UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Superior d'Enginyeries Industrial, Aeroespacial i Audiovisual de Terrassa

BACHELOR'S THESIS

HEAT AND MASS TRANSFER TECHNOLOGICAL CENTER

Disseny hidrodinàmic de hidrofoils per embarcacions esportives lleugeres a rem

ATTACHMENT 2: General Scheme for the joined model and its inner functions.

POLYTECHNIC UNIVERSITY OF CATALONIA

ESEIAAT

Author: Joan Altimira Marfà

Director: Asensio Oliva Llena

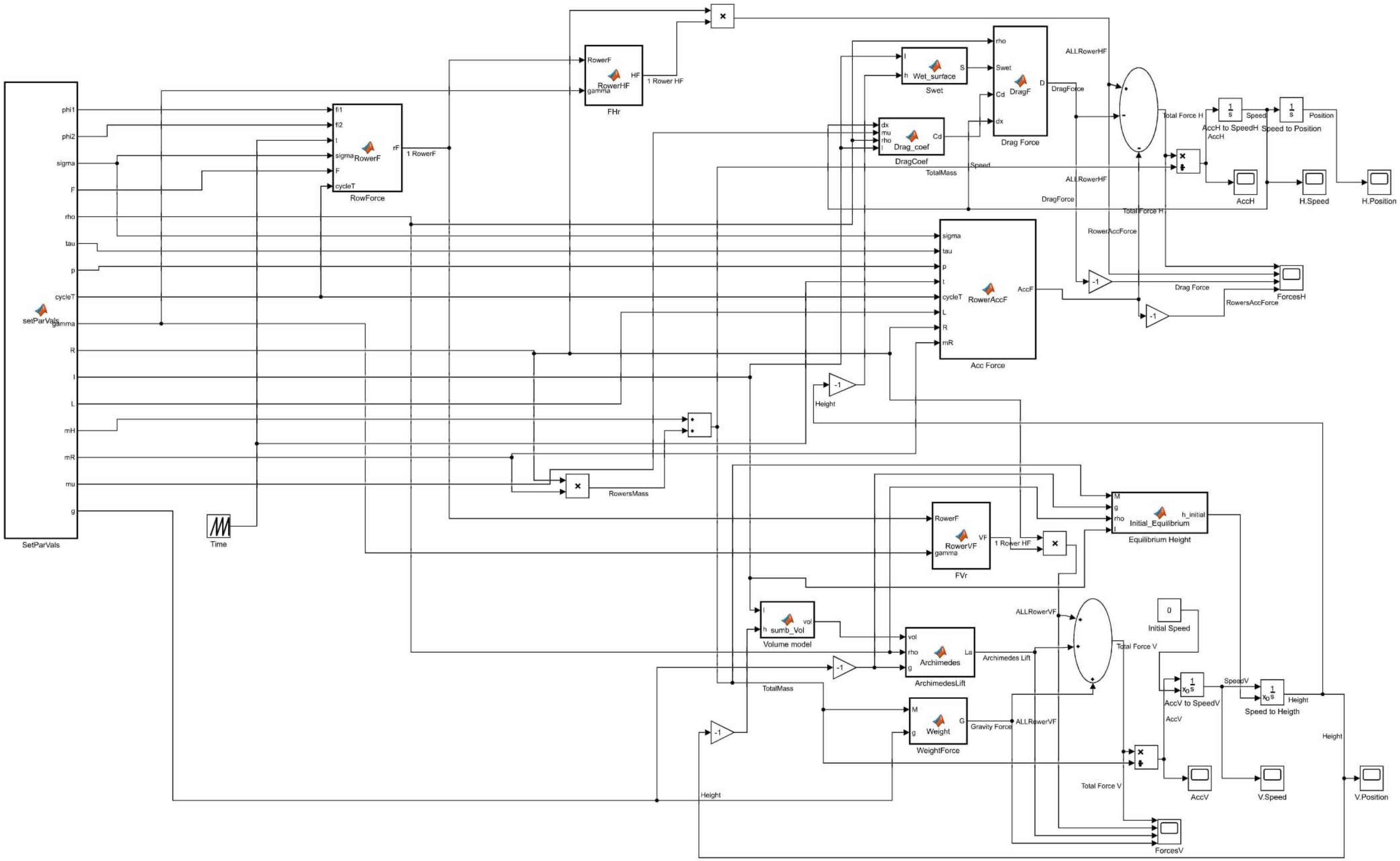
Co-Director: Aleix Baez Vidal

Delivery Date: 10th June, 2018

Studies: GrETA

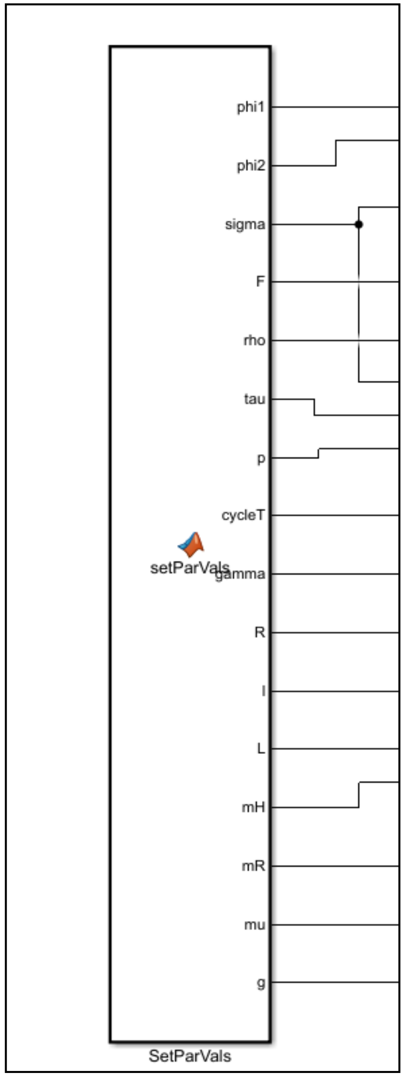
Contents

1. General sheme:

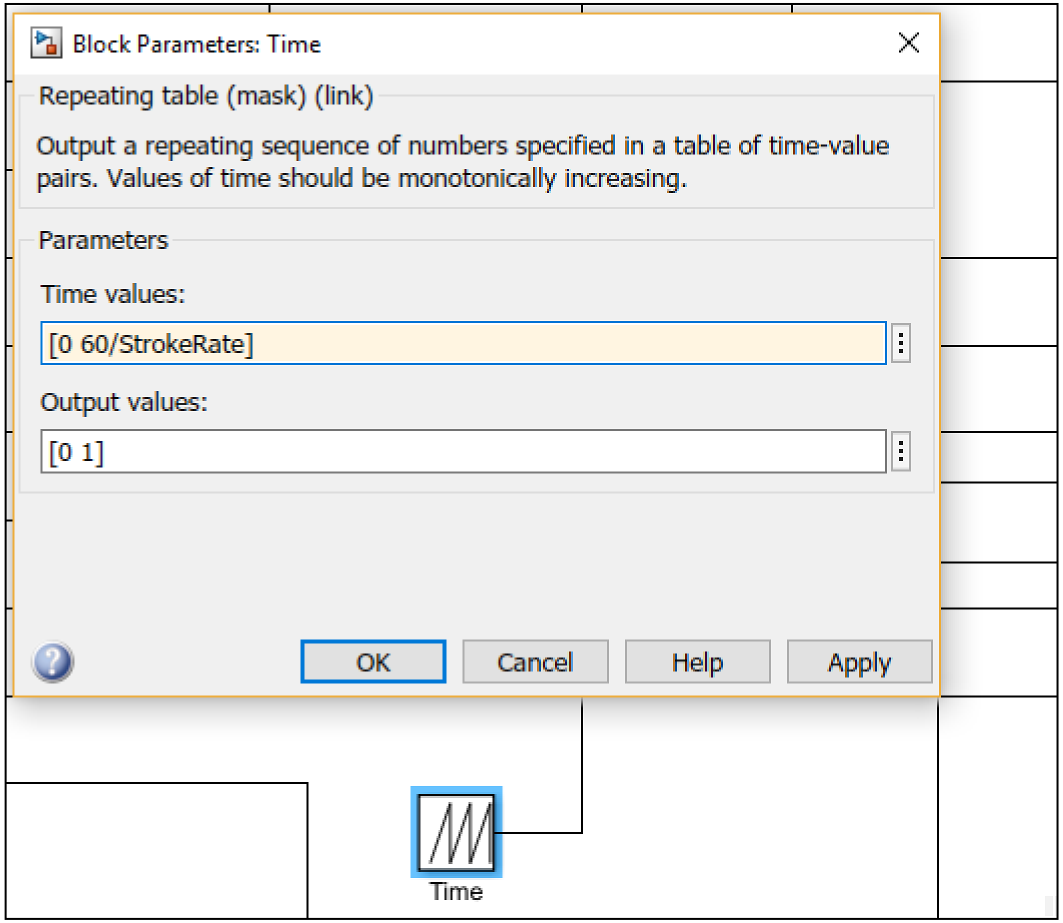


1.1 SetParVals:

```
function [phi1,phi2,sigma,F,rho,tau,p,cycleT,gamma,R,l,L,mH,mR,mu,g]=setParVals

    gamma = 10*pi/180; %Angle of the rower's force [rad]
    rho = 1000; %Density [kg/m^3]
    mu = 0.001; %Dynamic viscosity [Pa/s]
    phi1 = -1.1; %Catch angle [rad]
    phi2 = 0.6; %Finish angle [rad]
    L = 0.8; %Distance covered by the rower respect to the boat [m]
    F = 37; %Rower Force Parameter [N]
    l = 20; %Length of the boat[m]
    mH = 96+70; %Mass of the hull and the parts stationary respect
                %to it [kg]
    mR = 70; %Average mass of a rower [kg]
    R = 8; %Number of rowers []
    sigma = 0.367; %Part of the cycle time with the oar in the water []
    tau = 0.633; %Part of the cycle time with the oar out the water[]
    p = 0.646; %Part of the recovery accelerating []
    g = -9.8; %Acceleration of gravity [m/s^2]
    StrokeR = 40;
    cycleT = 60/StrokeR;
end
```

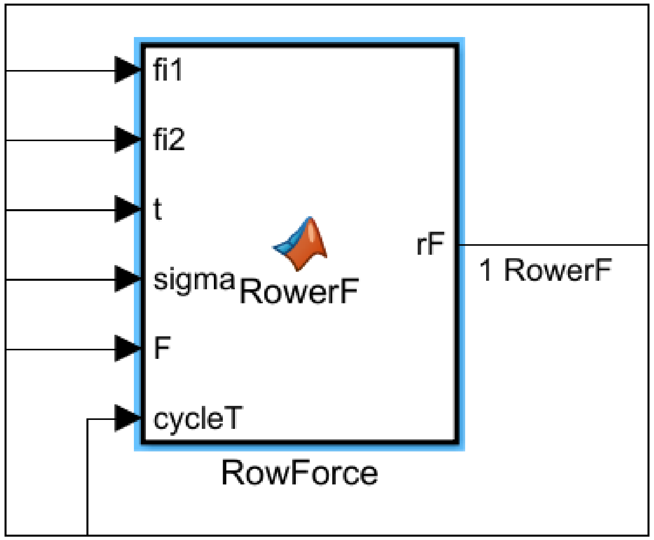


1.2 Time:


Block Parameters: Time
Repeating table (mask) (link)
Output a repeating sequence of numbers specified in a table of time-value pairs. Values of time should be monotonically increasing.
Parameters
Time values:
[0 60/StrokeRate]
Output values:
[0 1]
OK
Cancel
Help
Apply
Time

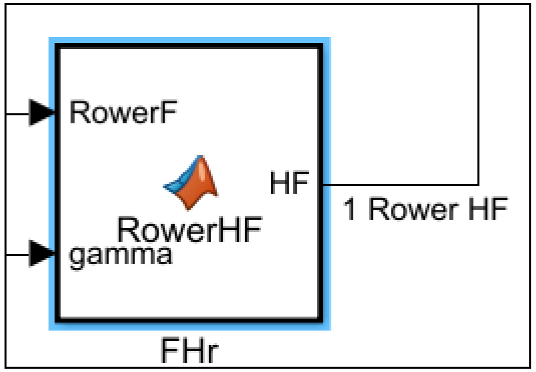
1.3 RowForce:

```
function rF  =  RowerF(fi1, fi2, t, sigma, F, cycleT)
%Function that calculates the rower's force for one rower.
    %one oar per one rower are considered.
%variables used
    %fi1 = catch angle.
    %fi2 = finish angle.
    %t = dimensionless time, ranging from 0 to 1.
    %sigma = portion of the rowing cycle with the oar underwater.
    %F = Rower Force Parameter

    if (t>sigma && t<=1)
        rF = 0;
    elseif (t>=0 && t<=sigma)
       ratio = t/sigma;
       Fmax = F*(-(fi2-fi1)/(sigma*cycleT))^2;
       rF = Fmax*(((fi2-fi1)^2)/(fi1*fi2))*(ratio-1)*ratio;
    elseif (t>1 && t<0)
        disp('error in t');
        rF = 0;
    elseif (sigma>1 && sigma<0)
        disp('error in sigma');
        rF = 0;
    else
        disp('unkown error');
        rF = 0;
    end
end
```



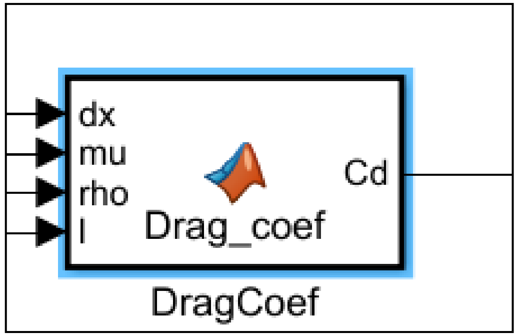
1.4 FHr:

```
function HF  =  RowerHF(RowerF, gamma)
%Function that calculates the rower's horizontal force for one rower.
%variables used
    %RowerF = Rower force.
    %gamma = angle which this total force is applicated.

    HF = RowerF*cos(gamma);
end
```



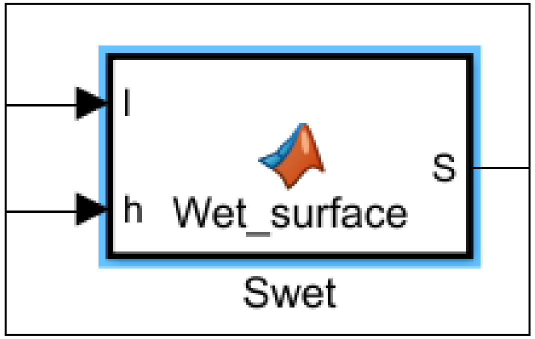
1.5 DragCoef:

```
function Cd    =  Drag_coef(dx, mu, rho, l)
%Function that calculates the drag coefficient of a rowing hull.
    %2D flat plate theory considered.
    %Blasius theory for laminar flow.
    %1/7 theory for turbulent flow.
%variables used
    %dx = velocity of the hull respect to the fluid.
    %mu = dynamic viscosity fluid.
    %rho = density of the fluid.
    %l = length of the boat.

    v = abs(dx);
    ReL = (rho*v*l)/mu;

    if ReL <= 5e5
       Cd = 1.328/sqrt(ReL); %calculation of the drag coefficient
    elseif ReL >5e5
        Cd = 0.027*(7/6)/((ReL)^(1/7));
    else
        Cd = -1;
end
```



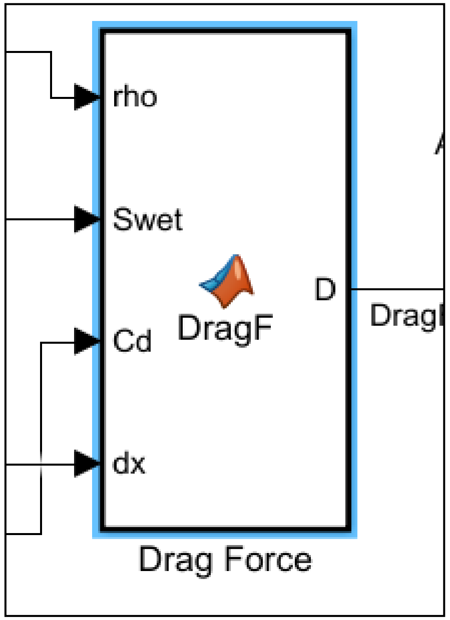
1.6 Swet:

```
function S  =  Wet_surface(l,h)
%Function that calculates the wet surface of a rowing hull.
    %Rectangular surface considered.
    %Curvature factor applied of 1.2.
%variables used
    %l = length of the boat.
    %h = height of the boat.

    if h>0
        CF = 1.2; %Curvature factor to take into account the 3dimensional
                  %shape of the hull
        S = l*h; %One side
        S = S*2; %Both sides
        S = S*CF;
    else
        S = 0;
    end
end
```



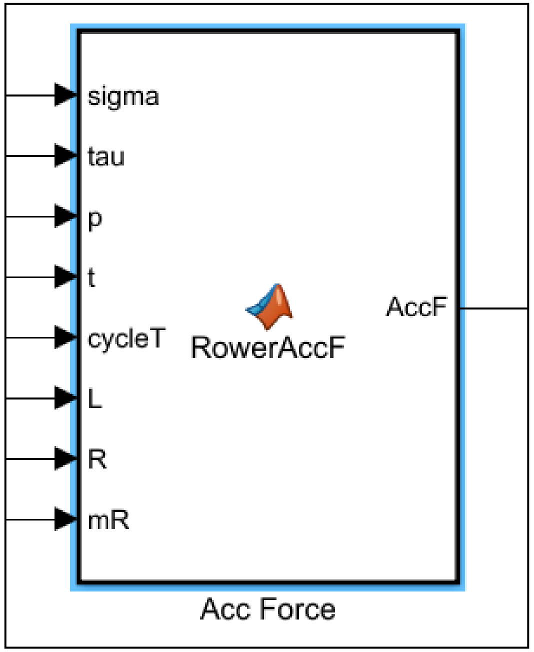
1.7 Drag Force:

```
function D  =  DragF(rho, Swet, Cd, dx)
%Function that calculates the drag force of the boat hull.
%variables used
    %rho = density of the fluid.
    %Swet = surface of the hull in contac with the fluid.
    %Cd = drag coefficient.
    %dx = velocity of the hull respect to the fluid.
    v = abs (dx);

    D = 0.5*rho*Swet*Cd*v^2*(dx/v);
end
```



1.8 Acc Force:

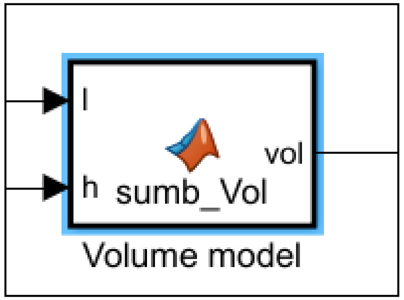
```
function AccF  =  RowerAccF(sigma, tau, p, t, cycleT, L, R, mR)
%Function that calculates the rower's acceleration during a cycle.
    %The acceleration of the rower respect to the hull.
%variables used
    %sigma = portion of the rowing cycle with the oar underwater.
    %tau = portion of the rowing cycle with the oar outside the fluid.
    %p = part of the return when the rower is accelerating.
    %cycleT = time used for one cycle.
    %L = distance travalled by the rower.
    %R = number of rowers.
    %mR = mass of a rower.

    if (t>=0 && t<sigma)
        ACC = ((-(pi^2)*L)/(2*(sigma^2)*(cycleT^2)))*(cos(pi*(1-t/sigma)));
    elseif (t>=sigma && t<(sigma+p*tau))
        ACC = ((-2*L)/(p*(cycleT^2)*(tau^2)));
    elseif (t>=(sigma+p*tau) && t<=1)
        ACC = ((2*L)/((1-p)*(cycleT^2)*(tau^2)));
    elseif (t>1 && t<0)
        disp('error in t');
        ACC = 0;
    elseif (sigma>1 && sigma<0)
        disp('error in sigma');
        ACC = 0;
    elseif (tau>1 && tau<0)
        disp('error in tau');
        ACC = 0;
    elseif (sigma+tau>1)
        disp('error in cycle variables');
        ACC = 0;
    else
        disp('unkown error');
        ACC = 0;
    end
    AccF = ACC*mR*R;
```



```
end
```

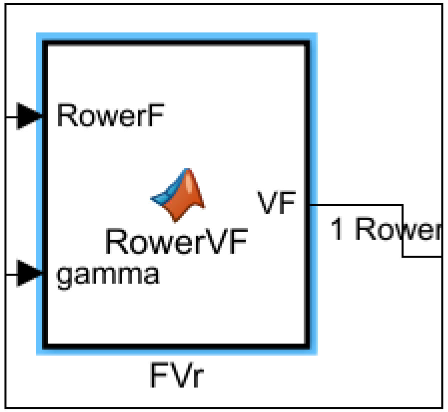
1.9 Volume model:

```
function vol  =  sumb_Vol(l,h)
%Function that calculates the horizontal surface of a rowing hull.
    %Diamond-shape surface considered.
    %the with is considerted parabolic with the height.
    %the length is considered constant with the height.
    %With values: w=0 h=0 // w=0.7 h=0.4 // w=0.6 h=0.2.
%variables used
    %l = length of the boat.
    %h = height of the boat.

    if h<=0
        vol = 0;
    else
        vol = 0;
        H = linspace(0,h);%Vector of heights in order to integrate the
                          %volume with a surface function.
                          %The resolution is 100 times the value of h
        for i = 2:(length(H)) %Numerical integration of the volume with the
                              %horizontal surface as a function of the height
            meanH = (H(i)+H(i-1))/2;
            diffH = H(i)-H(i-1);
            vol = vol + ((-25/4)*meanH^3 + (17/4)*meanH^2 + l*meanH/2)*diffH;
        end
    end
end
```



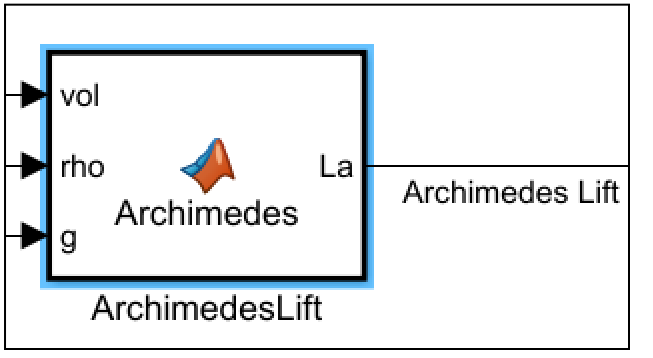
1.10 FVr:

```
function VF  =  RowerVF(RowerF, gamma)
%Function that calculates the rower's vertical force for one rower.
%variables used
    %RowerF = Rower total force.
    %gamma = angle which this total force is applicated.

    VF = RowerF*sin(gamma);
end
```

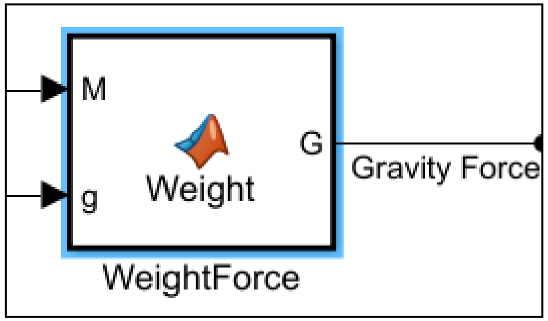


1.11 ArchimedesLift:

```
function La  =  Archimedes(vol, rho, g)
%Function that calculates the archimedes force for a body.
    %constant density and gravity considered.
    %the displaced volume is given by some other function.
%variables used
    %vol = submerged volume of the boat
    %rho = density of the fluid.
    %g = acceleration of gravity.

    La = rho*g*vol;
end
```

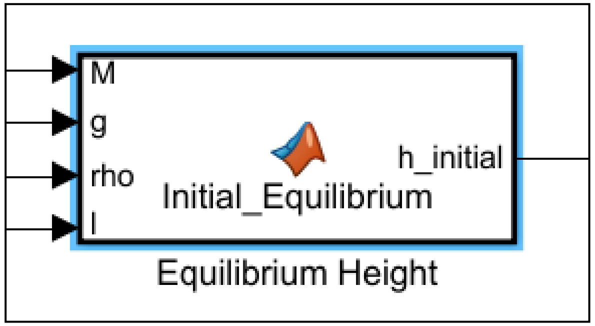


1.12 WeightForce:




```
function G  =  Weight(M, g)
%Function that calculates the gravity force for a rowing boat.
    %constant density and gravity considered.
%variables used
    %M = Total mass of the sum of forces.
    %g = acceleration of gravity.

    G = (M)*g; %Gravity force
end
```

1.13 Equilibrium Height:




```
function h_initial  =  Initial_Equilibrium(M, g, rho, l)
%Function that calculates the equilibrium height considering only gravity
%and archimedes lift, when the speed is 0.
%variables used
    %l = length of the boat.
    %g = acceleration of gravity.
    %M = total mass of the system.
    %rho = density of the water.

G = M*g;
volRef = G/(rho*g);
h = 0;
diff = 1;
    while abs(diff)>0.1
        vol = 0;
        H = linspace(0,h);%Vector of heights in order to integrate the
                       %volume with a surface function.
                       %The resolution is 100 times the value of h
            for i = 2:(length(H))
                meanH = (H(i)+H(i-1))/2;
                diffH = H(i)-H(i-1);
                vol = vol + ((-25/4)*meanH^3 + (17/4)*meanH^2 + l*meanH/2)*diffH;
            end
        diff = (volRef-vol)*100/volRef;
        if abs(diff) > 0.1
            h = h + 0.01*(volRef-vol);
        end
    end
h_initial = -h;
end
```