CONTROLO DA ORIENTAÇÃO DE UM MÍSSIL (MATLAB FILE)

CONTROLO PROPORCIONAL /PILOTO AUTOMÁTICO DE 3 LOOP´S

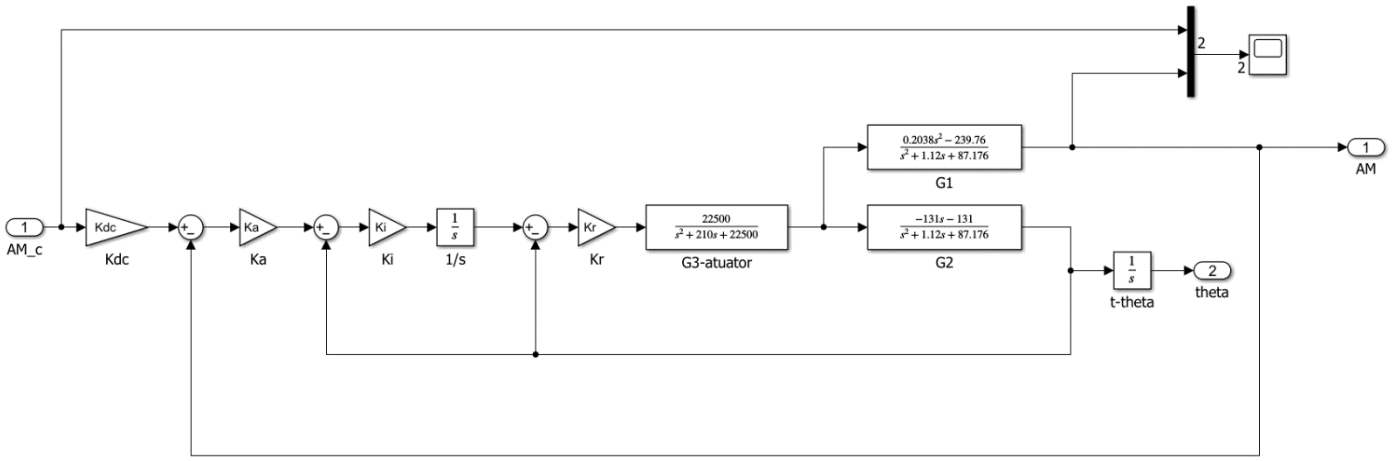
Condições Iniciais

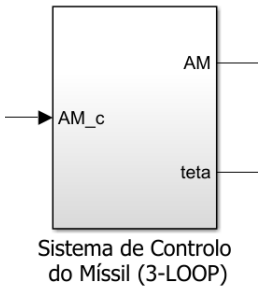
Num ficheiro MatLab são colocadas as condições iniciais da simulação.

```
%Condições Iniciais
%S.I Unidades (radiano, graus, metro)
d2r = pi/180;

%ALVO
V_T = 600;
X_T0 = 100;
Z_T0 = 20000;
l_T0 = -2.5*d2r;

%MISSIL
X_M0 = 2000;
Z_M0 = 0;
l_M0 = 80*d2r;

%GANHOS (do piloto automático de 3 Loops.)
Kdc = 1.1;
Ka = 4.5;
Ki = 14.3;
Kr = -0.37;
```

Sistema de Piloto Automático com Controlo de Aceleração

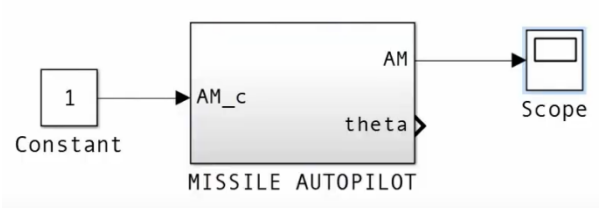
Vamos utilizar as funções de transferência e o atuador definidos no *Paper 2*, juntamente com os parâmetros do míssil e os loops de malha fechada.

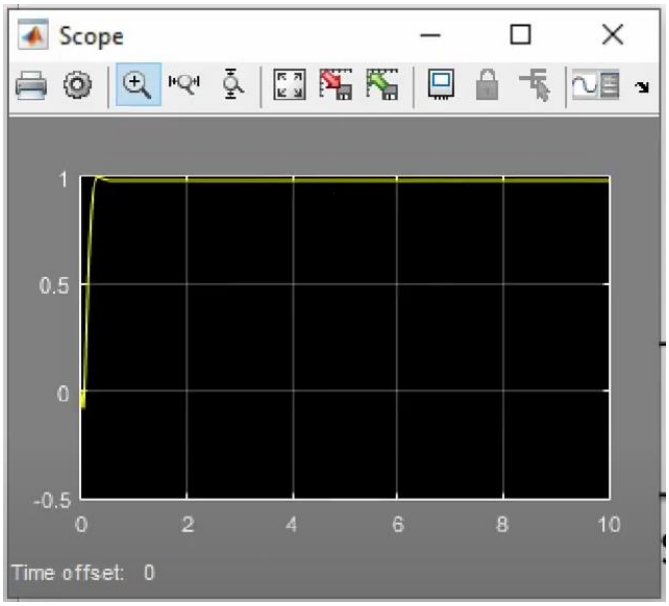




O sistema de controlo pode ser transformado num bloco só, com input o comando da Aceleração do Míssil e como output a aceleração do míssil e o seu ângulo de inclinação, θ.

Podemos fazer um pequeno teste para verificar se o comportamento vai de acordo com o esperado, colocando um degrau à entrada e observando a saída.





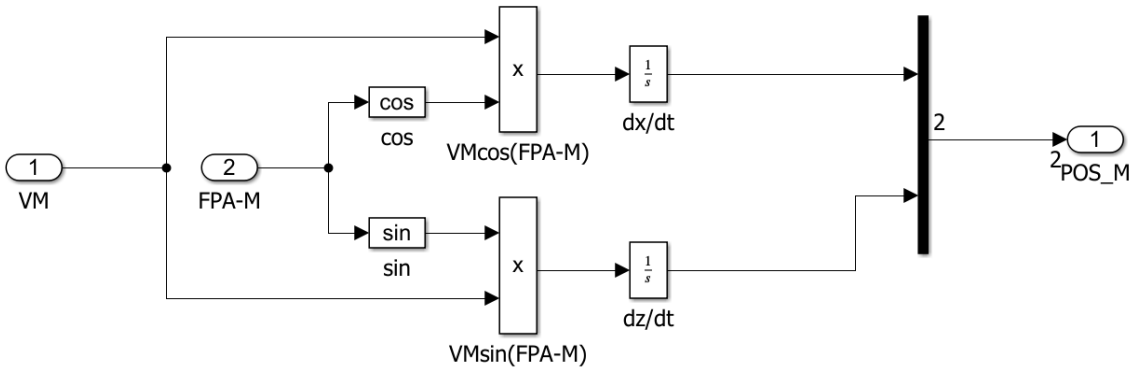
O resultado obtido vai de acordo com o resultado esperado definido no *Paper 2*, em que inicialmente temos uma força negativa e de seguida o míssil corrige a sua trajetória e vai de encontro ao comando de aceleração pretendido.

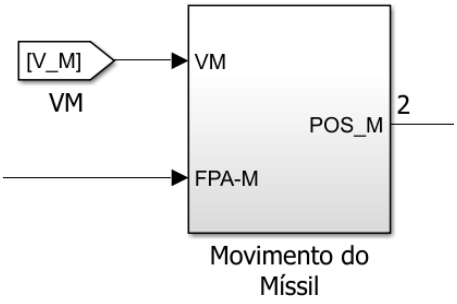
Modelo Matemático do Míssil

Agora vamos especificar o movimento do míssil de acordo com as suas equações da posição.

$$x_M = V_m \cdot \cos(\lambda_M)\,; \quad z_M = V_m \cdot \sin(\lambda_M)$$

Implementação das equações em diagramas de blocos.





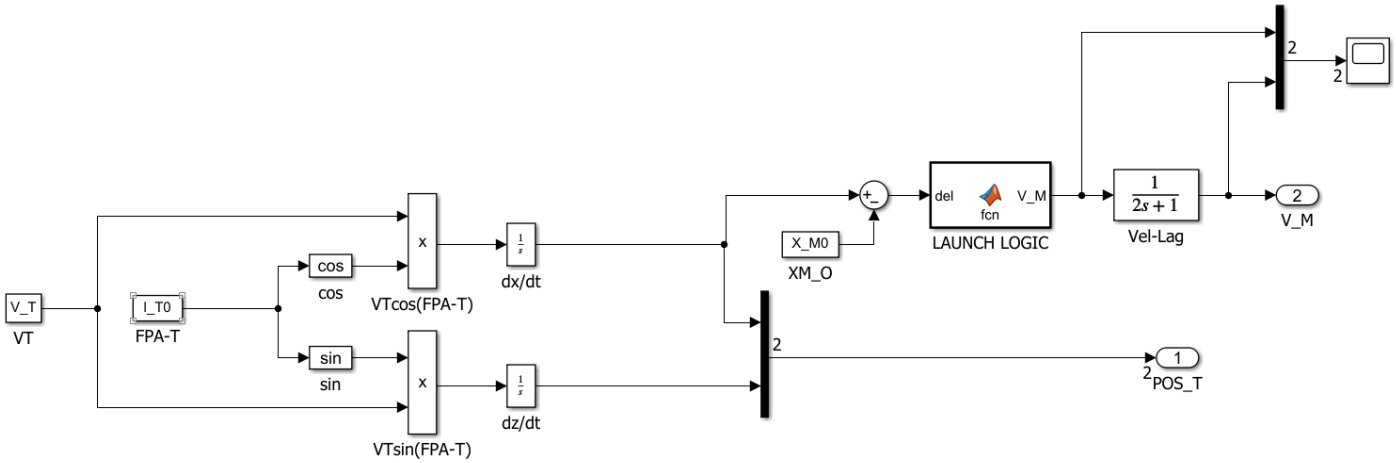
Este mesmo conjunto pode ser transformando num bloco, com inputs a velocidade do míssil, V_m e o ângulo de trajetória de voo do míssil, FPA-M, tendo como output a posição do míssil, POS-M.

Modelo Matemático do Alvo

Agora vamos realizar o mesmo processo, mas para o alvo. O alvo é regido pelas mesmas equações.

$$x_T = V_T \cdot \cos(\lambda_T)\,; \quad z_M = V_T \cdot \sin(\lambda_T)$$

Implementação das equações em diagramas de blocos.



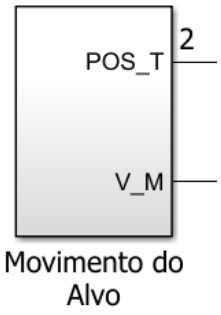
Neste sistema a velocidade do míssil será calculada baseado numa função do MatLab, o *Launch Logic*. A mesma função define que assim que o míssil esteja a uma distância de 200m do alvo, o míssil é lançado.

Para a velocidade do míssil existe um atraso, pois o míssil não atinge a velocidade terminar de imediato (Vel-Lag).

A função *Launch Logic* é definida com o seguinte código:

```
function V_M = fcn(del)
%#codegen

persistent f
persistent Vel

if isempty(f)
    f = 0;
end
if isempty(Vel)
    Vel = 0;
end

if f == 0 && del > -200
    f = 1;
    Vel = 1021;
end

V_M = Vel;
```

O output da função é a velocidade do míssil e o input será a distância entre o míssil e o alvo. Portanto se **del** for menor do que 200 m nós lançamos o míssil (**f=1**) com velocidade igual a 1021 m/s.



Mais uma vez podemos criar mais um subsistema definido como Movimento do Alvo. O mesmo tem como inputs os parâmetros iniciais e outputs a posição do alvo, POS_T, e a velocidade do míssil, V_M.

Bloco com a Lei de Comando de Navegação Proporcional

Agora vamos criar o sistema de comando de Navegação Proporcional, o mesmo vai calcular o comando de aceleração para o míssil baseado na Lei de Comando vista no *Paper 2*.

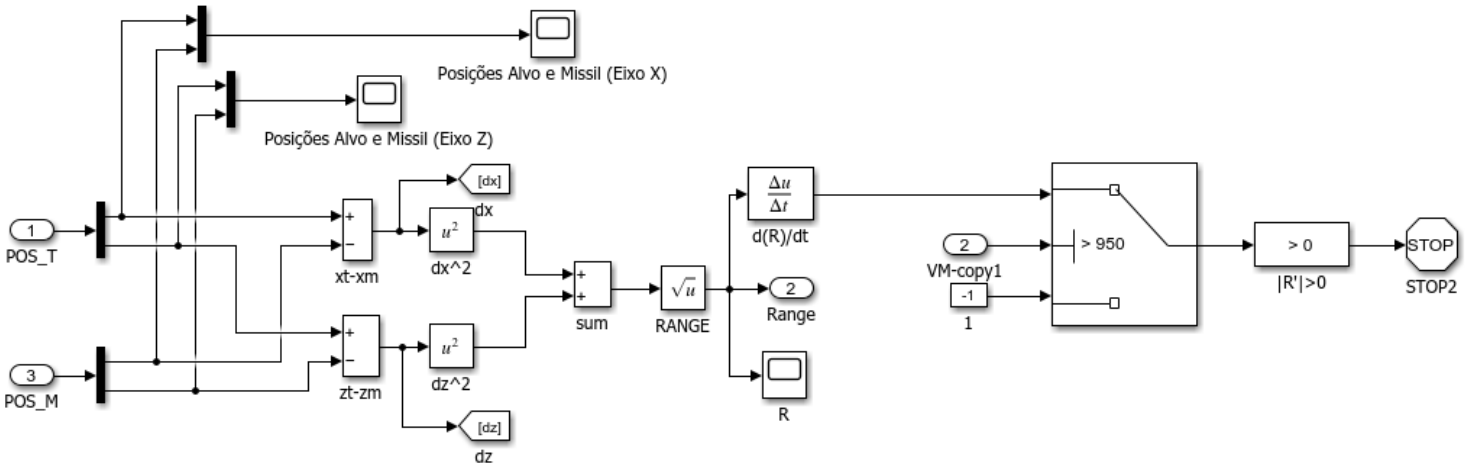
Inicialmente criamos um subsistema que calcula a distância entre o alvo e míssil consoante as suas respetivas posições.

$$R = \sqrt{(x_M - x_A)^2 + (z_M - z_A)^2}$$

Importante mencionar que após ser obtido a distância R é feito uma derivada em ordem ao tempo, e de seguida é colocado um switch em que consoante a velocidade do míssil for ou não superior a 950 m/s é testado se a derivada de R é ou não superior a zero, isto é, se a distância entre o míssil ao alvo está a aumentar ou não.

Caso a derivada de R seja maior que zero a simulação será parada, caso contrário irá continuar até a condição se verificar.

Implementação da equação anterior e respetivos raciocínios em diagrama de blocos.



Passando para a Lei de Controlo de Navegação Proporcional temos:

$$a_{M_c} = \mathrm{N} V_c \dot{\lambda}\,; \quad -0.3 < a_{M_c} < 0.2$$

Para calcular o comando de aceleração perpendicular à velocidade do míssil, a_{M_c}, utilizamos o seguinte diagrama de blocos, que recebe como input a variação do Ângulo de Trajetória de Voo, $\dot{\lambda}$, a velocidade do míssil, V_c, e a constante de navegação proporcional, N (=4 na simulação).

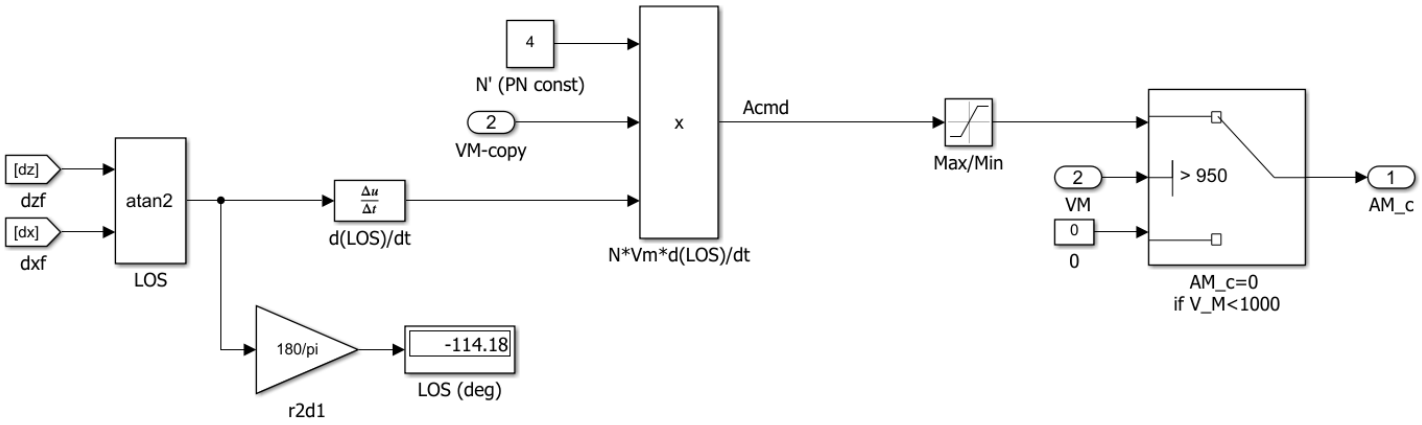
O comando de aceleração perpendicular à velocidade do míssil encontra-se saturada entre -0.3 e 0.2 por questões de segurança.

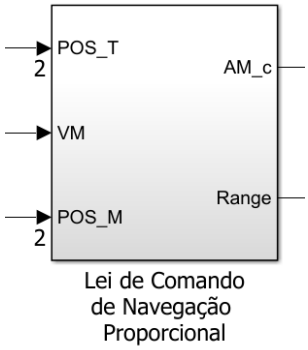
O ângulo da trajetória de voo é calculado segundo a seguinte expressão:

$$\lambda = \arctan\left(\frac{\Delta z}{\Delta x}\right) = \arctan\left(\frac{z_M - z_A}{x_M - x_A}\right)$$

O comando de aceleração do míssil será diferente de zero quando velocidade do míssil for superior a 950 m/s.

Implementação das equações anteriores e respetivos raciocínios em diagrama de blocos.

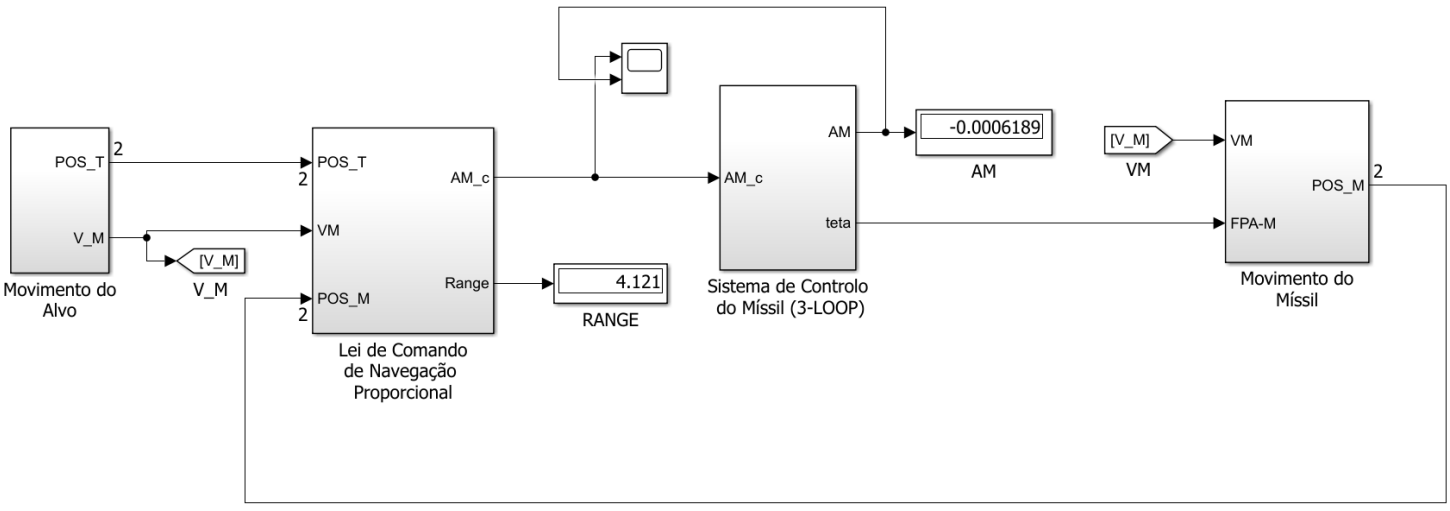




De seguida, juntamos os dois subsistemas em um só bloco, o mesmo toma como inputs a posição do objeto, POS_T, a posição do míssil (que virá da malha fechada), POS_M, e a velocidade do míssil, VM, e tem como output o comando de aceleração do míssil, AM_c e a distância entra o alvo e o míssil, Range.

<u>Sistema de Simulação Completo</u>

Após contruídos todos os blocos necessários é nós possível realizar as respetivas ligações e obter o seguinte ambiente de simulação:



Após corrida a simulação conseguimos, de acordo com os parâmetros estipulados, verificar que a detonação do míssil ocorre a uma distância de 4.121 m do míssil. De seguida é apresentada a variação da posição Z do míssil e do alvo de modo a constatar que ocorreu interseção entre os dois.