**NATIONAL UNIVERSITY OF SINGAPORE**

**CEG5303: Intelligent Autonomous Robotic Systems**

**Lab: Guidance for Marine surface vessels**

Instructions: Simple Example

In this part, the motion control problem is implemented for marine surface vessel with backstepping control.

You are required to:

* Select a specific type of marine surface vessel, collect relevant information from internet, describe the structure features of it in your report, and build its 3-DOF dynamic model.
* If you want to take the challenge of modeling of six degrees of freedom marine vessel, you are welcome to choose other underwater intelligent robotic systems like ROV and AUV.
* Implement the backstepping to control the marine surface vessel, so that it can track the desired input trajectory and reduce the tracking error to a certain interval at the stable steady state (≤1% of the peak value of the desired trajectory).
* Implement any other control methods of your choice.

Consider a 3-DOF model of a marine surface vessel.

A diagram of a boat with directions

Description automatically generated

A diagram of a geometrical figure

Description automatically generated

The dynamic model of surface vessel as



where

 



A black text on a white background

Description automatically generated

Backstepping controller

Step 1: We set





Then we choose Lyapunov function



And from the dynamic model, we have



Designing the virtual control as



Finally, we can get



Step 2: Then we choose Lyapunov function



The time derivative of Z2 is



And we have



Then, we design the control law as:



**Stability Analysis**





Example MATLAB implementation:

Open “MarineVessel.m”

Locate TODO\_1 and set the feedforward backstepping controller gains and :

Example:

% feedforward gain(adjustable)

K1=diag([50 50 70]);

K2=diag([6000000 6000000 6000000000]);

The desired trajectory is set to be the sinusoid for all states. Feel free to change it.

Locate TODO\_2 and set the mass matrix and Coriolis matrix:

Example:

%Set Mass matrix

M=[mass 0 0;...

0 mass 0;...

0 0 Iz];

% set coriolis matrix

C=[0 -mass\*v(3,i-1) 0;...

mass\*v(3,i-1) 0 0;...

0 0 0];

Locate TODO\_3 and set the rotation transformation matrix, its’ derivative and the inverse of its’ derivative:

Example:

% set rotation transformation matrix

psi=eta(3,i-1);

dpsi=deta(3,i-1);

J=[cos(psi) -sin(psi) 0;sin(psi) cos(psi) 0;0 0 1];

dJ=[-sin(psi)\*dpsi -cos(psi)\*dpsi 0;...

cos(psi)\*dpsi -sin(psi)\*dpsi 0;...

0 0 0;];

dinvJ=[-sin(psi)\*dpsi cos(psi)\*dpsi 0;...

-cos(psi)\*dpsi -sin(psi)\*dpsi 0;...

0 0 0;];

Locate TODO\_4 and set the backstepping error, z1, dz1, alpha, z2 and dalpha:

Example:

% set backstepping error

z1=[eta(1,i-1)-eta\_d(1,i-1);eta(2,i-1)-eta\_d(2,i-1);eta(3,i-1)-eta\_d(3,i-1)];

dz1=[deta(1,i-1)-deta\_d(1,i-1);deta(2,i-1)-deta\_d(2,i-1);deta(3,i-1)-deta\_d(3,i-1)];

alpha=J'\*(deta(1,i-1)-K1\*z1);

z2=v(:,i-1)-alpha;

dalpha=dinvJ\*(deta\_d(:,i-1)-K1\*z1)+J'\*(ddeta\_d(:,i-1)-K1\*dz1);

Locate TODO\_5 and calculate the control input:

Example:

% set real time input signal

Tau(:,i)=-J'\*z1-K2\*z2+C\*v(:,i-1)+g+M\*dalpha;

After finishing the code, run the MATLAB script to see the result.