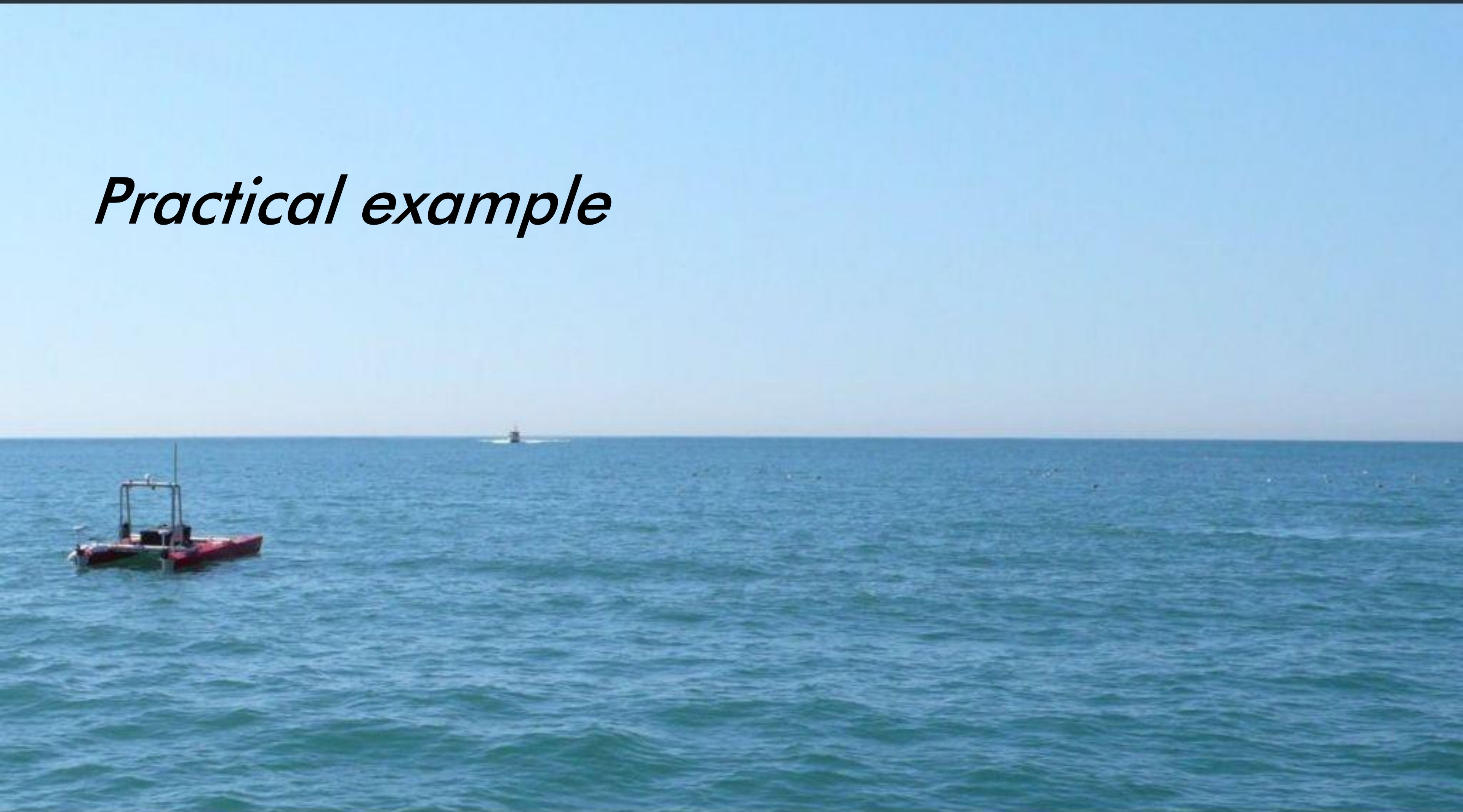
Marine Robotics

Unmanned Autonomous Vehicles in Air Land and Sea

Politecnico Milano – June 2016

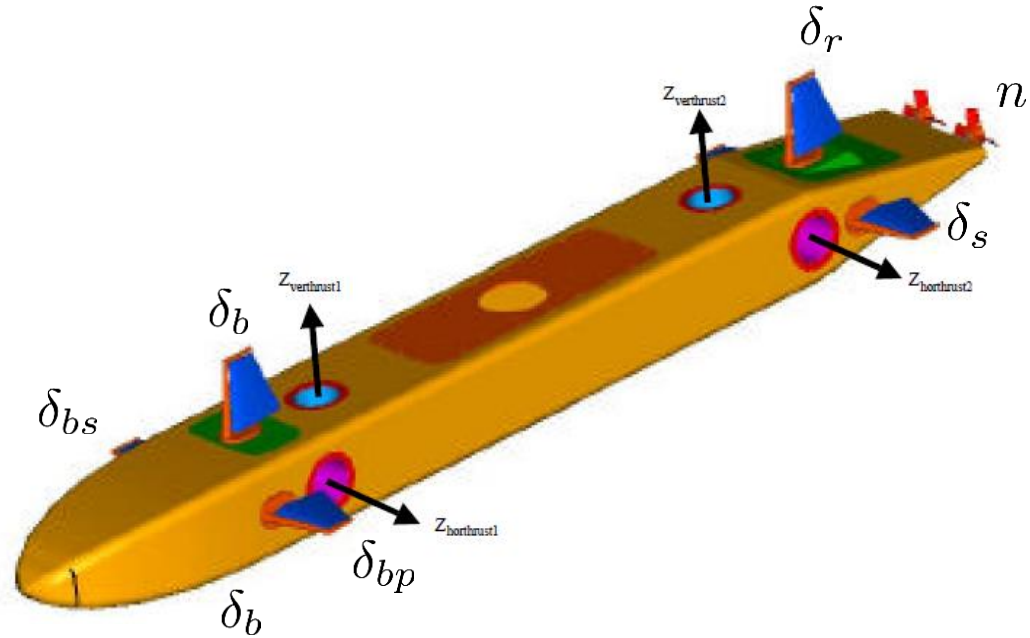
Alfredo Martins
INESC TEC / ISEP
Portugal
alfredo.martins@inesctec.pt

Practical example



NPS Phoenix AUV

- Classic AUV
- Full 6DOF AUV model
- Additional fin control when comparing to comparing to torpedo shaped AUVs (like REMUS)
- Very complete hydrodinamic model available



NPS Phoenix vehicle, Figure from [1]

Vehicle controls:

δ_r delta_r = rudder angle

δ_s delta_s = port and starboard stern plane

δ_b delta_b = top and bottom bow plane

δ_{bp} delta_bp = port bow plane

δ_{bs} delta_bs = starboard bow plane

propeller shaft speed

[1] J. Riedel et al." Design and Development of Low Cost Variable Buoyancy System for the Soft Grounding of Autonomous Underwater Vehicles“

Matlab model: npsauv.m

```
function [xdot,U] = npsauv(x,ui)
%
% [xdot,U] = NPSAUV(x,ui) returns the speed U in m/s (optionally)
%  and the time derivative of the state vector:
%  x = = [ u v w p q r x y z phi theta psi ]'
```

x = [u v w p q r x y z phi theta psi]‘

ui = [delta_r delta_s delta_b delta_bp delta_bs]’

State and controls

```
u     = surge velocity  (m/s)
v      = sway velocity  (m/s)
w      = heave velocity (m/s)
p      = roll velocity  (rad/s)
q      = pitch velocity (rad/s)
r      = yaw velocity   (rad/s)
xpos  = position in x-direction (m)
ypos  = position in y-direction (m)
zpos  = position in z-direction (m)
phi   = roll angle (rad)
theta = pitch angle (rad)
psi   = yaw angle   (rad)

delta_r  = rudder angle (rad)
delta_s  = port and starboard stern plane(rad)
delta_b  = top and bottom bow plane       (rad)
delta_bp = port bow plane                 (rad)
delta_bs = starboard bow plane            (rad)
n        = propeller shaft speed          (rpm)
```

Usage, simrun.m

```
for i=1:N+1,
    time = (i-1)*h;          % simulation time in seconds

    % control system
    % u a function of state to be defined
    % u       = [ delta_r delta_s delta_b delta_bp delta_bs n ]



    % ship model
    [xdot,U] = npsauv(x,u);

    % store data for presentation
    xout(i,:) = [time,x',U];

    % numerical integration
    x = euler2(xdot,x,h);              % Euler integration
end
```

Tasks

In Matlab M-file:

Step 1. Implement and simulate a decoupled steering controller

(leave other modes unactuated)

Step 2. Implement and simulate full decoupled speed, steering and diving controllers

Step 3. Using previous controllers implement a horizontal guidance law

Matlab/Simulink S-functions

- Allow generic block definition in Simulink
- In simulink the corresponding block is s-function
- Function with common interface, result interpreted according with passing parameter flag (choosen by simulink), return depend on the flag value

```
function [sys,x0,str,ts] = name(t,x,u,flag)
```

- m file with system description:
  - number of states, inputs and outputs (flag =0)
  - initial state, and sample time (when applicable)
  - output equation (flag=3)
  - discrete sate update (flag=2)
  - continuous state derivatives (continuous dynamics) (flag=1)
  - next sample time (for variable sample times) (flag=4)

Matlab/Simulink S-Functions

```
function [sys,x0,str,ts] = funcao(t,x,u,flag,param)

switch (flag),

case 0, % inicializaçao
     sizes=simsizes;

     sizes.NumContStates= 0;
     sizes.NumDiscStates= 0;
     sizes.NumOutputs= 0;
     sizes.NumInputs= 0;
     sizes.DirFeedthrough= 0;
     sizes.NumSampleTimes= 1;

     sys=simsizes(sizes);
     x0= [];  %estado inicial
     str=[];  % ordem dos estados
     ts =[0 0]; %sampling time [periodo offset]
```

```
case 1, % derivadas do estado contínuo
     sys=[ …..]; %derivadas do estado

case 2, % update estados discretos
     sys = [….]; %novos estados

case 3, % saidas
     sys = [….]; %saidas

case 4, % proximo sample time
     sys =[ ….] % proximo sample time

end
```

Matlab/Simulink S-Functions

- may be implemented by:
  - m-files
  - C MEX (compiled code)

- dynamics can be arbitrary (depends on the code executed with (flag=1)

Task 2 - Simulink

Step 1. Implement a Simulink block simulating the vehicle using the previous m-file

Step 2. Implement the previous controllers in Simulink

Step 3. Implement the guidance law into the Simulink model