2010 American Control Conference
Marriott Waterfront, Baltimore, MD, USA
June 30-July 02, 2010

Command Filtered Backstepping Design in MOOS-IvP Helm Framework for Trajectory Tracking of USVs

Vladimir Djapic[2] and Dula Nad[1,2]
[1]University of Zagreb, Faculty of Electrical Engineering and Computing, Zagreb, Croatia.
[2]Work performed while at NATO Undersea Research Centre (NURC), La Spezia, Italy.
[2]NATO Undersea Research Centre (NURC), La Spezia, Italy.

Abstract— **This article describes design and simulation implementation of a nonlinear controller for an underactuated surface vehicle. The controller is designed using a command filtered backstepping (CFBS) approach. Theoretical background for controller design is given in the first part of this article. This nonlinear controller can be used for accurate tracking of a complex trajectory, for example a circular trajectory. Second part of the article focuses on implementation in the MOOS-IvP framework. This framework allows for flexibility in control and mission planning. Guidance is covered by the MOOS-IvP implementation of the controller while the COTS autopilot handles low-level control. The control performance is verified in simulation which confirms arbitrarily small tracking error. This paper presents simulation results where external disturbances, such currents, are also simulated and compensated for.**

I. INTRODUCTION

Different approaches for motion control of autonomous vehicles (land, air, surface, and underwater robots) have been analyzed in recent past [11], [2].

The literature, generally, distinguishes among two different motion control problems:

1) path following - where the robot is required to converge to and follow a path where only spatial convergence is necessary without any temporal requirement, and
2) path/trajectory tracking - where the robot is required to track a time parametrized reference with temporal requirement.

Recently, the concept of path maneuvering was introduced in order to combine the properties of trajectory tracking and path following [2]. This problem is solved for the fully actuated systems and solutions can be found in the nonlinear control textbooks, such as in [8], pages 540-544. Even though fully actuated systems are able to independently control the motion of all their DOFs simultaneously they are impractical for vehicles moving at speeds above 1.5 - 2 $\frac{m}{s}$ since they would usually expend an unnecessary amount of energy for control action [2]. The authors in [2] have addressed the subject of straight-line high speed target tracking for unmanned surface vehicles (USVs) while here, in addition to straight-line tracking, we show the tracking of the arbitrary, smooth, complex trajectories with a USV. Thus, we focus on trajectory tracking which forces the system to follow a given point as it moves along an operator (or sensor) defined trajectory. The controller generates yaw rate and velocity commands. The velocity and yaw rate controller generates the force and torque commands to achieve the yaw rate and velocity commands.

Unmanned Surface Vehicles (USVs) are being considered for the following missions: Mine Countermeasures (MCM), Anti-Submarine Warfare (ASW), and Maritime Security (MS). NURC's USV Mandarina is currently used by the MCM and Port Protection Programs [10]. For some searching applications it is a requirement for a vehicle to accurately follow a specific trajectory, make accurate turns and continue to follow the next specified trajectory. Many of these missions require the vehicle to function in complex Nada cluttered environments.

The researchers at the Mobile Robotics Research Group at Oxford University, the Computer Science and Artificial Intelligence Lab and Dept. of Mechanical Eng. at MIT, and the Naval Undersea Warfare Center in Newport Rhode Island (NUWC-NPT) have developed the MOOS-IvP Autonomy Architecture [9], [1]. MOOS stands for "Mission Oriented Operating Suite", and IvP stands for "Interval Programming". This architecture consists of an open-source distributed autonomy architecture and an approach to behavior based control of autonomous vehicles that allows reactive control in complex environments with multiple constraints. Low-level control tasks such as navigation, depth keeping and vehicle safety are assigned to the AUV main vehicle computer, all high-level control inputs are derived from a separate vehicle payload computer running the MIT MOOS-IvP system. Our goal is to use the existing MOOS open-source features and its ability to dynamically react to its environment in order to increase the functional autonomy of the existing autonomous vehicles. One such an example is described in Section IX where an output from a sonar sensor is used to direct the robot to change its trajectory as the new mission plan is developed onboard the vehicle in response to the sensor data. In addition, this architecture enabled us to implement an advanced nonlinear controller based on CFBS onboard of the NURC's Mandarina USV. The program executing mission planning and control algorithms for stable trajectory tracking is interfaced with the COTS autopilot SPECTRE made by H-Scientific Ltd.

The paper is organized as follows. Section II introduces the problem of trajectory tracking control for a USV. Section II defines the USV dynamics. Section III outlines the control law signals. Sections IV, V, VI, and VII present a detailed

978-1-4244-7427-1/10/$26.00 ©2010 AACC

derivation of trajectory tracking control laws. Sections VIII and IX present the USV and MOOS implementation. The performance of the control system proposed is illustrated in simulation in Section X. Finally, Section XI contains the conclusions and describes some problems that warrant further research.

II. Problem Statement

The kinematic and dynamic equations for an unmanned surface vehicle are described as

$$\dot{x} = u\cos(\psi) - v\sin(\psi) \quad (1)$$
$$\dot{y} = u\sin(\psi) + v\cos(\psi) \quad (2)$$
$$\dot{\psi} = r \quad (3)$$
$$\dot{u} = g(u,r) + F \quad (4)$$
$$\dot{r} = f(u,r) + \tau \quad (5)$$

where x and y are the earth relative position, ψ is the yaw angle, u and v are the horizontal and lateral velocities in body frame, r is the yaw rate in body frame, F is the body-frame control force, τ is the body-frame control moment, $g(u,r)$ and $f(u,r)$ are friction and other forces acting on the robot. Since the USV is underactuated the lateral velocity v cannot be directly controlled but since it can be measured it will be the input to our control law. For this article, we assume unit values for the mass and inertia. Accounting for non-unit values is straightforward.

We are interested in the trajectory tracking problem where the objective is to force the system output $z(t) = [x(t), y(t)]^\top \in \Re^2$ to track a desired ideal output $z_d(t) = [x_c(t), y_c(t)]^\top \in \Re^2$. We use the CFBS approach [5], [4], [3], [6], [7] assuming that z_d and $\dot{z}_d$ are available as inputs to the control law.

III. Control Signal Implementation

This section summarizes the control law and the stability properties of the closed loop system. The following equations describe the control signals

$$u_c^o = u_{x_c}^o cos(\psi_c^o) + u_{y_c}^o sin(\psi_c^o) \quad (6)$$
$$\psi_c^o = atan2\left(-u_{x_c}^o,\ u_{y_c}^o\right) + \beta \quad (7)$$
$$r_c^o = -K_\psi\tilde{\psi} + \dot{\psi}_c - \psi_{bs} \quad (8)$$
$$F = -g(u,r) - K_u\tilde{u} + \dot{u}_c - u_{bs} \quad (9)$$
$$\tau = -f(u,r) - K_r\tilde{r} + \dot{r}_c - r_{bs}, \quad (10)$$

where β, $u_{x_c}^o$ and $u_{y_c}^o$, ψ_{bs}, u_{bs}, and r_{bs} are defined in eqns. (15), (18), (32), (33), and (34), respectively. The symbols K_ψ, K_u, and K_r represent positive design parameters. In addition, the control law implements the signals ξ_x, ξ_y, and ξ_ψ using eqns. (22) and (26). Including the three command filters defined below, the controller has nine states: ξ_x, ξ_y, ξ_ψ, ψ_c, $\dot{\psi}_c$, u_c, $\dot{u}_c$, r_c and $\dot{r}_c$. The control law is derived in sections IV, V, and VI.

We will use a certain subscript and superscript notation throughout the article. For example, in addition to the variable u, we introduce the variables u_c^o and u_c. The symbol u_c^o represents the ideal desired value for u. The symbol u_c represents a filtered version of u_c^o. The filter, with bandwidth determined by a parameter ω_n, is defined in Appendix I. This notation will also be used similarly to define ψ_c^o, ψ_c, r_c^o, and r_c. Given this notation, the tracking error variables are defined as

$$\tilde{x} = x - x_c \qquad \tilde{u} = u - u_c$$
$$\tilde{y} = y - y_c \qquad \tilde{r} = r - r_c$$
$$\tilde{\psi} = \psi - \psi_c$$

If $\psi_c = \psi_c^o$, $u_c = u_c^o$, and $r_c = r_c^o$, then eqns. (6 – 10) would implement a conventional backstepping control law. However, the conventional backstepping approach would require analytic expressions for $\dot{\psi}_c^o$, $\dot{u}_c^o$, and $\dot{r}_c^o$, which can be quite complicated, especially when the designer chooses K_{xy} as a function of the state as in Appendix II. The CFBS approach avoids the analytic derivation of these expressions by the use of filters. The approach is designed to maintain the exponential stability properties of the backstepping approach for a set of compensated tracking errors denoted by $\nu_x, \nu_y, \nu_\psi, \nu_u$, and ν_r, and to ensure that the control signals ψ_c, u_c, and r_c (see (21), (25), (29), and (30)) are the same as those of the conventional backstepping approach to within an error proportional to $\frac{1}{\omega_n}$. The closed loop system has the stability properties stated in Theorems 1 and 2 in [6], [7], [5], [3]. Theorem 1 shows that the compensated tracking errors of the 2D CFBS approach have the same properties as the tracking errors of the standard backstepping approach. Theorem 2 shows that, by increasing the command filter natural frequency ω_n, the solution to the CFBS closed-loop system can be made arbitrarily close to the backstepping solution that relies on analytic derivatives. The proofs of these Theorems are given in [6], [7], [5], [3].

IV. Trajectory Following

The inputs to the position control loop are $x_c(t)$, $y_c(t)$, and the derivatives, $\dot{x}_c$, $\dot{y}_c$. We assume that

$$\left\| \begin{array}{c} \dot{x}_c \\ \dot{y}_c \end{array} \right\| \geq \epsilon > 0$$

This section is concerned with the control of $[x, y]$ by specification of desired values for $[u, \psi]$.

A. Notation Definition

For clarity, we rewrite position dynamics as

$$\left[\begin{array}{c} \dot{x} \\ \dot{y} \end{array} \right] = \left[\begin{array}{c} u_x \\ u_y \end{array} \right]$$

where

$$\left.\begin{array}{rcl} u_x &=& u\ \mathrm{c}\psi\ -\ v\ \mathrm{s}\psi \\ u_y &=& u\ \mathrm{s}\psi\ +\ v\ \mathrm{c}\psi, \end{array}\right\} \quad (11)$$

where the symbols $c\psi$ and $s\psi$ represent $cos(\psi)$ and $sin(\psi)$. We also defined this same function for the commanded variables:

$$\left.\begin{array}{rcl} u_{x_c}^o &=& u_c^o\ \mathrm{c}\psi_c^o\ -\ v\ \mathrm{s}\psi_c^0 \\ u_{y_c}^o &=& u_c^o\ \mathrm{s}\psi_c^o\ +\ v\ \mathrm{c}\psi_c^0, \end{array}\right\} \quad (12)$$

and command filtered variables:

$$\left.\begin{array}{rcl} u_{x_c} &=& u_c\ c\psi_c\ -\ v\ s\psi_c \\ u_{y_c} &=& u_c\ s\psi_c\ +\ v\ c\psi_c. \end{array}\right\} \quad (13)$$

Note that v in eqn. (12) and (13) is not a commanded value, but is included into the command signal for compensation of disturbances acting lateral to vehicle movement.

Eqn. (12) can be inverted to give the desired physical values for u_c^o and ψ_c^o

$$\left.\begin{array}{rcl} u_c^o &=& u_{x_c}^o\ c\psi_c^o + u_{y_c}^o\ s\psi_c^o \\ \psi_c^o &=& atan2\ (-u_{x_c}^o,\ u_{y_c}^o)\ + \beta, \end{array}\right\} \quad (14)$$

where $u_{x_c}^o$, $u_{y_c}^o$ are known and β is defined as:

$$\beta = \arccos \frac{v}{\left\| \begin{bmatrix} u_{x_c}^o \\ u_{y_c}^o \end{bmatrix} \right\|} \quad (15)$$

The error signals

$$\left.\begin{array}{rcl} \tilde{u}_x &=& u_x - u_{x_c} \\ \tilde{u}_y &=& u_y - u_{y_c}, \end{array}\right\} \quad (16)$$

will be important in the subsequent analysis.

B. Control Design and Error Analysis

The dynamic equation for x and y can be written as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} u_{x_c}^o \\ u_{y_c}^o \end{bmatrix} + \begin{bmatrix} \tilde{u}_x \\ \tilde{u}_y \end{bmatrix} + \begin{bmatrix} u_{x_c} - u_{x_c}^o \\ u_{y_c} - u_{y_c}^o \end{bmatrix} \quad (17)$$

For the purpose of the control signal design, we select the signals $[u_{x_c}^o, u_{y_c}^o]^\top$ as

$$\begin{bmatrix} u_{x_c}^o \\ u_{y_c}^o \end{bmatrix} = \begin{bmatrix} -K_{xy}\tilde{x} + \dot{x}_c \\ -K_{xy}\tilde{y} + \dot{y}_c \end{bmatrix} \quad (18)$$

where $K_{xy} > 0$ can be time varying. The selection of K_{xy} is discussed in Appendix II. With the control signal in eqn. (18) the x and y position error dynamics are

$$\begin{bmatrix} \dot{\tilde{x}} \\ \dot{\tilde{y}} \end{bmatrix} = \begin{bmatrix} -K_{xy}\tilde{x} \\ -K_{xy}\tilde{y} \end{bmatrix} + \begin{bmatrix} \tilde{u}_x \\ \tilde{u}_y \end{bmatrix} + \begin{bmatrix} u_{x_c} - u_{x_c}^o \\ u_{y_c} - u_{y_c}^o \end{bmatrix}. \quad (19)$$

The $\tilde{u}_x$ and $\tilde{u}_y$ terms can be manipulated by two very similar approaches (see Appendix 2 in [7]). In either case this term can be expressed in the form

$$\begin{bmatrix} \tilde{u}_x \\ \tilde{u}_y \end{bmatrix} = \begin{bmatrix} A & Bg(\tilde{\psi}) \end{bmatrix} \begin{bmatrix} \tilde{u} \\ \tilde{\psi} \end{bmatrix} \quad (20)$$

Thus, the position error dynamics can be expressed as

$$\begin{bmatrix} \dot{\tilde{x}} \\ \dot{\tilde{y}} \end{bmatrix} = \begin{bmatrix} -K_{xy}\tilde{x} \\ -K_{xy}\tilde{y} \end{bmatrix} + \begin{bmatrix} A & Bg(\tilde{\psi}) \end{bmatrix} \begin{bmatrix} \tilde{u} \\ \tilde{\psi} \end{bmatrix} + \begin{bmatrix} u_{x_c} - u_{x_c}^o \\ u_{y_c} - u_{y_c}^o \end{bmatrix}$$

which is a form suitable for stability analysis.

Define the compensated tracking error signals ν_x and ν_y as

$$\begin{bmatrix} \nu_x \\ \nu_y \end{bmatrix} = \begin{bmatrix} \tilde{x} - \xi_x \\ \tilde{y} - \xi_y \end{bmatrix}. \quad (21)$$

where ξ_x and ξ_y are defined as

$$\begin{bmatrix} \dot{\xi}_x \\ \dot{\xi}_y \end{bmatrix} = \begin{bmatrix} -K_{xy}\xi_x \\ -K_{xy}\xi_y \end{bmatrix} + \begin{bmatrix} A & Bg(\tilde{\psi}) \end{bmatrix} \begin{bmatrix} \xi_u \\ \xi_\psi \end{bmatrix} + \begin{bmatrix} u_{x_c} - u_{x_c}^o \\ u_{y_c} - u_{y_c}^o \end{bmatrix}. \quad (22)$$

with $\xi_x(0) = 0$ and $\xi_y(0) = 0$. With these definitions, the dynamics of the compensated tracking errors are

$$\begin{bmatrix} \dot{\nu}_x \\ \dot{\nu}_y \end{bmatrix} = \begin{bmatrix} -K_{xy}\nu_x \\ -K_{xy}\nu_y \end{bmatrix} + \begin{bmatrix} A & Bg(\tilde{\psi}) \end{bmatrix} \begin{bmatrix} \nu_u \\ \nu_\psi \end{bmatrix} \quad (23)$$

where ν_u and ν_ψ are defined in eqns. (25) and (29).

V. Yaw Control

The objective of this section is to design a controller to stabilize the dynamic system of eqns. (1) through (3). Because the position dynamics were already discussed, this section focuses on selection of r_c^o to stabilize the ψ dynamics. For ψ tracking control, the input is the yaw command $\psi_c(t)$ and its derivative $\dot{\psi}_c(t)$ which are produced by a command filter with input ψ_c^o as discussed in Appendix I.

For yaw control, based on eqn. (3), we define the signal

$$r_c^o = -K_\psi\tilde{\psi} + \dot{\psi}_c - \psi_{bs}$$

where K_ψ is a positive constant. Using this definition, the closed-loop tracking error corresponding to eqn. (3) is

$$\begin{array}{rcl} \dot{\psi} &=& r_c^o + (r - r_c) + (r_c - r_c^o) \\ &=& -K_\psi\tilde{\psi} + \dot{\psi}_c - \psi_{bs} + \tilde{r} + (r_c - r_c^o) \\ \dot{\tilde{\psi}} &=& -K_\psi\tilde{\psi} - \psi_{bs} + \tilde{r} + (r_c - r_c^o). \end{array} \quad (24)$$

The compensated tracking error signal for the ψ dynamics is defined as

$$\nu_\psi = \tilde{\psi} - \xi_\psi. \quad (25)$$

The signal ξ_ψ is defined as

$$\dot{\xi}_\psi = -K_\psi\xi_\psi + (r_c - r_c^o) + \xi_r \quad (26)$$

with $\xi_\psi(0) = 0$. With these definitions, the dynamic equation of ν_ψ is

$$\begin{array}{rcl} \dot{\nu}_\psi &=& \dot{\tilde{\psi}} - \dot{\xi}_\psi \\ &=& -K_\psi\nu_\psi - \psi_{bs} + \nu_r, \end{array} \quad (27)$$

where ν_r is defined in (30).

VI. Velocity and Yaw Rate Control

The objective of this section is to design a controller to stabilize the dynamic system of eqns. (1–5) using backstepping. This section focuses on selection of F and τ to stabilize the u and r tracking error dynamics. For u and r tracking control, the inputs are the horizontal speed and yaw rate commands $u_c(t)$ and $r_c(t)$ and their derivatives $\dot{u}_c(t)$ and $\dot{r}_c(t)$ which are produced by command filters with input u_c^o and r_c^o as discussed in Appendix I.

For tracking control using eqns. (4-5) we select the control force and the control torque as

$$F = -g(u,r) - K_u\tilde{u} + \dot{u}_c - u_{bs}$$
$$\tau = -f(u,r) - K_r\tilde{r} + \dot{r}_c - r_{bs}, \quad (28)$$

where K_u and K_r are positive constants.

With this choice of the control signal the dynamics of the u and r tracking errors are

$$\begin{aligned} \dot{\tilde{u}} &= -K_u\tilde{u} - u_{bs} \\ \dot{\tilde{r}} &= -K_r\tilde{r} - r_{bs} \end{aligned}$$

The signals ξ_u and ξ_r are identically zero; therefore,

$$\nu_u = \tilde{u} \quad (29)$$

$$\nu_r = \tilde{r} \quad (30)$$

and the dynamics of the compensated tracking errors are

$$\left.\begin{aligned} \dot{\nu}_u &= -K_u\nu_u - u_{bs} \\ \dot{\nu}_r &= -K_r\nu_r - r_{bs}. \end{aligned}\right\} \quad (31)$$

VII. Backstepping Terms

The backstepping terms that are inputs to our control law are defined as

$$\psi_{bs} = g(\tilde{\psi})^\top B^T \begin{bmatrix} \nu_x \\ \nu_y \end{bmatrix} \quad (32)$$

$$u_{bs} = A^T \begin{bmatrix} \nu_x \\ \nu_y \end{bmatrix} \quad (33)$$

$$r_{bs} = \nu_\psi. \quad (34)$$

The stability proof is beyond the scope of this conference paper since a similar proof is shown for a land robot in [7] and the reader can refer to this paper for the stability analysis.

VIII. Unmanned Surface Vehicle (USV)

NURC's Mandarina USV [10] was designed with the maximal use of COTS hardware in mind. The prototype vehicle, a 4.6-meter rubber boat, was build around the SPECTRE autopilot made by H-Scientific Ltd for the capability of countering underwater intruders.

Mandarina is equipped with forward and side-looking multibeam sonar for target following and close observation. Navigation sensors onboard include a motion reference unit (MRU) and GPS.

MOOS-IvP infrastructure uses serial communication to transmit guidance information to SPECTRE.

IX. MOOS-IvP Implementation

The similar controller to the one described in Section III was previous applied to the second order system [5] and to the land robot [3]. In this paper we apply the same control design to the USV described in VIII. Controller implementation is done via MOOS-IvP.

Frontseat/Backseat control infrastructure is shown in Figure 1. Guidance and sonar processing is performed on the backseat computer. Access to actuators from the backseat is not allowed. Course, course rate and speed control will

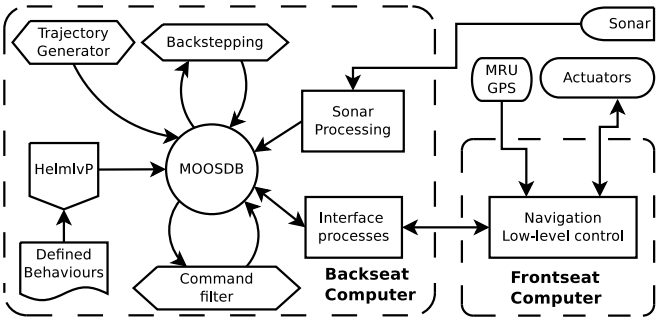

Fig. 1. Block diagram of the control system implemented on the Mandarina USV.

be performed by the frontseat. Due to this limitation, only equations (6) through (8) are implemented on the backseat controller.

MOOS processes communicate through a MOOS database in a publish-subscribe manner. Variables of interest are published to this database by processes, while others subscribe to variables they need. Modular design is achieved this way. For example, backstepping process uses information about X and Y vehicle position. X and Y is supplied by a different module. We could easily switch between modules that estimates X and Y from navigation data without resetting the backstepping process.

Trajectory generator in Figure 1 generates a circular trajectory around a target point. Idea behind the circular trajectory is keeping the target in the side-looking sonar's field of view. This way the target can be mapped from all sides enabling easier identification. The target point is determined by sonar processing. An application has been developed (by Alberto Grati, NURC) that utilizes information from one or more BlueView P450 multi-beam sonars. Idea behind this is to detect a potential target through sonar image processing. Currently an operator identifies the target from a sonar image and clicks on it to feed-back the target position. It is obvious that with addition of sonar image processing this could be a fully autonomous system as described above.

Speed, course or course rate are calculated by the CFBS. Implementation of the controller is separated in two processes as can be seen on Figure 1. Calculated values are then used by the HelmIvP process, [1]. HelmIvP uses Interval Programming to determine the best point inside the guidance domain (speed/course or speed/course rate). HelmIvP publishes this desired guidance set points into MOOSDB from where they are sent to the frontseat autopilot. Reason why we use HelmIvP, instead of directly sending set points to the frontseat, is because we have multiple vehicle behaviours that we want to run concurrently.

Parallel to trajectory following behaviour we have an obstacle avoidance behaviour. Suppose we have an obstacle on our circular path. Without an obstacle avoidance behaviour we would need to adjust our trajectory generation to avoid the obstacle. With an obstacle behaviour we need not worry about adjusting the trajectory. HelmIvP will, based on priorities and objective functions calculated by each behaviour,

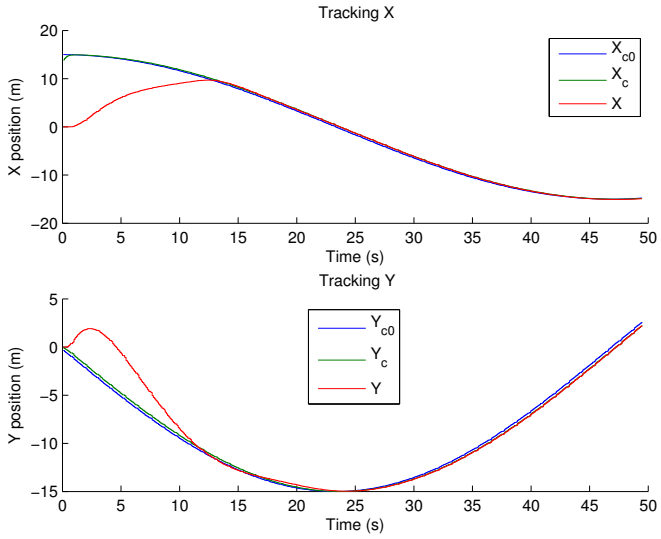

Fig. 2. X and Y position vs. Time: Red line is the actual vehicle trajectory, blue line is the command, and the green line is filtered command.

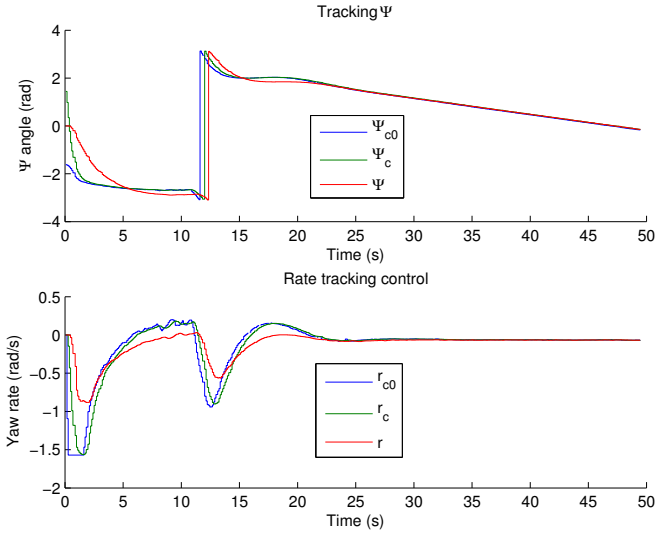

Fig. 3. Yaw and Yaw rate vs. Time: Red line is the actual vehicle state, blue line is the command, and the green line is filtered command.

choose valid set points to avoid the target and, when the danger of collision passes, get back on the desired trajectory.

In section X we show simulation results of our trajectory tracking behaviour.

X. Simulation Results

Simulations were performed in a MOOS framework as well. We used MOOS for simulation, instead of Matlab, because the controller is implemented in this framework. This way we can get better insight how the controller will function in practice.

Mission objective was to circle around the origin of the local coordinate system with a cruising speed of 1 m/s, at a radius of 15 meters. Current of magnitude 0.6 m/s in a 45° direction was added as a disturbance.

Parameter K_{xy} is changed adaptively as described in Appendix II. The control law parameters were as follows:$\omega_n = 5\frac{rad}{s}$, $\zeta = 0.9$, $\bar{k} = 0.5$, $\alpha = 0.75$, $K_\psi = 0.75$. K_u and K_r are not set since only r_c^o and u_c^o are valid inputs for the SPECTRE autopilot. IvPHelm and the CFBS controller are running with sampling frequency $10\ Hz$.

Result are shown in figures 2 through 6. Figure 2 shows the position converge to the desired trajectory. When the vehicle converges to the trajectory is keeps tracking the command filtered trajectory. Static error in tracking the desired trajectory can be made arbitrarily small as mentioned in section III.

On figure 3 yaw angle behaviour and tracking is shown. We observe that yaw angle converges to the desired value. Yaw angle convergence is achieved through yaw rate control. Yaw rate is shown in figure 3. We observe that signals converge after the vehicle has stabilized on the desired trajectory. We observe that the vehicle has a hard time following these signals. This problem occurs because we do not implement the whole CFBS controller.

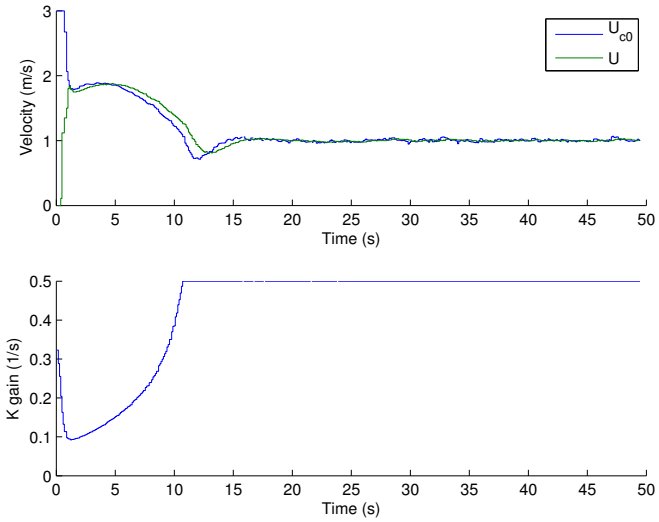

Fig. 4. Speed and variable gain K_{xy} error vs. Time: Green line is the estimated vehicle speed, and the blue line is speed command. K_{xy} is displayed on the bottom picture.

Position gain K_{xy} is recalculated in each step to limit maximum speed command signal and avoid unnecessary turning in case of trajectory lead point overshoot. Gain change versus time is displayed in figure 4. Vehicle starting point was chosen far from the starting point of the trajectory to demonstrate benefits of adaptive K_{xy} gain. We limited the maximum speed value by choosing $\bar{u}$, defined in Appendix II, appropriately. K_{xy} is changing while the vehicle is far from the trajectory lead point. Large position error would result in a large speed command, therefore K_{xy} is decreased to compensate for this and ensure that the command signal is feasible. Once position errors become smaller K_{xy} gain becomes constant.

Under normal operating conditions in land vehicles, lateral forces are small or non-existent. Therefore, they do not represent a problem. However, in surface crafts forces

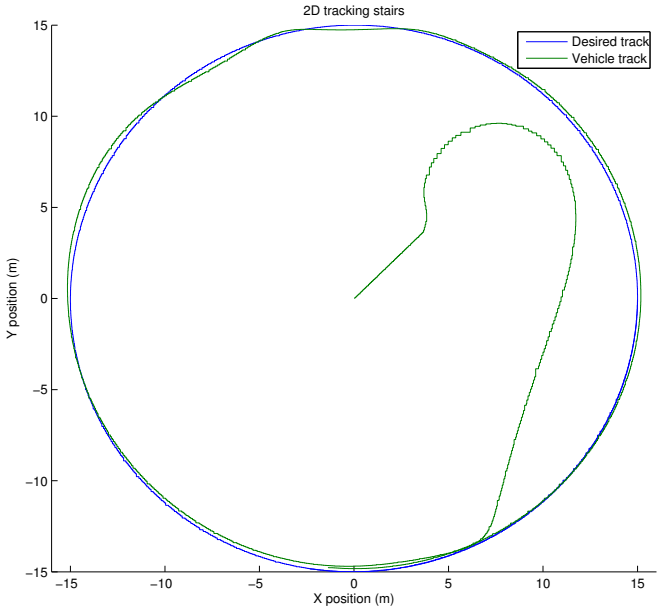

Fig. 5. Trajectory of the vehicle on the x-y plane. The actual vehicle trajectory (green) starts at the position it was translated in by the currents while the controller was offline. The commanded trajectory (blue) is a 15.0 m clockwise circle starting at $(15, 0)$.

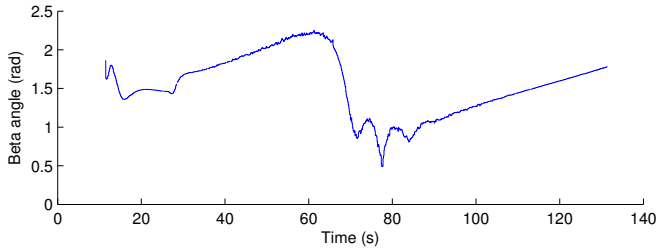

Fig. 6. Beta angle. Angle is changing to direct the vehicle into the current so that lateral movement can be compensated.

caused by currents and wave motions can have a great impact on vehicle movement. Partly compensating for this disturbances can yield better and more accurate control over the vehicle. Lateral disturbances acting on the vehicle have been integrated into the CFBS controller via the term v. With estimation of this value we can compensate for lateral disturbances. In figures 5 and 6 we display simulation results with this compensation enabled.

Figure 5 show the vehicle converging toward the desired trajectory. Compensation of lateral forces is achieved by manipulating vehicle yaw and steering the vehicle to counter currents. Forces acting perpendicular to movement are canceled out. Vehicle is effectively crabbing along the track. Yaw angle manipulation is achieved by calculating the factor β in eqn. (7). Factor β versus time is displayed in figure 6.

XI. Conclusion

This article has discussed the design and derivation of a CFBS approach to design a stable translational controller (i.e., $z(t) = [x(t), y(t)]^\top$) applicable to a USV. The mission scenario specifies the position and desired speed commands which are command filtered to produce inputs (together with their derivatives) for the translational and yaw controllers. The commands (u_c) and (r_c), generated by the translational and yaw loop, are inputs to the velocity and yaw rate controllers. The u and r controller determines the appropriates actuator force/torque. Simulations have shown that it is feasible to implement CFBS controller in the MOOS-IvP structure. MOOS-IvP is becoming popular in surface and underwater vehicle control. Authors hope that in time a complete CFBS controller implementation will be possible on the real vehicle. Several plans had been laid for the future. Implementing the whole CFBS controller is our first priority. Testing the range of target detection with a vehicle mounted sonar is planned for October, 2010. CFBS controller described here will be used to perform these testing.

Appendix I
Command Filter

The purpose of this appendix is to provide an example and discussion of a command filter. Advanced control approaches often assume the availability of a continuous and bounded desired trajectory $y_c(t)$ and its first r derivatives $y_c^{(r)}(t)$. The first time that this assumption is encountered it may seen unreasonable, since a user will often only specify a command signal $y_d(t)$. However, this assumption can always be satisfied by passing the commanded signal $y_d(t)$ through a single-input, multi-output prefilter. This procedure is explained in detail in for example [6].

Throughout, this article refers to filtering of a signal x_c^o to produce a bandwidth limited signal x_c and its derivative $\dot{x}_c$. The state space implementation of such a filter is

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -2\zeta w_n x_2 - w_n^2 (x_1 - x_c^o) \end{aligned}$$

where $x_c = x_1$ and $\dot{x}_c = x_2$. Note that if x_c^o is bounded, then x_c and $\dot{x}_c$ are bounded and continuous. The transfer function from x_c^o to x_c is

$$\frac{X_c(s)}{X_c^o(s)} = H(s) = \frac{w_n^2}{s^2 + 2\zeta w_n s + w_n^2} \tag{35}$$

which has a unity gain at low frequencies, damping ratio ζ and undamped natural frequency ω_n. The error $|x_c^o(t) - x_c(t)|$ is small if the bandwidth of $x_c^o(s)$ is less than the bandwidth of $H(s)$. If the bandwidth of x_c^o is known and the goal of the filter is to generate x_c and its derivative with $|x_c^o - x_c|$ small, then the designer simply chooses ω_n sufficiently large.

Note that the signal $\dot{x}_c$ is computed by integration, not differentiation. This helps to decrease the effects of measurement noise; nonetheless, noise will impose a tradeoff in how large of a value can be selected for ω_n.

Appendix II
Selection of Control Gain

Eqn. (18) has the form,

$$\begin{bmatrix} u_{x_c}^o \\ u_{y_c}^o \end{bmatrix} = v_d - K_{xy} E$$

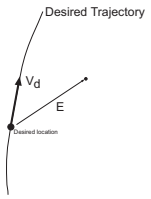

Fig. 7. Trajectory depiction

where

$$v_d = \begin{bmatrix} \dot{x}_c \\ \dot{y}_c \end{bmatrix} \text{ and } E = \begin{bmatrix} \tilde{x} \\ \tilde{y} \end{bmatrix}$$

The quantity v_d is the velocity vector that causes the vehicle to follow the trajectory given that the vehicle was currently on the trajectory. The quantity $K_{xy}E$ is the feedback term that causes the vehicle to converge toward the trajectory.

The definition of ($u^o_{x_c}$,$u^o_{y_c}$) from eqn. (18) defines (u^o_c,ψ^o_c) as in eqn. (14). This subsection discusses technical details related to the selection of K_{xy} in eqn. (18). The gain K_{xy} is changed as a function of the control state information to achieve two objectives. First, the vehicle yaw should never differ from the trajectory heading by more than 90 degrees. Second, the norm of the the term $K_{xy}E$ should always be less than a parameter $\bar{u} > 0$. The purpose of the second constraint is to ensure that poor initial conditions do not yield unreasonably large speed commands to the vehicle.From definition of $K_{xy}(t)$ we can see that when $\|E\|$ is large enough K_{xy} is proportional to $\bar{u}$. Based on this relation we can see that $\bar{u}$ can limit position error involvement in generation of speed command signals, limiting speed command values.

For the stability analysis, the value of K_{xy} must be positive; however, its magnitude can be manipulated to achieve secondary performance objectives such as those stated above. Consider the situation depicted in Figure 7 where the inner product of v_d and E is positive (i.e., the robot is ahead of the current desired trajectory position). Depending on the value of K_{xy}, the commanded yaw angle could result in the vehicle circling to get to the desired location. In particular, when $K_{xy}\|E\| > \|v_d\|$, the vehicle yaw may be commanded in a direction opposite to the direction of v_d. Typically, this is not desirable. To prevent this we must ensure that the angle between v_d and $v_d - K_{xy}E$ is less than $90\,\deg$:

$$\langle v_d, (v_d - K_{xy}E)\rangle \geq 0 \quad (36)$$

$$\|v_d\|^2 \geq K_{xy}\langle v_d, E\rangle. \quad (37)$$

There are three possible cases:

1) $\langle v_d, E\rangle > 0$: This is the problematic case that could result in the vehicle pointing opposite to the desired velocity if K_{xy} is too big. The value of K_{xy} should be selected such that

$$K_{xy} \leq \frac{\|v_d\|^2}{\langle v_d, E\rangle}.$$

2) $\langle v_d, E\rangle = 0$: In this case, the value of K_{xy} does not matter.
3) $\langle v_d, E\rangle < 0$: In this case, any positive value of K_{xy} satisfies eqn. (37).

Therefore, the designer specifies positive constants $\bar{k}$ and $0 < \alpha < 1$. The parameter $\bar{k}$ is the largest allowable position control feedback gain. At each time instant, the value of K_{xy} is selected as

$$K_{xy}(t) = \begin{cases} \min\left(\bar{k}, \frac{\bar{u}}{\|E\|}\right) & \text{if} \quad \langle v_d, E\rangle \leq 0 \\ \min\left(\frac{\alpha\|v_d\|^2}{\langle v_d, E\rangle}, \bar{k}, \frac{\bar{u}}{\|E\|}\right) & \text{if} \quad \langle v_d, E\rangle > 0. \end{cases}$$

This definition of $K_{xy}(t)$ is a positive, continuous function of time. In situations such as that in Figure 7, this approach results in the vehicle driving towards the trajectory with the tangential component small enough that the trajectory point will ultimately catch up to the vehicle. In the case where the vehicle is on the trajectory directly in front of the desired location, this approach causes the vehicle to drive slower than the desired point is moving, in effect waiting for the desired position to catch up.

ACKNOWLEDGMENT

The authors gratefully acknowledge the NATO Undersea Research Centre (NURC) for funding Mr. Dula Nad under the Visiting Researcher Programme (VRP). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NURC.

REFERENCES

[1] M. R. Benjamin, H. Schmidt, and J. Leonard. *A Guide to the IvP Helm for Autonomous Vehicle Control*, December 2007.
[2] M. Breivik, V.E. Hovstein, and T.I. Fossen. Straight-line target tracking for unmanned surface vehicles. *Modeling, Identification and Control*, 2008.
[3] V. Djapic, J. A. Farrell, and W. Dong. Hybrid control design applied in land vehicle behavior based switching controller. In *Proceedings of the 2008 IEEE Multi-conference on Systems and Control*, 2008.
[4] V. Djapic, J. A. Farrell, and W. Dong. Land vehicle control using command filtered backstepping approach. In *Proceedings of the American Control Conference*, 2008.
[5] V. Djapic, J. A. Farrell, and W. Dong. Unifying behavior-based control design and hybrid stability theory. In *Proceedings of the 2008 IEEE Multi-conference on Systems and Control*, 2008.
[6] J. A. Farrell, M. Polycarpou, M. Sharma, and W. Dong. Command filtered backstepping. *IEEE Transactions on Automatic Control*, 2007.
[7] J. A. Farrell, M. Polycarpou, M. Sharma, and W. Dong. Command filtered backstepping. In *Proceeding of the IEEE American Control Conference*, 2008.
[8] H. K. Khalil. *Nonlinear Systems*. Prentice-Hall, New Jersey, 3 edition, 2002.
[9] Paul Newman. *Introduction to Programming with MOOS*, December 2007.
[10] T.J. Pastore. An unmanned surface vessel for subsurface threats in harbours. In *Proceedings of the AUVSI's Unmanned Systems Europe 2009*, 2009.
[11] D. Soetanto, L. Lapierre, and A. Pascoal. Adaptive, non-singular path-following control of dynamic wheeled robots. In *Proceedings of the 42th IEEE Conference on Decision and Control*, 2003.