi) annotation(
    Line(points = {{304.95, 32.71}, {340.95, 32.71},
{340.95, -19.29}, {-59.05, -19.29}, {-59.05, 0.710002}, {-
59.05, 0.710002}}, color = {0, 0, 127}, thickness = 0.5));
  connect(Q1.OutAngVel[3], OriC.derPsi) annotation(
    Line(points = {{304.95, 26.11}, {332.95, 26.11},
{332.95, -11.89}, {-25.05, -11.89}, {-25.05, 0.109998}, {-
25.05, 0.109998}}, color = {0, 0, 127}, thickness = 0.5));
  connect(Q1.OutAngVel[2], OriC.derTheta) annotation(
    Line(points = {{304.95, 26.11}, {332.95, 26.11},
{332.95, -11.89}, {-31.05, -11.89}, {-31.05, 0.109998}, {-
31.05, 0.109998}}, color = {0, 0, 127}, thickness = 0.5));
  connect(Q1.OutAngVel[1], OriC.derPhi) annotation(
    Line(points = {{304.95, 26.11}, {332.95, 26.11},
{332.95, -11.89}, {-37.05, -11.89}, {-37.05, 0.109998}, {-
37.05, 0.109998}}, color = {0, 0, 127}, thickness = 0.5));
  connect(Q1.OutW[4], FeedW4.u2) annotation(
    Line(points = {{304.62, 10.6}, {324.62, 10.6}, {324.62,
-3.4}, {152.62, -3.4}, {152.62, 6.6}, {152.62, 6.6}}, color
= {0, 0, 127}, thickness = 0.5));
  connect(Q1.OutW[3], FeedW3.u2) annotation(
    Line(points = {{304.62, 10.6}, {324.62, 10.6}, {324.62,
-3.4}, {138.62, -3.4}, {138.62, 22.6}, {138.62, 22.6}},
color = {0, 0, 127}, thickness = 0.5));
```

```
  connect(Q1.OutW[2], FeedW2.u2) annotation(
    Line(points = {{304.62, 10.6}, {324.62, 10.6}, {324.62,
-3.4}, {126.62, -3.4}, {126.62, 36.6}, {126.62, 36.6}},
color = {0, 0, 127}, thickness = 0.5));
  connect(Q1.OutW[1], FeedW1.u2) annotation(
    Line(points = {{304.62, 10.6}, {324.62, 10.6}, {324.62,
-3.4}, {110.62, -3.4}, {110.62, 54.6}, {110.62, 54.6}},
color = {0, 0, 127}, thickness = 0.5));
  connect(ttoW1.W1, FeedW1.u1) annotation(
    Line(points = {{88.2, 59.8}, {102.2, 59.8}}, color =
{0, 0, 127}));
  connect(ttoW1.W2, FeedW2.u1) annotation(
    Line(points = {{88.2, 43.06}, {104.2, 43.06}, {104.2,
42.06}, {118.2, 42.06}}, color = {0, 0, 127}));
  connect(ttoW1.W3, FeedW3.u1) annotation(
    Line(points = {{88.2, 26.94}, {130.2, 26.94}, {130.2,
28.94}, {130.2, 28.94}}, color = {0, 0, 127}));
  connect(ttoW1.W4, FeedW4.u1) annotation(
    Line(points = {{88.2, 10.2}, {142.2, 10.2}, {142.2,
12.2}, {144.2, 12.2}}, color = {0, 0, 127}));
  connect(PosC.Thrust, ttoW1.T) annotation(
    Line(points = {{-105.55, 58.65}, {10.45, 58.65},
{10.45, 60.65}, {14.45, 60.65}}, color = {0, 0, 127}));
  connect(OriC.TauPsi, ttoW1.TauPsi) annotation(
    Line(points = {{-17.8, 14.6}, {8.2, 14.6}, {8.2, 9.6},
{14.2, 9.6}}, color = {0, 0, 127}));
  connect(OriC.TauTheta, ttoW1.TauTheta) annotation(
    Line(points = {{-17.8, 23}, {-2.8, 23}, {-2.8, 22},
{14.2, 22}}, color = {0, 0, 127}));
  connect(OriC.TauPhi, ttoW1.TauPhi) annotation(
    Line(points = {{-17.8, 31.4}, {10.2, 31.4}, {10.2,
36.4}, {14.2, 36.4}}, color = {0, 0, 127}));
  connect(PosC.Phi, OriC.Phid) annotation(
    Line(points = {{-105.55, 26.25}, {-89.55, 26.25}, {-
89.55, 38.25}, {-67.55, 38.25}, {-67.55, 38.25}}, color =
{0, 0, 127}));
  connect(PosC.Theta, OriC.Thetad) annotation(
    Line(points = {{-105.12, 12.92}, {-83.12, 12.92}, {-
83.12, 24.92}, {-67.12, 24.92}, {-67.12, 26.92}}, color =
{0, 0, 127}));
  connect(TGen.z, PosC.zd) annotation(
    Line(points = {{-237.9, 18.5}, {-221.9, 18.5}, {-221.9,
22.5}, {-201.9, 22.5}, {-201.9, 22.5}}, color = {0, 0,
127}));
  connect(TGen.accz, PosC.derderZd) annotation(
    Line(points = {{-237.9, 7.02}, {-205.9, 7.02}, {-205.9,
3.02}, {-201.9, 3.02}}, color = {0, 0, 127}));
  connect(TGen.vz, PosC.derZd) annotation(
    Line(points = {{-237.9, 12.76}, {-203.9, 12.76}, {-
203.9, 14.76}, {-201.9, 14.76}}, color = {0, 0, 127}));
```

```
  connect(TGen.accy, PosC.derderYd) annotation(
    Line(points = {{-237.9, 33.26}, {-205.9, 33.26}, {-
205.9, 29.26}, {-201.9, 29.26}}, color = {0, 0, 127}));
  connect(TGen.vy, PosC.derYd) annotation(
    Line(points = {{-237.9, 39}, {-205.9, 39}, {-205.9,
39}, {-201.9, 39}}, color = {0, 0, 127}));
  connect(TGen.y, PosC.yd) annotation(
    Line(points = {{-237.9, 44.74}, {-205.9, 44.74}, {-
205.9, 48.74}, {-201.9, 48.74}}, color = {0, 0, 127}));
  connect(TGen.accx, PosC.derderXd) annotation(
    Line(points = {{-237.9, 61.96}, {-221.9, 61.96}, {-
221.9, 55.96}, {-201.9, 55.96}, {-201.9, 55.96}}, color =
{0, 0, 127}));
  connect(TGen.vx, PosC.derXd) annotation(
    Line(points = {{-237.9, 67.7}, {-205.9, 67.7}, {-205.9,
65.7}, {-201.9, 65.7}}, color = {0, 0, 127}));
  connect(TGen.x, PosC.xd) annotation(
    Line(points = {{-237.9, 73.44}, {-203.9, 73.44}, {-
203.9, 73.44}, {-201.9, 73.44}}, color = {0, 0, 127}));
  connect(Z.y, TGen.zd) annotation(
    Line(points = {{-355, 12}, {-351, 12}, {-351, 20}, {-
335, 20}, {-335, 20}}, color = {0, 0, 127}));
  connect(X.y, TGen.xd) annotation(
    Line(points = {{-355, 70}, {-351, 70}, {-351, 60}, {-
335, 60}, {-335, 62}}, color = {0, 0, 127}));
  connect(Y.y, TGen.yd) annotation(
    Line(points = {{-355, 40}, {-337, 40}, {-337, 42}, {-
335, 42}}, color = {0, 0, 127}));
  annotation(
    uses(Modelica(version = "3.2.2")));
end Control;
```