2020 American Control Conference
Denver, CO, USA, July 1-3, 2020

Backstepping Control of Gliding Robotic Fish for Trajectory Tracking in 3D Space

Demetris Coleman and Xiaobo Tan

Abstract— **Autonomous underwater gliders have become valuable, energy-efficient tools for a myriad of applications including ocean exploration, fish tracking, and environmental sampling. Many applications, such as, exploring a large area of underwater ruins or navigating through a coral reef, would benefit from fine trajectory tracking. However, trajectory tracking control of underwater gliders is particularly challenging due to their under-actuated, nonlinear dynamics. Taking gliding robotic fish as an example, in this work we propose a backstepping-based controller for the gliding motion to track a desired reference for the pitch angle and position in the 3D space. In particular, the challenge of under-actuation is addressed by exploiting the coupled dynamics and introducing a new modified error term that combines pitch and horizontal position tracking errors. The effectiveness of the proposed control scheme is demonstrated via simulation and its advantages are shown via comparison with a PID controller.**

I. Introduction

Underwater gliders are autonomous vehichles that use variable buoyancy and hydrofoils to realize horizontal travel by shifting the center of gravity. They are known for high energy efficiency, allowing them to have exceptionally long operational periods. Their typical motion patterns are sawtooth-like gliding. Another useful motion is spiraling, enabled by controlling the vehicle's roll angle or deflecting the control surfaces. The concept, introduced by Henrey Stommel [1], motivated development of gliders such as SLOCUM [1], Spray [2], and Seaglider [3]. The success of the early gliders has inspired development of other underwater vehicles that exploit gliding [4], [5] like the gliding robotic fish [5], which combines the gliding mechanism with the tail-actuated swimming [6] to realize both high energy-efficiency and high maneuverability. It has demonstrated promise in environmental sensing applications [7].

Control of gliders presents a significant challenge, due to their highly nonlinear and under-actuated dynamics. Early work in control of gliders saw the use of PID controllers for their simplicity [2], [3]. In the past two decades, more advanced and model-based control methodologies have been proposed. For example, neural network-based control was used to implement a self-tuning PID controller to track the velocity along a single axis in the inertial frame [8]. Isa and Arshad analyzed the use of a neural network as a model predictive controller and a gain-tuner algorithm to the control pitch angle and linear velocities of a linearized glider model [9]. Nag et al. [10] compared fuzzy logic control against PID for pitch and depth tracking. Zhang et al. [11] used nonlinear passivity-based control to stabilize the glide path of a glider in the sagittal plane with a whale-like tail. Sliding mode control has also been explored because of its robustness to disturbances. Yang and Ma used sliding mode control to track trajectories of the pitch angle and ballast mass [12]. Mat-Noh et al. used a linearized glider model to compare an Integral Super Twisting Sliding Mode controller with several other sliding mode variants for stabilizing a gliding path between 30 and 45 degrees [13]. Leonard and Graver used a linear quadratic regulator on linearized dynamics to control the magnitude of velocity on a steady-state glide path [14], [15]. Mahmoudian and Woosely developed an efficient path planning strategy that concatenates equilibrium turning and gliding motions, and then implemented the strategy using PID controllers to reach specified center of gravity and center of buoyancy [16]. In [17], several different control strategies for underwater gliders are compared.

Despite the aforementioned extensive work on glider control, current approaches have largely focused on stabilization based on linearized models, or single-input single-output control of heading, pitch or velocity control. In fact, most work on underwater glider control focuses on controllers designed to reach a desired pitch angle, velocity, or specified depth [18]. In particular, the study of trajectory tracking control in the 3D space is scarce, if any. Trajectory tracking for underwater gliders and other gliding-type vehicles is a basic functionality that is of direct relevance to various sampling and target-tracking applications for the underwater environment. It is a valuable ability to have for exploring complex environments, such as coral reefs and underwater ruins. In addition, it also enables improved performance in many other applications in oceanography, marine science, water quality monitoring, and surveillance. Trajectory tracking and path following have been heavily studied for propeller-driven underwater vehicles [19]–[26]. However, the literature on these topics for underwater gliders is very limited, with [18] being one of very few examples considering the full dynamic model of a gliding system. The authors of [18] proposed an adaptive backstepping control for tracking the yaw angle, the pitch angle, and the velocity magnitude of an underwater glider, but not position trajectories. However, trajectory tracking for positions, as in the works cited above on propeller-driven underwater vehicles, is arguably more valuable when operating in cluttered underwater environments.

In this paper, a backstepping-based trajectory tracking

*This research was supported in part by by National Science Foundation (IIS 1715714, IIS 1848945) and the Rose Graduate Fellowship Fund in Water Research.

Demetris Coleman and Xiaobo Tan are with the Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI 48824, USA. Email: colem404@msu.edu (D.C.), xbtan@egr.msu.edu (X.T)

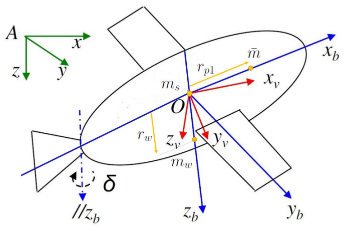

Fig. 1. Illustration of robot reference frames and mass distribution [29].

controller is proposed to control the 3D motion of underwater gliders, tracking reference trajectories for both the 3D position and the pitch angle. Backstepping-based control design presents a practical and promising approach because it offers a systematic framework that guarantees the stability of the closed-loop system, and allows the accommodation of input constraints [27]. To put the discussion in context, we focus on the model of a gliding robotic fish. With only three control inputs, the proposed control scheme addresses simultaneous tracking of pitch and 3D position. To facilitate the control design, the tracking errors are expressed in a cylindrical coordinate system with its origin coinciding with that of the robot's body-fixed frame. To address the under-actuated problem, a hybrid error function is introduced by modifying the pitch tracking error with a term regulated by the position tracking error in the horizontal plane. Lyapunov analysis, which drives the backstepping control design, shows that the depth tracking error, the aforementioned error, and the error between the yaw angle and the direction of x-y planar position tracking error, all converge to zero. We further sketch the procedure, using two-time-scale analysis, for establishing the tracking of all four components of the reference trajectories (pitch and 3D position).

This work represents a significant extension to our prior work [28], which deals with trajectory tracking of pitch and 2D position in the saggital plane. Aside from extension from 2D to 3D, a critical improvement over [28] is the elucidation of why the proposed controller is able to achieve tracking with under-actuation. Extensive simulation results are presented to show that the proposed controller is able to track different trajectories in the 3D space and it outperforms a set of well-tuned PID controllers in these tracking tasks. Similar to our previous work [28], a sliding mode observer is implemented to estimate the body-fixed velocities, which are otherwise not directly accessible, but the design is excluded due to space limitation.

The rest of this paper is organized as follows. Section II describes the system model and the problem formulation. Section III provides an overview of the control design process and analysis of the closed-loop system, followed by simulation results in Section IV and the conclusion in Section-V.

II. System Modeling and Problem Formulation

A. Gliding Robotic Fish Model

The robot, depicted in Fig. 1, has two relevant reference frames. The first is the inertial frame, represented by A_{xyz}. The A_z axis is along the direction of gravity, and A_x/A_y are defined in the horizontal plane, with the origin A as a fixed point in space. The body-fixed frame is denoted by $O_{x_b y_b z_b}$ with the origin O at the geometric center of the glider body. The O_{x_b} axis is along the body longitudinal axis pointing toward the robot's front, the O_{z_b} axis is perpendicular to the O_{x_b} axis in the sagittal plane of the robot pointing towards the robot's underbelly, and the O_{y_b} axis is formed according to the right-hand orthonormal principle. The glider is modeled as a 6 degree-of-freedom (DOF) rigid body with an internal moving mass, a water tank, and a servo-actuated tail that has its own axis of rotation parallel to the robot's O_{z_b} axis, at an offset along the O_{x_b} axis. The tail can be used for both propulsion and steering, and in the context of this paper, it is used for the latter only. The internal movable mass, which is restricted to the longitudinal axis, controls the robot's pitch angle. The last control input, representing the negative net buoyancy, is given as the sum of the uniformly distributed stationary mass m_s (including mass of water in tank), internal movable mass $\bar{m}$, and non-uniformly distributed mass m_w minus the mass m of the water displaced by the robot. This can be expressed as $m_0 = m_s + \bar{m} + m_w - m$, where $m_0 < 0$ causes the robot to float and $m_0 > 0$ causes the robot to sink. Effectively, the robot controls m_0 by changing the amount of water in the tank. In summary, the control inputs include the negative net buoyancy m_0, the distance r_{p1} of the movable mass from the body-frame origin, and the tail angle δ.

The state vector, consisting of the position $b_i = [x, y, z]^T$ of the robot, the Euler angles (roll, pitch, and yaw) $\Psi = [\phi, \theta, \psi]^T$ given in the inertial frame, and the body-fixed linear velocities $v_b = [v_1, v_2, v_3]^T$ and angular velocities $\omega_b = [\omega_1, \omega_2, \omega_3]^T$, is given by

$$X = [x, y, z, \phi, \theta, \psi, v_1, v_2, v_3, \omega_1, \omega_2, \omega_3]^T \quad (1)$$

The dynamic equations are

$$\begin{cases} \dot{b}_i = R v_b \\ \dot{\Psi} = R_\omega \omega_b \\ \dot{v}_b = M^{-1}(M v_b \times \omega_b + m_0 g R^T k + F_{ext}) \\ \dot{\omega}_b = J^{-1}(-J\omega_b + J\omega_b \times \omega_b + M v_b \times v_b + T_{ext} \\ \quad + m_w g r_w \times (R^T k) + \bar{m} g r_p \times (R^T k)) \end{cases} \quad (2)$$

with

$$R_\omega = \begin{pmatrix} 1 & \tan(\theta)\sin(\phi) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \frac{\sin(\phi)}{\cos(\theta)} & \frac{\cos(\phi)}{\cos(\theta)} \end{pmatrix}$$

where $r_p = [0, 0, r_{p1}]^T$, $M = \text{diag}\{m_1, m_2, m_3\}$ is the added mass matrix due to the surrounding fluid, $J = \text{diag}\{J_1, J_2, J_3\}$ is the added inertia matrix, $T_{ext} = R_{bv}\,[M_1, M_2, M_3]^T$ is the hydrodynamic moment vector,

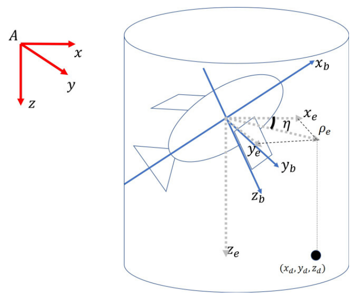

Fig. 2. Illustration of the robot error frame. A is the inertial frame and the point (x_d, y_d, z_d) is the desired position for the robot. The position error vector (x_e, y_e, z_e) is the difference between the desired position and the center of the robot. The axes x_b, y_b, and z_b represent the body-fixed coordinate frame.

$F_{ext} = R_{bv}\,[-D, F_s, -L]^T$ is the hydrodynamic force vector, $k = [0, 0, 1]^T$, and $r_w = [0, 0, r_{w3}]^T$. R is a 3×3 rotation matrix parameterized by the Euler angles $\Psi = [\phi, \theta, \psi]^T$ and R_{bv} is a 3×3 rotation matrix parameterized by α and β. These matrices, the hydrodynamic forces of lift L, drag D, side force F_s, and the yaw moment M_3, pitch moment M_2, and roll moment M_1 are as given in [5]. Furthermore, note that $\alpha = \arctan\frac{v_3}{v_1}$ is the angle of attack and $\beta = \arcsin\frac{v_2}{\sqrt{v_1^2+v_2^2+v_3^2}}$ is the side-slip angle.

For convenience, we will abstract the linear and angular velocity dynamics as

$$\begin{bmatrix} \dot{v_1} \\ \dot{v_2} \\ \dot{v_3} \\ \dot{\omega_1} \\ \dot{\omega_2} \\ \dot{\omega_3} \end{bmatrix} = \begin{bmatrix} f_{v11} + a_{v1}\sin(\theta)u_1 + f_{v12}u_3 + f_{v13}u_3^2 \\ f_{v21} + f_{v21}u_3 + f_{v22}u_3^2 \\ f_{v31} + a_{v3}\cos(\theta)u_1 + f_{v32}u_3 + f_{v33}u_3^2 \\ f_{\omega 11} + f_{\omega 12}u_3 \\ f_{\omega 21} + a_{\omega_2}\cos(\theta)u_2 \\ f_{\omega 31} + f_{\omega 32}u_3 \end{bmatrix} \tag{3}$$

where $u_1 = m_0$, $u_2 = r_{p1}$, and $u_3 = \delta$ are the controls, a_{v1}, a_{v3} and a_{ω_2} are constants, and f_{vij} and $f_{\omega ij}$ are nonlinear functions of the state vector.

B. Problem Formulation

The problem of trajectory tracking involves controlling a robot to follow a time-dependent path. In our work, we strive to have the robot pose $P = [x, y, z, \theta]^T$ follow a trajectory in the inertial coordinate system, given by the desired path $P_d(t) = [x_d(t), y_d(t), z_d(t), \theta_d(t)]^T$. $\dot{P}_d(t)$ and $\ddot{P}_d(t)$ are assumed to be sufficiently smooth with $|\theta_d| < \pm\frac{\pi}{2}$. To solve this problem, we define the inertial frame error $P_e(t) = [x_e, y_e, z_e, \theta_e]^T$ as

$$P_e(t) = \begin{bmatrix} x - x_d \\ y - y_d \\ z - z_d \\ \theta - \theta_d \end{bmatrix} \tag{4}$$

According to Fig. 2, the Cartesian error coordinates can be rewritten in cylindrical coordinates, which is more control-friendly for the system. This can be done by representing the position error vector (x_e, y_e) in the plane as a magnitude $\rho_e = \sqrt{x_e^2 + y_e^2}$ and angle $\eta = \arctan(y_e/x_e)$ suitably defined to give the correct quadrant. The vector is expressed in the inertial frame A, but it is attached to the origin of the robot's body-fixed frame $[x_b, y_b, z_b]^T$. The cylindrical representation of the error vector becomes $P_e^c(t) = [\rho_e, \psi_e, z_e, \theta_e]^T$ where $\psi_e = \psi - \eta$, denotes the difference between the yaw angle ψ and the direction of the planar tracking error vector, η. When $\psi_e = 0$, the robot will point in the direction of fastest reduction of planar tracking error. To handle the under-actuated nature of the robot, we introduce a hybrid error function

$$\xi = \theta_e - c\tanh\theta\tanh(\rho_e\cos(\psi_e)) \tag{5}$$

where $c > 0$ is a constant. We note that the function ξ is not unique and can be more generally written as $\xi = \theta_e - cf_{\xi 1}(\theta)f_{\xi 2}(x_e, y_e, \psi_e)$, satisfying $f_{\xi 1}(0) = 0$ and $f_{\xi 2}(0, 0, \psi_e) = 0$. Furthermore, $cf_{\xi 1}(\theta)f_{\xi 2}(x_e, y_e, \psi_e)$ should be designed such that when $\psi_e = 0$ and $\rho_e > 0$, $|\xi| < |\theta_e|$, and when $\psi_e = \pm\pi$ and $\rho_e > 0$, $|\xi| > |\theta_e|$. These conditions are satisfied by the choice of ξ in Eq. (5).

With the hyprid error function, we define the modified tracking error vector $P_{e_a} = [\psi_e, z_e, \xi]^T$, which will be used in the backstepping control design. The derivative of the error vector P_{e_a} can be expressed in terms of the state variables, using the fact that $x_e = \rho_e\cos\eta$ and $y_e = \rho_e\sin\eta$ to derive

$$\begin{cases} \dot{\rho}_e = \cos(\eta)\dot{x}_e + \sin(\eta)\dot{y}_e \\ \dot{\eta} = \dfrac{1}{\rho_e}(\cos(\eta)\dot{y}_e - \sin(\eta)\dot{x}_e) \end{cases} \tag{6}$$

which are needed for computing $\dot{\xi}$ and $\dot{\psi}_e$ later.

With this formulation, trajectory tracking becomes a stabilization problem with respect to the error vector. The control objective is now to drive the modified error vector P_{e_a} to the origin. Later we discuss how P_{e_a} converges to zero implies the convergence of all elements of the original tracking error vector P_e to zero.

III. Backsteping-Based Control Design

A. Overview of Control Design

First we define the Lyapunov function $V(\xi, z_e, \psi_e) = \frac{1}{2}(\xi^2 + z_e^2 + \psi_e^2)$, and seek a controller that can make $\dot{V}$ negative-definite. One can write

$$\begin{aligned} \dot{V} &= \xi\dot{\xi} + z_e\dot{z}_e + \psi_e\dot{\psi}_e \\ &= \begin{bmatrix} \dot{\theta}_e - c\left(f_{\xi 1}\dot{f}_{\xi 2} + \dot{f}_{\xi 1}f_{\xi 2}\right) \\ \dot{z}_e \\ \dot{\psi}_e \end{bmatrix}^T \begin{bmatrix} \xi \\ z_e \\ \psi_e \end{bmatrix} \end{aligned}$$

Following the backstepping methodology, we define the new error variables

$$\begin{aligned} \zeta_1 &= \dot{z}_e + k_z z_e \implies \dot{z}_e = \zeta_1 - k_z z_e \\ \zeta_2 &= \dot{\psi}_e + k_\psi\psi_e \implies \dot{\psi}_e = \zeta_2 - k_\psi\psi_e \\ \zeta_3 &= \dot{\xi} + k_\xi\xi \implies \dot{\xi} = \zeta_3 - k_\xi\xi \end{aligned}$$

where k_z, k_ψ, and k_ξ are positive design constants. This leads to $\dot{V} = z_e(\zeta_1 - k_z z_e) + \psi_e(\zeta_2 - k_\psi\psi_e) + \xi(\zeta_3 - k_\xi\xi)$. The augmented Lyapunov function can then be defined as $V_A =$

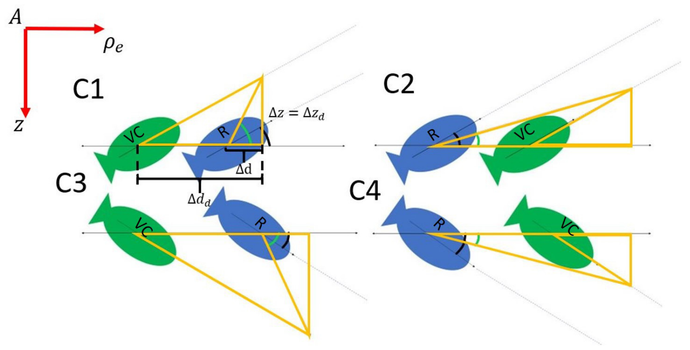

Fig. 3. Illustration of the desired behavior for the robot (R), when tracking trajectory is given by a virtual copy (VC) gliding in a plane for four different cases. Black angle marker represents $\theta = \theta_d$ and green angle marker represents $\theta = \theta_d + f_\xi(\theta, x_e, y_e, \psi_e)$. Δz and Δd represent the vertical and horizontal travel of robot when $\theta = \theta_d + f_\xi(\theta, x_e, y_e, \psi_e)$, respectively, while Δz_d and Δd_d represent the vertical and horizontal travel of the virtual copy, respectively.

$V + \frac{1}{2}(\zeta_1^2 + \zeta_2^2 + \zeta_3^2)$. The time-derivative $\dot{V}_A$ can be made negative-definite by using the control inputs which show up in the derivatives of ζ_i. Choosing the inputs such that $\dot{\zeta}_i = -k_i\zeta_i$ leads to $\dot{V}_A = \dot{V} + (-k_1\zeta_1^2 - k_2\zeta_2^2 - k_3\zeta_3^2)$, where k_1, k_2, and k_3 are positive constants to be chosen. Adding and subtracting the term $(\frac{1}{4k_z}\zeta_1^2 + \frac{1}{4k_\psi}\zeta_2^2 + \frac{1}{4k_\xi}\zeta_3^2)$ reveals that $\dot{V}_A < 0$ for the gain choices satisfying $k_1 k_z > \frac{1}{4}$, $k_2 k_\psi > \frac{1}{4}$, and $k_3 k_\xi > \frac{1}{4}$. This step in the Lyapunov analysis enables us to choose inputs u_1, u_2, and u_3 to ensure stabilization of the system. The equations $\dot{\zeta}_1 = \ddot{z}_e + k_z \dot{z}_e = -k_1\zeta_1$, $\dot{\zeta}_2 = \ddot{\psi}_e + k_\psi \dot{\psi}_e = -k_2\zeta_2$, and $\dot{\zeta}_3 = \ddot{\xi} + k_\xi \dot{\xi} = -k_3\zeta_3$ can be rewritten as

$$\dot{\zeta}_1 = f_{11}u_1 + f_{12}u_2 + f_{13}u_3 + f_{14}u_3^2 + f_{15} = -k_1\zeta_1$$
$$\dot{\zeta}_2 = f_{21}u_1 + f_{22}u_2 + f_{23}u_3 + f_{24}u_3^2 + f_{25} = -k_2\zeta_2$$
$$\dot{\zeta}_3 = f_{31}u_1 + f_{32}u_2 + f_{33}u_3 + f_{34}u_3^2 + f_{35} = -k_3\zeta_3$$

where $f_{ij} = \frac{\partial \dot{\zeta}_i}{\partial u_j}$ for $i = 1, 2, 3$, $j = 1, \cdots, 4$ and $f_{i5} = \dot{\zeta}_i - \sum_{j=1}^{4}(u_j f_{ij})$ with $u_4 = u_3^2$. These equations give us the means to solve for the inputs such that $\dot{\zeta}_i = -k_i\zeta_i$, in which case $\dot{V}_A = -\frac{1}{4k_z}(\zeta_1 - 2k_z z_e)^2 - \zeta_1^2(k_1 - \frac{1}{4k_z}) - \frac{1}{4k_\psi}(\zeta_2 - 2k_\psi \psi_e)^2 - \zeta_2^2(k_2 - \frac{1}{4k_\psi}) - \frac{1}{(4k_\xi)}(\zeta_3 - 2k_\xi \xi)^2 - \zeta_3^2(k_3 - \frac{1}{4k_\xi})$.

The controller can be derived in a much simpler form if we assume $\phi = 0$ and $\ddot{\eta} = 0$, both of which are reasonable for typical operating conditions of the gliding robotic fish. With this assumption, the equations for solving the control inputs can be written in a matrix form as follows:

$$\begin{bmatrix} f_{11} & 0 & f_{13} & f_{14} \\ 0 & 0 & f_{23} & 0 \\ f_{31} & f_{32} & f_{33} & f_{34} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_3^2 \end{bmatrix} = \begin{bmatrix} \Gamma_1 \\ \Gamma_2 \\ \Gamma_3 \end{bmatrix}$$

where $\Gamma_i = -f_{i5} - k_i\zeta_i$. The inputs can be then computed directly as

$$\begin{cases} u_3 = \dfrac{\Gamma_2}{f_{23}} \\ u_1 = \dfrac{1}{f_{11}}(\Gamma_1 - f_{13}u_3 - f_{14}(u_3)^2) \\ u_2 = \dfrac{1}{f_{32}}(\Gamma_3 - f_{33}u_3 - f_{34}(u_3)^2) \\ \quad - \dfrac{f_{31}}{f_{11}f_{32}}(\Gamma_1 - f_{13}u_3 - f_{14}(u_3)^2) \end{cases} \tag{7}$$

B. Analysis of the Closed-Loop System

Under the control design from the previous subsection, one can guarantee that ξ, ψ_e, and z_e all approach zero. Note that the original tracking goal is for all errors of P_e in Eq. (4) to approach zero. The closed-loop system is given by

$$\begin{bmatrix} \dot{z}_e \\ \dot{\psi}_e \\ \dot{\xi} \\ \dot{\theta}_e \\ \dot{\rho}_e \\ \dot{\zeta}_1 \\ \dot{\zeta}_2 \\ \dot{\zeta}_3 \end{bmatrix} = \begin{bmatrix} \zeta_1 - k_z z_e \\ \zeta_2 - k_\psi \psi_e \\ \zeta_3 - k_\xi \xi \\ \dot{\xi} - c\left(f_{\xi 1}\dot{f}_{\xi 2} + \dot{f}_{\xi 1} f_{\xi 2}\right) \\ \cos(\psi - \psi_e)\dot{x}_e + \sin(\psi - \psi_e)\dot{y}_e \\ -k_1\zeta_1 \\ -k_2\zeta_2 \\ -k_3\zeta_3 \end{bmatrix} \tag{8}$$

An instrumental tool for analysis of the original error vector will be singular perturbation theory for multi-time scale systems [30], where the dynamics of ρ_e and θ_e are at a time scale slower than the other dynamics in Eq. (8). Due to the limited space, we will skip the detailed rigorous analysis; instead, we examine only what happens in the boundary layer, where $\xi = 0, z_e = 0, \psi_e = 0, \zeta_1 = 0, \zeta_2 = 0, \zeta_3 = 0$, to shed the key light. When $\xi = 0$, $\theta_e = cf_{\xi 1}(\theta)f_{\xi 2}(x_e, y_e, \psi_e)$. This implies that $\theta = \theta_d + cf_{\xi 1}(\theta_e + \theta_d)f_{\xi 2}(x_e, y_e, \psi_e)$. In addition, $z_e = 0$ implies $z \equiv z_d$ and $\dot{z}_e = 0$, while $\psi_e = 0$ implies that the robot is pointing toward the desired direction (when projected onto the horizontal plane). From [5], [14], [15], it is understood that, similar to flight kinematics, glider kinematics can be expressed in terms of the magnitude of velocity and a glide angle θ_g. The glide angle is defined as $\theta_g = \theta - \alpha$, which can be approximated by $\frac{\Delta d}{\Delta z}$, where Δd and Δz are the horizontal and vertical distances traveled in a given amount of time. In practice, α is fairly small compared to θ; so $\theta_g \approx \theta$. This is illustrated by comparing the glide angle and the pitch angle in Section IV (see, for example, Figs. 4, 6). θ and θ_g differ in sign only for $|\theta| < 3°$ in simulations. At these small angles, the effect of the hybrid function are nearly non-existent due to the $\tanh(\theta)$ term. In essence, perturbing the pitch angle effectively changes the glide path, slowing or speeding up horizontal travel, so that the robot converges onto the trajectory in planar position. This behavior is further illustrated in Fig. 3. Once the $x - y$ planar position error converges to zero ($\rho_e = 0$), $\theta_e \to \xi = 0$.

IV. Simulation Results

The backstepping controller proposed in this paper is compared against a PID controller to show its effectiveness,

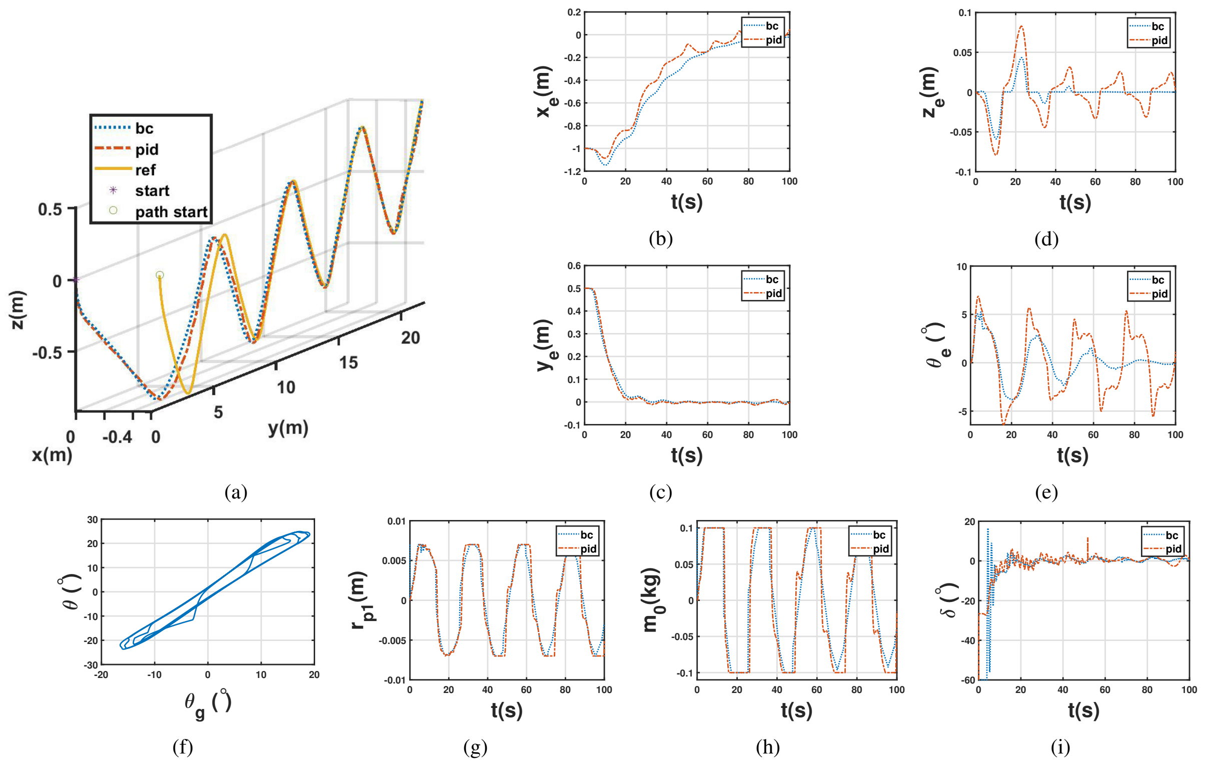

Fig. 4. Simulation results with a sawtooth-shaped reference trajectory constrained to a vertical plane. The legends “bc” and “pid” indicate results from the proposed backstepping controller and the PID controller, respectively. (a): Reference and controlled trajectories in the 3D space; (b)-(e): the trajectory of tracking errors $(x_e, y_e, z_e, \theta_e)$; (f): the graph showing the pitch angle vs. the glide angle under the proposed controller; (g)-(i): the trajectories of the control inputs (r_{p1}, m_0, δ).

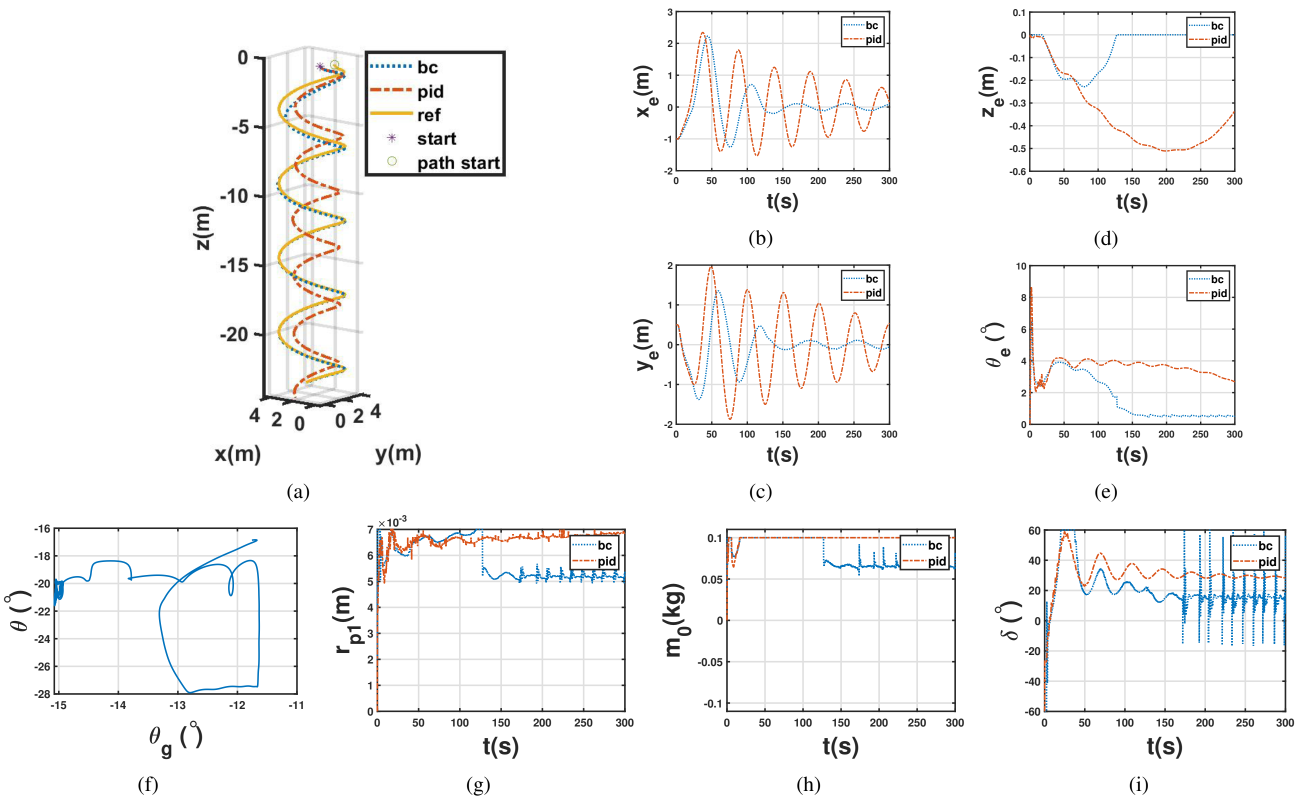

Fig. 5. Simulation results with a spiral reference trajectory. The legends “bc” and “pid” indicate results from the proposed backstepping controller and the PID controller, respectively. (a): Reference and controlled trajectories in the 3D space; (b)-(e): the trajectory of tracking errors $(x_e, y_e, z_e, \theta_e)$; (f): the graph showing the pitch angle vs. the glide angle under the proposed controller; (g)-(i): the trajectories of the control inputs (r_{p1}, m_0, δ)..

where three different trajectories representative of the types of motion underwater gliders experience are used. The model parameters for simulation are taken from [5] with limits of ± 0.1 kg on m_0, $\pm\frac{\pi}{3}$ on δ, and ± 7 mm on r_{p1}. In simulation, we redefine the tracking error ψ_e as $\psi - \arctan(y_h/x_h)$ if $\rho_e < \epsilon$ and $\psi - \eta$ otherwise, where $x_h = x_e - l\cos\psi$ and $y_h = y_e - l\sin\psi$, for some small $l > 0$. This allows the tracking error to be defined at the point of singularity when $\rho_e = 0$. We take ϵ to be 0.1 and l to be 0.2. A small $\epsilon > 0$ is added to ρ_e in $\dot{\eta}$ given in Eq. (6) as well for numerical stability.

The PID controller is based on the error vector P_{e_a}. The error ψ_e is used to calculate δ with gains $k_p = 1$, $k_i = 0.001$ and $k_d = 0.1$. The error ξ is used to calculate r_{p1} with gains $k_p = 0.083$, $k_i = 0.036$ and $k_d = 0.042$. The error z_e is used to calculate m_0 with gains $k_p = 10$, $k_i = 0.01$ and $k_d = 0$. The PID controller gains are tuned with the Matlab PID tuner and then refined through simulation runs to give good tracking performance on the trajectory shown in Fig. 6. The parameters for both the PID and the backstepping controller are kept the same over all three trajectories.

The backstepping controller requires state feedback; however, in practice, only measurements of Euler angles, angular velocities, and depth are readily available. Therefore, a sliding mode observer, as suggested in [4], is implemented to obtain the estimates of v_1, v_2, and v_3 using the aforementioned measurements. The desired trajectories are generated by a virtual copy of the robot using the same parameters as the actual robot.

The three reference trajectories include a sawtooth-shaped gliding pattern, a spiral pattern, and a more difficult trajectory generated by a time-dependent input to the virtual robot. Fig. 4 shows the simulation results for the case of the sawtooth-like reference trajectory. It can be seen that while the PID controller reduces x_e slightly faster, the backstepping controller results in smaller z_e and θ_e. The control effort under the backstepping controller is less than the effort under the PID controller for r_{p1} and m_0, with the PID controller hitting the saturation more often. For the tail angle δ, the PID controller has the smaller control effort for the first 15 to 20 seconds, but the backstepping controller has the smaller control effort for the rest of the trajectory.

Fig. 5 shows the simulation results for tracking the spiral reference. Here, the backstepping controller shows evident advantages in tracking all four elements of the trajectory, resulting in significantly smaller x_e, y_e, z_e, and θ_e. A chattering effect occurs in the input for the backstepping controller during most of the last half of the trajectory. This is likely due to the definition of ψ_e. It can change rapidly when $\rho_e = \sqrt{x_e^2 + y_e^2}$ approaches ϵ, causing the the tail angle δ to be large and discontinuous at times.

Fig. 6 shows the results for the case of a more arbitrary reference trajectory in the 3D space. While the PID controller is able to track the reference reasonably well, it is again outperformed by the backstepping controller, the resulting trajectory of which almost perfectly overlaps with the desired one after an initial transient. The chattering effect takes place in both controllers during the last 100 seconds.

V. Conclusion and Future Work

In this work we presented a backstepping-based controller for a gliding robotic fish to track a 3D reference trajectory along with a pitch angle reference trajectory using only three actuation inputs. The key enabling factor was the introduction of a hybrid error function, combining the pitch tracking error with the planar position tracking error. We argued that, at a slower time scale, the hybrid error function so constructed drives both the pitch error and the planar tracking error to zero. Simulation with three different reference trajectories showed that the backstepping controller is indeed able to track both 3D position and pitch references successfully. Overall, the proposed controller also showed advantages over a tuned PID controller in terms of tracking performance and control effort. We attribute this to the fact that the backstepping controller, unlike the PID controller, is able to incorporate the inherent coupling of the error dynamics. Although the proposed control approach addresses gliding robotic fish specifically in this work, we anticipate the methodology to be relevant to underwater gliders in general.

Future work will include formalizing the multi-time-scale singular perturbation analysis to establish the convergence proof. The controller will be further refined to address the chattering issue observed in simulation. We will also examine estimating $x - y$ planar position using an observer based on sporadic surface measurements, emulating the practical operation scenario of underwater gliders. The observer and controller designs will be implemented on a gliding robotic fish, and experimentally validated in pool and lake environments.

References

[1] H. Stommel, "The Slocum Mission," *Oceanography*, vol. 2, no. 1, pp. 22–25, 1989.

[2] J. Sherman, R. E. Davis, W. Owens, and J. Valdes, "The autonomous underwater glider "Spray"," *IEEE Journal of Oceanic Engineering*, vol. 26, no. 4, pp. 437–446, 2001.

[3] J. Sliwka, B. Clement, and I. Probst, "Sea glider guidance around a circle using distance measurements to a drifting acoustic source," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on.* IEEE, 2012, pp. 94–99.

[4] J. Yuan, Z. Wu, J. Yu, and M. Tan, "Sliding mode observer-based heading control for a gliding robotic dolphin," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 8, pp. 6815–6824, 2017.

[5] F. Zhang, "Modeling, design and control of gliding robotic fish," Dissertation, Michigan State University. Electrical Engineering, 2014.

[6] J. Wang and X. Tan, "A dynamic model for tail-actuated robotic fish with drag coefficient adaptation," *Mechatronics*, vol. 23, no. 6, pp. 659–668, 2013.

[7] F. Zhang, O. Ennasr, E. Litchman, and X. Tan, "Autonomous sampling of water columns using gliding robotic fish: Algorithms and harmful-algae-sampling experiments," *IEEE Systems Journal*, vol. 10, no. 3, pp. 1271–1281, 2016.

[8] E. Dong, S. Guo, X. Lin, X. Li, and Y. Wang, "A neural network-based self-tuning PID controller of an autonomous underwater vehicle," in *the Proceedings of the International Conference on Mechatronics and Automation, Chengdu, China*, 2012, pp. 5–8.

[9] K. Isa and M. R. Arshad, "Neural network control of buoyancy-driven autonomous underwater glider," in *Recent Advances in Robotics and Automation.* Springer, 2013, pp. 15–35.

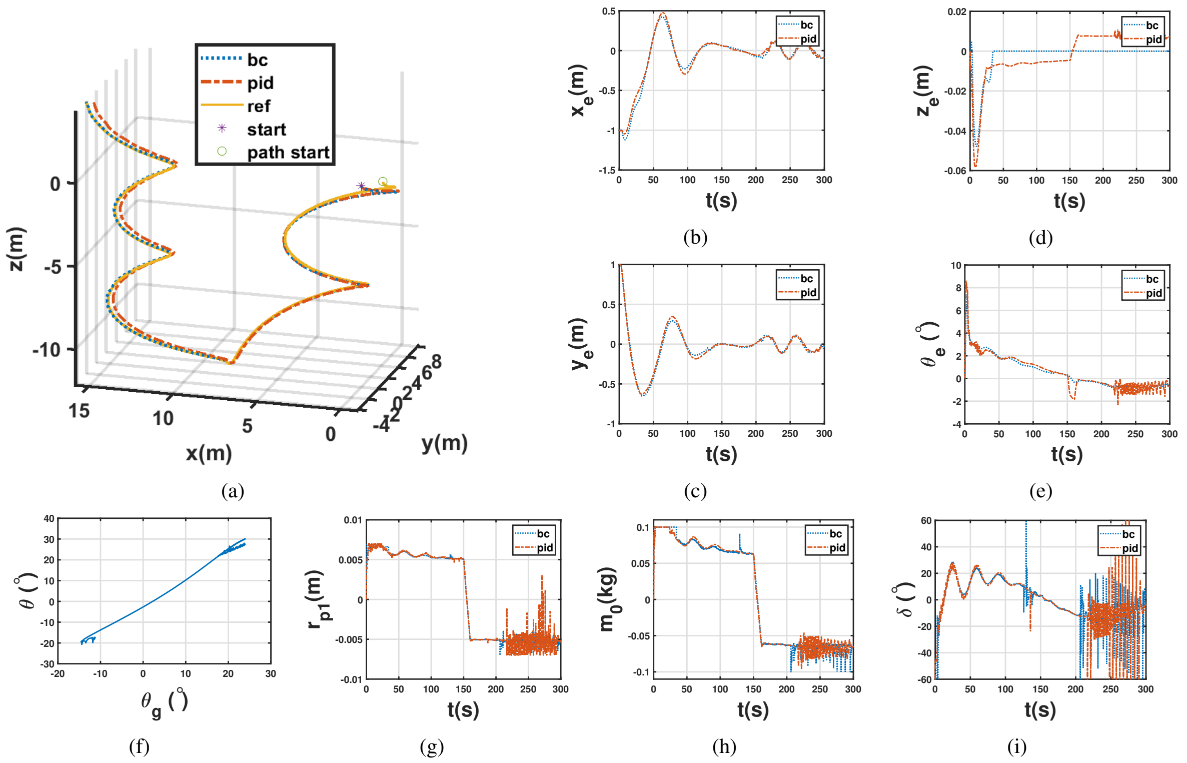

Fig. 6. Simulation results with a more arbitrary reference trajectory in 3D. The legends “bc” and “pid” indicate results from the proposed backstepping controller and the PID controller, respectively. (a): Reference and controlled trajectories in the 3D space; (b)-(e): the trajectory of tracking errors $(x_e, y_e, z_e, \theta_e)$; (f): the graph showing the pitch angle vs. the glide angle under the proposed controller; (g)-(i): the trajectories of the control inputs (r_{p1}, m_0, δ).

[10] A. Nag, S. S. Patel, and S. Akbar, “Fuzzy logic based depth control of an autonomous underwater vehicle,” in *Automation, Computing, Communication, Control and Compressed Sensing (iMac4s), 2013 International Multi-Conference on*. IEEE, 2013, pp. 117–123.

[11] F. Zhang, X. Tan, and H. K. Khalil, “Passivity-based controller design for stablization of underwater gliders,” in *American Control Conference (ACC), 2012*. IEEE, 2012, pp. 5408–5413.

[12] H. Yang and J. Ma, “Sliding mode tracking control of an autonomous underwater glider,” in *2010 International Conference on Computer Application and System Modeling (ICCASM 2010)*, vol. 4. IEEE, 2010, pp. V4–555.

[13] M. Mat-Noh, M. Arshad, R. Mohd-Mokhtar, and Q. Khan, “Control of an autonomous underwater glider using integral super-twisting sliding mode control (ISTSMC),” in *Underwater System Technology: Theory and Applications (USYS), 2017 IEEE 7th International Conference on*. IEEE, 2017, pp. 1–6.

[14] J. G. Graver and N. E. Leonard, “Underwater glider dynamics and control,” in *12th International Symposium on Unmanned Untethered Submersible Technology*, 2001, pp. 1742–1710.

[15] N. E. Leonard and J. G. Graver, “Model-based feedback control of autonomous underwater gliders,” *IEEE Journal of Oceanic Engineering*, vol. 26, no. 4, pp. 633–645, 2001.

[16] N. Mahmoudian and C. Woolsey, “Underwater glider motion control,” in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*. IEEE, 2008, pp. 552–557.

[17] B. Ullah, M. Ovinis, M. B. Baharom, M. Javaid, and S. Izhar, “Underwater gliders control strategies: A review,” in *Control Conference (ASCC), 2015 10th Asian*. IEEE, 2015, pp. 1–6.

[18] J. Cao, J. Cao, Z. Zeng, and L. Lian, “Nonlinear multiple-input-multiple-output adaptive backstepping control of underwater glider systems,” *International Journal of Advanced Robotic Systems*, vol. 13, no. 6, 2016. [Online]. Available: https://doi.org/10.1177/1729881416669484

[19] A. P. Aguiar and J. P. Hespanha, “Position tracking of underactuated vehicles,” in *Proceedings of the 2003 American Control Conference, 2003.*, vol. 3. IEEE, 2003, pp. 1988–1993.

[20] K. D. Do, J. Pan, and Z.-P. Jiang, “Robust and adaptive path following for underactuated autonomous underwater vehicles,” *Ocean Engineering*, vol. 31, no. 16, pp. 1967–1997, 2004.

[21] K. D. Do and J. Pan, *Control of Ships and Underwater Vehicles: Design for Underactuated and Nonlinear Marine Systems*. Springer Science & Business Media, 2009, ch. 12-13, pp. 109–124.

[22] F. Rezazadegan, K. Shojaei, F. Sheikholeslam, and A. Chatraei, “A novel approach to 6-DOF adaptive trajectory tracking control of an auv in the presence of parameter uncertainties,” *Ocean Engineering*, vol. 107, pp. 246–258, 2015.

[23] M. Karkoub, H.-M. Wu, and C.-L. Hwang, “Nonlinear trajectory-tracking control of an autonomous underwater vehicle,” *Ocean Engineering*, vol. 145, pp. 188–198, 2017.

[24] J. Wang, C. Wang, Y. Wei, and C. Zhang, “Command filter based adaptive neural trajectory tracking control of an underactuated underwater vehicle in three-dimensional space,” *Ocean Engineering*, vol. 180, pp. 175–186, 2019.

[25] Z. Zheng, L. Ruan, and M. Zhu, “Output-constrained tracking control of an underactuated autonomous underwater vehicle with uncertainties,” *Ocean Engineering*, vol. 175, pp. 241–250, 2019.

[26] R. Xu, G. Tang, L. Han, and D. Xie, “Trajectory tracking control for a CMG-based underwater vehicle with input saturation in 3D space,” *Ocean Engineering*, vol. 173, pp. 587–598, 2019.

[27] H. K. Khalil, *Nonlinear Systems*, 3rd ed. Prentice Hall, Upper Saddle River, NJ, 2002.

[28] D. Coleman, M. Castanon, and X. Tan, “Backstepping-based trajectory tracking for underwater gliders,” in *ASME 2019 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers Digital Collection, 2019, paper 9028.

[29] F. Zhang, J. Wang, J. Thon, C. Thon, E. Litchman, and X. Tan, “Gliding robotic fish for mobile sampling of aquatic environments,” in *Networking, Sensing and Control (ICNSC), 2014 IEEE 11th International Conference on*. IEEE, 2014, pp. 167–172.

[30] P. Kokotovic, H. K. Khali, and J. O’reilly, *Singular Perturbation Methods in Control: Analysis and Design*. SIAM, 1999, vol. 25.