Proceedings of the ASME 2019
Dynamic Systems and Control Conference
DSCC2019
October 8-11, 2019, Park City, Utah, USA

DSCC2019-9134

ADAPTIVE SINGLE ACTION CONTROL POLICIES FOR LINEARLY PARAMETERIZED SYSTEMS *

Osama Ennasr†
Smart Microsystems Lab
Department of Electrical and Computer Engineering
Michigan State University
East Lansing, MI 48823, USA
ennasros@msu.edu

Giorgos Mamakoukas
Neuroscience and Robotics Lab
Department of Mechanical Engineering
Northwestern University
Evanston, IL 60208, USA
giorgosmamakoukas@u.northwestern.edu

Maria Castaño
Smart Microsystems Lab
Department of Electrical and Computer Engineering
Michigan State University
East Lansing, MI 48823, USA
castanom@msu.edu

Demetris Coleman
Smart Microsystems Lab
Department of Electrical and Computer Engineering
Michigan State University
East Lansing, MI 48823, USA
colem404@msu.edu

Todd Murphey
Neuroscience and Robotics Lab
Department of Mechanical Engineering
Northwestern University
Evanston, IL 60208, USA
t-murphey@northwestern.edu

Xiaobo Tan
Smart Microsystems Lab
Department of Electrical and Computer Engineering
Michigan State University
East Lansing, MI 48823, USA
xbtan@egr.msu.edu

ABSTRACT

This paper presents an adaptive, needle variation-based feedback scheme for controlling affine nonlinear systems with unknown parameters that appear linearly in the dynamics. The proposed approach combines an online parameter identifier with a second-order sequential action controller that has shown great promise for nonlinear, underactuated, and high-dimensional constrained systems. Simulation results on the dynamics of an underwater glider and robotic fish show the advantages of introducing online parameter estimation to the controller when the model parameters deviate from their true values or are completely unknown.

*This work was supported by the National Science Foundation (IIS 1319602, ECCS 1446793, IIS 1715714)

†Address all correspondence to this author.

Introduction

Model-based techniques such as single-action control policies (SAP) [1–3], Nonlinear Model Predictive Control (NMPC) [4], differential dynamic programming (DDP) [5], and trajectory optimization [6] can generate highly efficient motions that leverage, rather than fight, the dynamics of robotic mechanisms. To effectively generate and track such trajectories, good knowledge of the underlying system model and its parameters is often required. In most control applications, however, various unknowns or uncertainties exist in the plant and its environment. Therefore, control systems are required to perform autonomously and intelligently under a variety of operating conditions. To maintain stability and performance, a successful design must deal with the unknown model parameters. Whether these parameters are completely unknown or change with time in an unpredictable manner, it is natural and effective to use adaptive control strategies [7–9].

Recently, SAP controllers such as first-order and second-order sequential action control (SAC) have emerged as a fam-

ily of model-based algorithms that seek to find short-duration control actions that maximally *improve*, rather than optimize, an objective over a prediction horizon [1, 3]. These controllers can be executed quickly and efficiently, which is often a limitation to traditional methods such as NMPC. In fact, the control solution has a closed-form expression, allowing single-action policies to compute real-time, closed-loop responses to a robot's dynamical environment [10].

In some control applications, model parameters can be difficult to measure accurately and in some cases they cannot be measured at all. Rather than estimating these parameters offline, adaptive control relies on an online parameter estimator that works in conjunction with the controller to measure these parameters. This approach is especially useful to systems whose parameters depend on the environment they operate in. The most popular method of designing adaptive control is the certainty equivalence principle. That is, a perfect knowledge control is first designed assuming all parameters are known; then, the adaptive control is the same except that the unknown parameters are replaced by their corresponding estimates; last, adaptation laws are synthesized to generate the estimates in such a way that the closed-loop stability and performance are guaranteed. Standard adaptive control results in [11] and [12] are based on the certainty equivalence principle and are shown to be effective for the class of systems whose unknown parameters appear linearly in their dynamics [13].

Adaptive control of linear time invariant systems is a well-established discipline whose major theoretical issues have been fully addressed. Predominant examples of adaptive strategies for such systems include adaptive pole placement control (APPC) [14], model reference adaptive control (MRAC) [15], and adaptive model predictive control (AMPC) [16]. These strategies, however, are generally inapplicable to real-world problems as many applications are inherently nonlinear.

Adaptive control of nonlinear systems, on the other hand, is an active research area as finding a control law that stabilizes the nonlinear dynamics is challenging and system-specific. Among the available techniques, adaptive input-output feedback linearization [17, 18] involves linearizing via feedback the nonlinear system such that the control law cancels all nonlinearities and forces the closed-loop system to behave as a linear time-invariant (LTI) system, while a parameter estimator is applied to estimate the system's unknown parameters. This approach is limited as it can only be applied to systems that are feedback-linearizable. Moreover, feedback linearization may not be able to cancel out the nonlinearities if they are large due to control saturation. Another popular nonlinear adaptive control strategy is adaptive backstepping control [11, 19, 20]. Backstepping control is also system specific as it requires cancellation of the cross-coupling terms to ensure the negativeness of the constructed Lyapunov function. These controllers exploit the specific structure of the nonlinear dynamics to simplify the process of adaptively controlling the system.

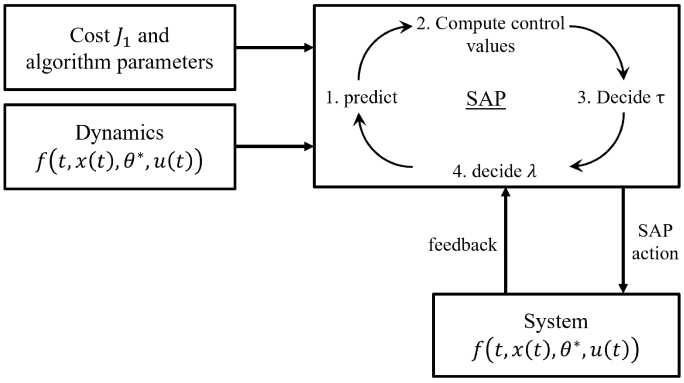


FIGURE 1: Overview of the SAP control process.

On the other hand, SAP controllers have shown applicability to a wide range of nonlinear systems. They also exhibit robustness to model and parameter inaccuracies, primarily by means of computing fast control-responses and operating in a reactive, rather than predictive, fashion [2, 10]. To further improve performance under model inaccuracies, we build on the second-order SAP controller presented in [3] by employing an adaptive structure. Specifically, we leverage the robustness and convergence guarantees of second-order SAP in stabilizing and controlling a large class of controllable, nonlinear and nonsmooth systems, coupled with an online parameter estimator.

The remainder of this paper is organized as follows. First, we provide an overview of the second-order SAC controller and how it computes the optimal control values. Then, we present the adaptive approach for controlling systems whose parameters are uncertain or completely unknown. We demonstrate the applicability of this approach with simulation results to trajectory tracking for tail-actuated swimming using a robotic fish and buoyancy-driven gliding in the sagittal plane. Finally, we provide some concluding remarks and future research directions.

Second-order single action policies

As a receding horizon scheme, SAP controllers incorporate feedback and apply a single control action at each time step as the horizon recedes, resulting in a real-time closed-loop response to the robot's dynamical environment. Figure 1 depicts the process in which a single-action policy applies a burst action—defined by the control value $u \in \mathbb{R}^m$, short application duration $\lambda \in \mathbb{R}^+$, and application time $\tau \in \mathbb{R}^+$—at each cycle. In this section, we review how the optimal control value of an action is derived for the specific second-order single-action policy we use. For a complete discussion on this controller, the reader can refer to [3].

We consider nonlinear, control-affine dynamics of the form

$$f\big(t,x(t),\theta^*,u(t)\big) = g\big(t,x(t),\theta^*\big) + h\big(t,x(t),\theta^*\big)u(t), \quad (1)$$

where $x : \mathbb{R} \mapsto \mathbb{R}^n$ is the state, $\theta^* \in \mathbb{R}^p$ is a vector of the true system parameters, and $u : \mathbb{R} \mapsto \mathbb{R}^m$ is the control.

Each cycle begins with the prediction phase,which forward simulates the motion using the nominal dynamics f_1 with control u_1 and defined by

$$f_1 \triangleq f(t,x(t),\theta^*,u_1(t)). \quad (2)$$

Let $l_1 : \mathbb{R}^n \mapsto \mathbb{R}$ and $m_1 : \mathbb{R}^n \mapsto \mathbb{R}$, the trajectory performance of the system is measured by the cost function

$$J_1 = \int_{t_0}^{t_f} l_1(x(t))dt + m_1(x(t_f)). \quad (3)$$

The prediction phase concludes with simulation of (2) and (3) over the interval $[t_0,t_f]$, where $t_f = t_0 + T$ is the end of the prediction horizon.

Next, the single-action policy computes a schedule (curve), $u_2^* : (t_0,t_f) \mapsto \mathbb{R}^m$, an action value at every moment along the predicted motion. Then, it chooses an application time τ to apply a single, fixed-value, action $u_2(\tau)$.

Given the application time $\tau \in [t_0,t_f]$, a (short) duration λ, and the optimal action value $u_2^*(\tau)$, the perturbed control signal is piecewise continuous. The resulting control signals u are real and bounded, such that

$$u(t) = \begin{cases} u_1(t), & t \notin [\tau - \frac{\lambda}{2}, \tau + \frac{\lambda}{2}] \\ u_2^*(\tau), & t \in [\tau - \frac{\lambda}{2}, \tau + \frac{\lambda}{2}] \end{cases}. \quad (4)$$

Hence, over each receding horizon, the controller assumes that the system evolves according to nominal dynamics f_1 except for a brief duration, where it switches to the alternate mode

$$f_2 \triangleq f(t,x(t),\theta^*,u_2^*(t)). \quad (5)$$

This problem borrows content from the mode scheduling literature [1], in the sense that the change in cost (3) due to short application of $u_2^*(\tau)$ can be reinterpreted as one of finding the change in cost due to inserting a new dynamic mode f_2 into the nominal trajectory for a short duration around $t = \tau$. Dropping time and parameter dependencies for brevity, the mode insertion gradient

$$\frac{dJ_1}{d\lambda_+}(\tau,u_2^*(\tau)) = \rho(\tau)^T \Big[f\big(x(\tau),u_2^*(\tau)\big) - f\big(x(\tau),u_1(\tau)\big) \Big], \quad (6)$$

provides a first-order model of the change in cost (3) relative to the duration of mode f_2, where the adjoint variable $\rho : \mathbb{R} \mapsto \mathbb{R}^n$ is also calculated from the nominal trajectory and is given by

$$\dot{\rho} = -\nabla l_1^T(x) - (\nabla f_1^T)\rho, \quad (7)$$

with $\rho(t_f) = \nabla m_1^T(x(t_f))$.

To incorporate the second-order information, the mode insertion Hessian, derived in [3], is given by

$$\begin{aligned} \frac{d^2J_1}{d\lambda_+^2}(\tau,u_2^*(\tau)) =& (f_2 - f_1)^T \Omega (f_2 - f_1) \\ &+ \rho^T(\nabla f_2 \cdot f_2 + \nabla f_1 \cdot f_1 - 2\nabla f_1 \cdot f_2) \\ &- \nabla l_1 \cdot (f_2 - f_1), \end{aligned} \quad (8)$$

where $\Omega : \mathbb{R} \mapsto \mathbb{R}^{n\times n}$ is the second-order adjoint state calculated from the nominal trajectory and is given by

$$\dot{\Omega} = -\nabla f_1^T \Omega - \Omega \nabla f_1 - \nabla^2 l_1 - \sum_{i=1}^{n} \rho_i \nabla^2 f_1^i, \quad (9)$$

subject to $\Omega(t_f) = \nabla^2 m_1^T(x(t_f))$.

Note that (6)–(9) assume the state in f_1 and f_2 corresponds to the default trajectory, which is simulated in the prediction phase with nominal control u_1.

The control solution that minimizes the second-order expansion of the cost sensitivity to the injected action is given by

$$u_2^*(t) = \left(\frac{\lambda^2}{2}\Gamma + R\right)^{-1} \left(\frac{\lambda^2}{2}\Delta + \lambda(-h^T\rho)\right) \quad (10)$$

where

$$\begin{aligned} \Delta \triangleq & \left(\left(h^T(\Omega^T + \Omega)h + 2h^T \cdot \left[\sum_{k=1}^{n} (\nabla h_k)\rho_k \right]^T \right) u_1 \right. \\ & \left. + (\nabla g \cdot h)^T \rho - \left[\sum_{k=1}^{n} (\nabla h_k)\rho_k \right] \cdot g + h^T \nabla l^T \right) \\ \Gamma \triangleq & \left(h^T(\Omega^T + \Omega)h + h^T \cdot \left[\sum_{k=1}^{n} (\nabla h_k)\rho_k \right]^T + \sum_{k=1}^{n} (\nabla h_k)\rho_k \cdot h^T \right). \end{aligned}$$

The parameter R is a positive definite matrix denoting a metric on control effort. After computing the schedule u_2^*, the second-order SAP chooses an application time τ and duration λ to apply a single action. For a complete discussion on choosing these parameters, along with proofs of convergence guarantees and convergence rate, the reader is referred to [3] and [21].

Adaptive single action policies

Adaptive control combines a parameter estimator, which generates parameter estimates online, with a feedback law to control classes of systems with unknown parameters. The way the parameter estimator is combined with the control law gives rise to two possible approaches [11]. In the first approach, referred to as indirect adaptive control, the plant parameters are estimated online and used to calculate the controller parameters. In the second approach, referred to as direct adaptive control, the plant model is parameterized in terms of desired control parameters, which are then estimated directly without intermediate calculations involving parameter estimates.

As indirect adaptive control can be applied to a wider class of systems (i.e. with different controller structures [11]), we employ an indirect approach to control the class of nonlinear systems given in (1), where the system parameters θ^* appear linearly in the dynamics. To that end, we consider the class of systems that can be written as

$$z(t) = \theta^{*T}\phi(t), \tag{11}$$

where $\theta^* \in \mathbb{R}^p$ is the vector of the (unknown) true parameters, and $z \in \mathbb{R}, \phi \in \mathbb{R}^p$ are signals available for measurement. A recursive least-squares (RLS) approach is employed to update an estimate. Assuming that $\phi \in \mathscr{L}_\infty$, the vector of parameter estimate $\theta(t) \in \mathbb{R}^p$ is updated by

$$\dot{P} = -P\phi\phi^T P, \qquad P(0) = P_0, \tag{12}$$

$$\dot{\theta} = P\varepsilon\phi, \qquad \theta(0) = \theta_0. \tag{13}$$

Here, $P_0 \in \mathbb{R}^{p\times p}$ is a symmetric positive definite matrix, while $\varepsilon(t) \in \mathbb{R}$ is the estimation error defined as

$$\varepsilon(t) \triangleq z(t) - \theta(t)^T\phi(t). \tag{14}$$

Figure 2 depicts the structure of the proposed adaptive controller. The online RLS parameter estimator in (13)–(14) is combined with second-order SAP and generates an estimate θ of the true parameters θ^*. The estimated parameters are fed back into the second-order SAP, where they are used in the prediction phase (2)-(3) and in computing optimal control values in (10). It is worth noting that the online parameter estimator in (13)–(14) monitors the system and updates the estimates continuously. However, since the second-order SAP controller only requires the latest available estimates at each cycle, it is possible to employ a discrete version of the RLS estimator, given by

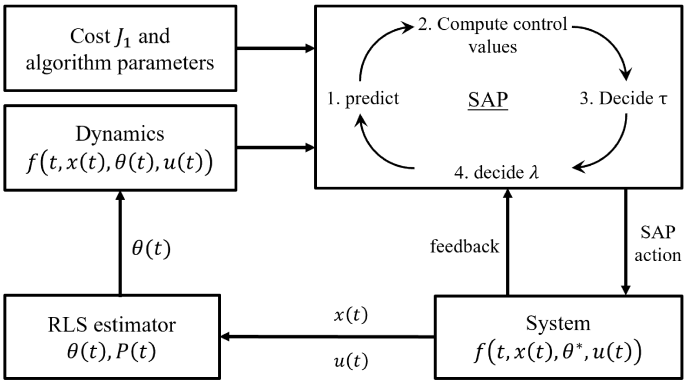


FIGURE 2: Overview of the proposed indirect adaptive SAP control architecture. The parameter estimator runs either continuously or discretely to update the modeled dynamics used by the controller for prediction and control synthesis.

$$P(k+1) = P(k) - \frac{P(k)\phi(k)\phi(k)^T P(k)}{m + \phi(k)^T P(k)\phi(k)}, \qquad P(0) = P_0, \tag{15}$$

$$\theta(k+1) = \theta(k) + \frac{P(k)\varepsilon(k)\phi(k)}{m}, \qquad \theta(0) = \theta_0. \tag{16}$$

Here, $P_0 \in \mathbb{R}^{p\times p}$ is a symmetric positive definite matrix, while $\varepsilon(k) \in \mathbb{R}$ is the estimation error at time instant k defined as

$$\varepsilon(k) \triangleq z(k) - \theta(k)^T\phi(k). \tag{17}$$

The proposed indirect approach to adaptive scheme is expected to perform well for linearly parametrized systems that can be written in the form (11). Since the schedule of SAC actions in (10) stabilizes the control-affine system in (1), the overall system can be shown to be stable by converse Lyapunov arguments as is discussed in [13]. While linear parameterization of a nonlinear system is a big assumption, it is not too restrictive as it is possible in many cases to overparameterize the system to simplify the adaptive control design at the expense of parameter convergence [13]. In the following section, we present how this adaptive scheme can be applied to nonlinear systems where overparameterization is utilized.

Simulation results

In this section we present simulation results for applying adaptive second-order SAP to achieve trajectory tracking for two nonlinear and underactuated robotic systems with unknown parameters: underwater gliding robotic fish in the sagittal plane, and tail-enabled swimming for robotic fish.

Adaptive control of gliding robotic fish in sagittal plane

We consider the case of applying the adaptive scheme to control the position of an underwater gliding robotic fish in the sagittal plane [22] that achieves its locomotion by adjusting its buoyancy and center of gravity. When restricting their motion in the sagittal plane, the dynamical model for these robots is expressed as

$$\dot{x} = v_1\cos(\theta) + v_3\sin(\theta) \tag{18}$$

$$\dot{z} = -v_1\sin(\theta) + v_3\cos(\theta) \tag{19}$$

$$\dot{\theta} = \omega_2 \tag{20}$$

$$\begin{aligned}\dot{v}_1 =& (m_1+\bar{m})^{-1}\left((m_3+\bar{m})v_3\omega_2 - m_0 g\sin(\theta)\right.\\ &\left.+L\sin(\alpha) - D\cos(\alpha)\right)\end{aligned} \tag{21}$$

$$\begin{aligned}\dot{v}_3 =& (m_3+\bar{m})^{-1}\left((m_1+\bar{m})v_1\omega_2 - m_0 g\cos(\theta)\right.\\ &\left.-L\sin(\alpha) - D\cos(\alpha)\right)\end{aligned} \tag{22}$$

$$\begin{aligned}\dot{\omega}_2 =& J_2^{-1}\left(M_2 + (m_3-m_1)v_1v_3\right.\\ &\left.-m_w g r_{w3}\sin(\theta) - \bar{m}g r_{p1}\cos(\theta)\right)\end{aligned} \tag{23}$$

with

$$L = \frac{1}{2}\rho V^2 S\left(C_{L0} + C_L^{\alpha}\alpha\right), \tag{24}$$

$$D = \frac{1}{2}\rho V^2 S\left(C_{D0} + C_D^{\alpha}\alpha^2\right), \tag{25}$$

$$M_2 = \frac{1}{2}\rho V^2 S\left(C_{M0} + C_{MP}^{\alpha}\alpha + K_{q2}\omega_2\right). \tag{26}$$

The translational velocities v_1 and v_3, and the angular velocity ω_2 are expressed in the body-fixed frame. The angle θ represents the robot's pitch, while $\alpha = \arctan(\frac{v_3}{v_1})$ is the angle of attack. The parameters m_1 and m_3 represent the stationary and added mass elements of the system, and J_2 is the inertia due to the stationary mass distribution and the added inertia in water. The term m_w accounts for the nonuniform hull mass distribution modeled as a point mass at a displacement of r_{w3} along the body-fixed z-axis. The movable mass $\bar{m}$ can move along the body-fixed x-axis. The control inputs are r_{p1} and m_0 which are, respectively, the position of the movable mass and the net buoyancy of the robot. The terms L, D, and M_2 are the hydrodynamic lift force, drag force, and pitch moment, respectively. The density of water is given by ρ, and S represents the robot's surface area. The term $V = \sqrt{v_1^2+v_3^2}$ is the velocity magnitude. Finally, the terms $C_{L0}, C_L^{\alpha}, C_{D0}, C_D^{\alpha}, C_{M0}, C_{MP}^{\alpha}$ and K_{q2} are the various hydrodynamic coefficients described in [22].

To fit (21)–(23) into a linear parametric model, the system is overparameterized by substituting (24)–(26) into (21)–(23). For example, substituting (24) and (25) into (21), and expanding the terms yields

$$\frac{s}{s+1}v_1 = \begin{bmatrix}\theta_{11}^* & \theta_{12}^* & \theta_{13}^* & \theta_{14}^* & \theta_{15}^* & \theta_{16}^*\end{bmatrix}\begin{bmatrix} -\frac{v_3\omega_2}{s+1} \\ -\frac{m_0\sin(\theta)}{s+1} \\ \frac{V^2\sin(\alpha)}{s+1} \\ \frac{\alpha V^2\sin(\alpha)}{s+1} \\ -\frac{V^2\cos(\alpha)}{s+1} \\ -\frac{\alpha^2 V^2\cos(\alpha)}{s+1}\end{bmatrix}, \tag{27}$$

with

$$\theta_{11}^* = \frac{m_3+\bar{m}}{m_1+\bar{m}}, \qquad \theta_{12}^* = \frac{g}{m_1+\bar{m}},$$
$$\theta_{13}^* = \frac{\rho S C_{L0}}{2(m_1+\bar{m})}, \qquad \theta_{14}^* = \frac{\rho S C_L^{\alpha}}{2(m_1+\bar{m})},$$
$$\theta_{15}^* = \frac{\rho S C_{D0}}{2(m_1+\bar{m})}, \qquad \theta_{16}^{*\,*} = \frac{\rho S C_D^{\alpha}}{2(m_1+\bar{m})}.$$

Similarly, the process was repeated for (22) and (23), yielding

$$\frac{s}{s+1}v_3 = \begin{bmatrix}\theta_{21}^* & \cdots & \theta_{26}^*\end{bmatrix}\begin{bmatrix}\phi_{21} & \cdots & \phi_{26}\end{bmatrix}^T, \tag{28}$$

$$\frac{s}{s+1}\omega_3 = \begin{bmatrix}\theta_{31}^* & \cdots & \theta_{36}^*\end{bmatrix}\begin{bmatrix}\phi_{31} & \cdots & \phi_{36}\end{bmatrix}^T. \tag{29}$$

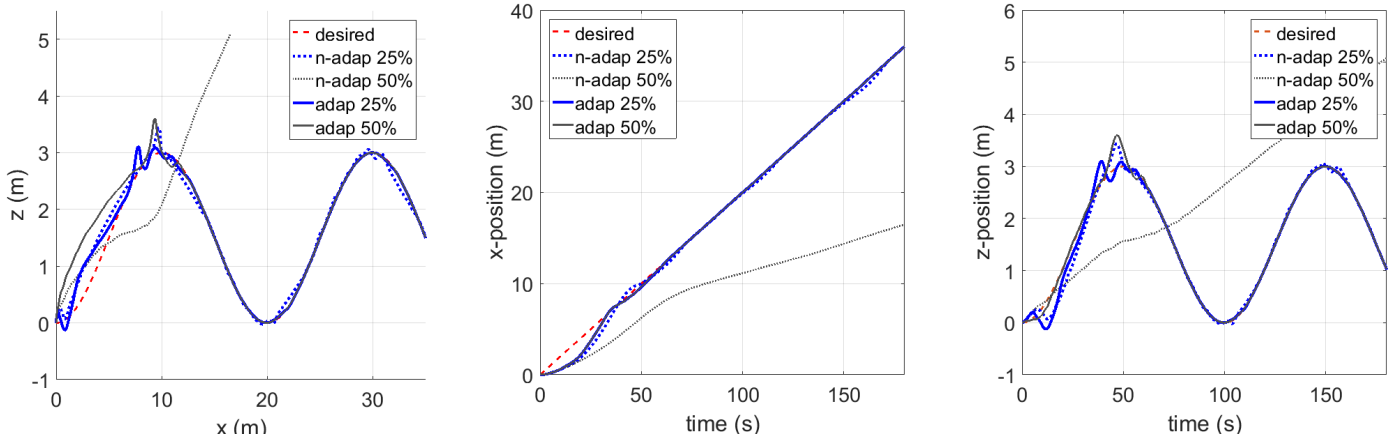
The second-order SAP controller and its adaptive counterpart were applied to the system, with a prediction horizon of $T = 3$ s and a sampling period of $t_s = 0.1$ s, where the true parameters of the system were obtained from [22]. The cost function used to measure the trajectory performance is given by

$$J_1 = \int_{t_0}^{t_f}\tilde{x}(t)^T Q_J\tilde{x}(t)dt + \tilde{x}(t_f)P_J\tilde{x}(t_f),$$

where $\tilde{x}(t) \triangleq x(t) - x^*(t)$ is the trajectory tracking error, while $Q_J = \mathrm{diag}(10^4, 10^4, 0, 0, 0, 0)$ and $P_J = \mathrm{diag}(10^3, 10^3, 0, 0, 0, 0)$ represent a metric for the running and terminal tracking error, respectively. The desired trajectory $x^*(t) = [x_d(t), z_d(t), 0, 0, 0, 0]^T$ was constructed to mimic multiple dives in the z-direction while the robot continues to travel forward, i.e.,

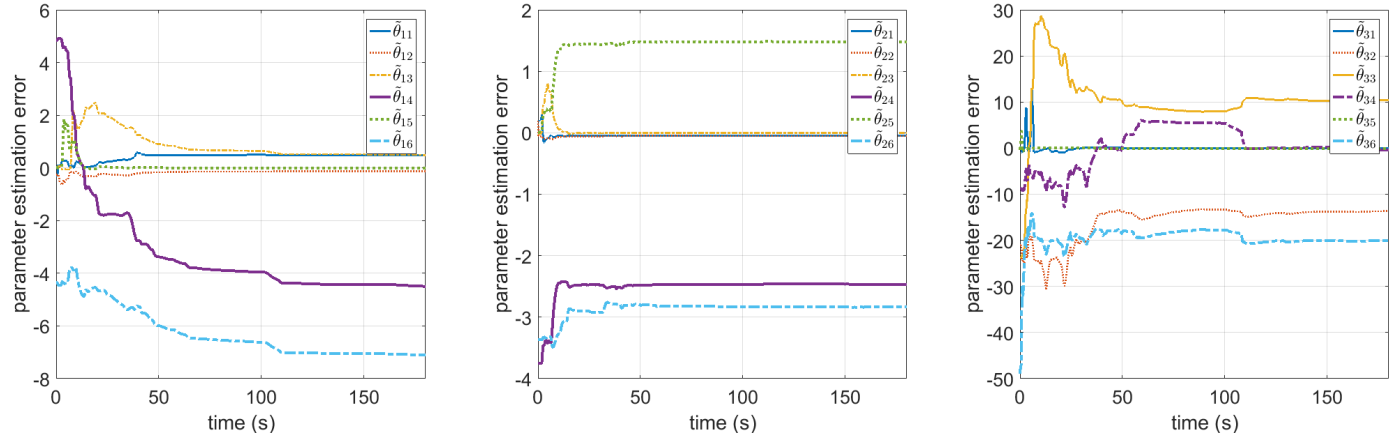
$$x_d(t) = 0.2t,$$
$$z_d(t) = 3\sin^2\left(\frac{2\pi}{200}t\right).$$

The parameters were estimated by applying the discrete-time RLS parameter identifier (15)–(17) to each of the parametric



(a) Trajectory tracking in sagittal plane. (b) Trajectory tracking in in x-axis. (c) Trajectory tracking in in z-axis.

FIGURE 3: Comparison between the adaptive approach (*adap*) and the non-adaptive SAP controller (*n-adap*) for trajectory tracking using an underwater glider in the sagittal plane. The percentage indicates the deviation of the model parameters from their true values. For the adaptive controller, this indicates the initial deviation of the parameter estimates from their true values.



(a) Parameter estimation error for $\theta_{11} \dots \theta_{16}$. (b) Parameter estimation error for $\theta_{21} \dots \theta_{26}$. (c) Parameter estimation error for $\theta_{31} \dots \theta_{36}$.

FIGURE 4: Error in parameter estimation for the underwater glider in the sagittal plane for the case where the parameters where initialized with $\pm 50\%$ of their true values.

models in (27)–(29). Each of the parameter identifiers was initialized with $P_1(0) = P_2(0) = P_3(0) = 5 \times 10^4 I$, while the initial parameter estimates were set through varying the system's true parameters by a certain percentage, i.e., $\theta(0) = (1 \pm \zeta)\theta^*$.

Figure 3 depicts the controller performance in tracking the desired trajectory. We compare the performance of the adaptive and non-adaptive controllers when the parameters deviate by $\pm 25\%$ and $\pm 50\%$ from their true values. The trajectories labeled *n-adap XX%* show the performance when the wrong parameters were used by the non-adaptive controller (i.e. the parameters were not updated and the initial parameters are used throughout the simulation). On the other hand, the trajectories *adap XX%* represent the system's performance when using the adaptive controller, and the parameters are estimated online. As is seen from Figure 3, estimating the parameters improves the system's performance. More importantly, comparing the tracking performance with 50% wrong parameters, the adaptive structure results in good tracking performance while the non-adaptive controller fails at tracking the desired trajectory.

Figure 4 shows the parameter estimation errors $\theta_1^* - \theta_1(t)$, $\theta_2^* - \theta_2(t)$, and $\theta_3^* - \theta_3(t)$. Only partial parameter convergence was achieved, as some parameters did not converge to their true

values. This issue is commonly encountered when the system is overparametrized to allow for linear parameterization of the system. However, the adaptive controller was still able to track the desired trajectory tracking with high accuracy.

Adaptive control of tail-actuated swimming for robotic fish

We now consider the case of applying the adaptive controller to perform trajectory tracking using a robotic fish with tail-actuated swimming. These robots achieve locomotion by continuously flapping their tail. Rather than controlling the tail position at every moment in time, it is typical to control the amplitude and bias of the tail flapping under a constant flapping frequency [23, 24]. The simplified averaged model of the robot is given by

$$\dot{x} = v_1\cos(\psi) - v_2\sin(\psi) \tag{30}$$

$$\dot{z} = v_1\sin(\psi) + v_2\cos(\psi) \tag{31}$$

$$\dot{\psi} = \omega_z \tag{32}$$

$$\begin{aligned}\dot{v}_1 =& \frac{m_2}{m_1}v_2\omega_z - \frac{c_1}{m_1}v_1\sqrt{v_1^2+v_2^2} \\ &+ \frac{c_2}{m_1}v_2\sqrt{v_1^2+v_2^2}\arctan(\frac{v_2}{v_1}) \\ &+ \frac{K_f mL^2}{12m_1}\omega_\alpha^2\alpha_a^2(3-\frac{3}{2}\alpha_0^2-\frac{3}{8}\alpha_a^2)\end{aligned} \tag{33}$$

$$\begin{aligned}\dot{v}_2 =& -\frac{m_1}{m_2}v_1\omega_z - \frac{c_1}{m_2}v_2\sqrt{v_1^2+v_2^2} \\ &- \frac{c_2}{m_2}v_2\sqrt{v_1^2+v_2^2}\arctan(\frac{v_2}{v_1}) \\ &+ \frac{K_f mL^2}{4m_2}\omega_\alpha^2\alpha_a^2\alpha_0\end{aligned} \tag{34}$$

$$\begin{aligned}\dot{\omega}_z =& (m_1-m_2)v_1v_2 - c_4\omega_z^2 sgn(\omega_z) \\ &- \frac{K_m mL^2}{4J_3}c\omega_\alpha^2\alpha_a^2\alpha_0\end{aligned} \tag{35}$$

where v_1,v_2, and ω_z are the surge, sway, and angular velocities of the robot, while x,y are the robot's position in 2D and ψ is its yaw angle. The various model parameters are represented by the variables $m_1, m_2, c, c_1, c_2, K_m, K_f, m, L$, and c_4. Finally, the tail flapping frequency is given by ω_α, while α_a and α_0 represent the flapping amplitude and bias, respectively.

Similarly to the underwater glider model, we can rewrite (33)–(35) in a linear parametric form

$$\frac{s}{s+1}v_1 = \begin{bmatrix}\theta_{11}^* & \cdots & \theta_{14}^*\end{bmatrix}\begin{bmatrix}\phi_{11} & \cdots & \phi_{14}\end{bmatrix}^T, \tag{36}$$

$$\frac{s}{s+1}v_2 = \begin{bmatrix}\theta_{21}^* & \cdots & \theta_{24}^*\end{bmatrix}\begin{bmatrix}\phi_{21} & \cdots & \phi_{24}\end{bmatrix}^T, \tag{37}$$

$$\frac{s}{s+1}\omega_z = \begin{bmatrix}\theta_{31}^* & \cdots & \theta_{33}^*\end{bmatrix}\begin{bmatrix}\phi_{31} & \cdots & \phi_{33}\end{bmatrix}^T. \tag{38}$$

The second-order SAP controller and its adaptive counterpart were applied to the system, controlling the squared flapping amplitude α_a^2 and flapping bias α_0 with the flapping frequency ω_z fixed at $3\pi/2$. The prediction horizon for both controller was set to $T = 10$ s with a sampling period of $t_s = 0.2$ s, where the true parameters of the system were obtained from [24]. The cost function used to measure the trajectory performance is given by

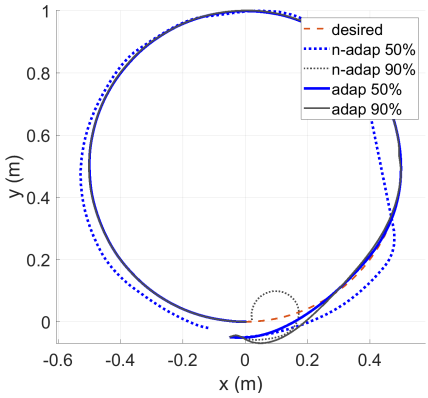
$$J_1 = \int_{t_0}^{t_f}\tilde{x}(t)^T Q_J\tilde{x}(t)dt + \tilde{x}(t_f)P_J\tilde{x}(t_f),$$

where $\tilde{x}(t) \triangleq x(t) - x^*(t)$ is the trajectory tracking error, while $Q_J = \text{diag}(10^4, 10^4, 0, 0, 0, 0)$ and $P_J = \text{diag}(10^3, 10^3, 0, 0, 0, 0)$ represent a metric for the running and terminal tracking error, respectively. The desired trajectory $x^*(t) = [x_d(t), y_d(t), 0, 0, 0, 0]^T$ was that of a circular trajectory centered around $(x, y) = (0, 0.5)$, i.e.,

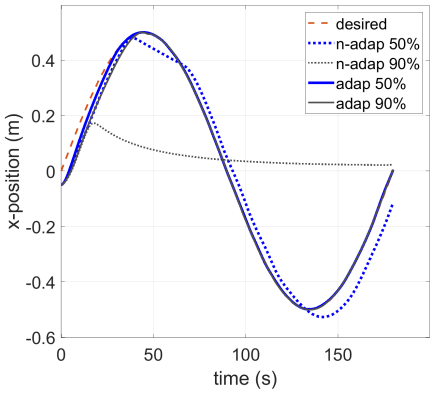
$$\begin{aligned}x_d(t) =& 0.5\sin\left(\frac{2\pi}{180}t\right), \\ y_d(t) =& 0.5\left(1-\cos\left(\frac{2\pi}{180}t\right)\right).\end{aligned}$$

The parameters were estimated by applying the discrete-time RLS parameter identifier (15)–(17) to each of the parametric models in (36)–(38). Each of the parameter identifiers was initialized with $P_1(0) = P_2(0) = P_3(0) = 10^8 I$, while the initial parameter estimates were set through varying the system's true parameters by a certain percentage, i.e., $\theta(0) = (1 \pm \zeta)\theta^*$.

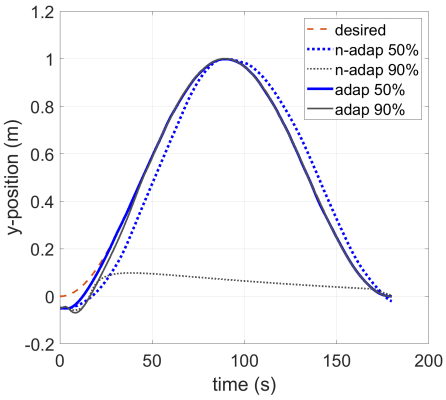
Figure 5 depicts the controller performance in tracking the desired trajectory using the tail-actuated robotic fish. We compare the performance of the adaptive and non-adaptive controllers when the parameters deviate by $\pm 50\%$ and $\pm 90\%$ from their true values. The trajectories labeled *n-adap XX%* show the performance when the wrong parameters were used by the non-adaptive controller (i.e. the parameters were not updated and the initial parameters are used throughout the simulation). On the other hand, the trajectories *adap XX%* represent the system's performance when using the adaptive controller, and the parameters are estimated online. As is seen from Figure 5, estimating



(a) Tracking a circular trajectory in the XY plane.

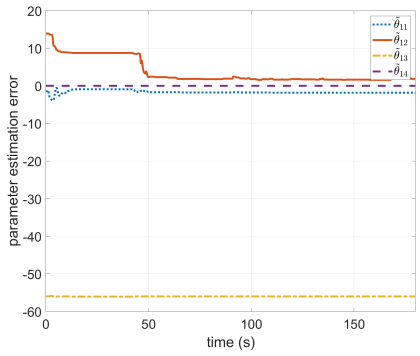


(b) Trajectory tracking in in x-axis.

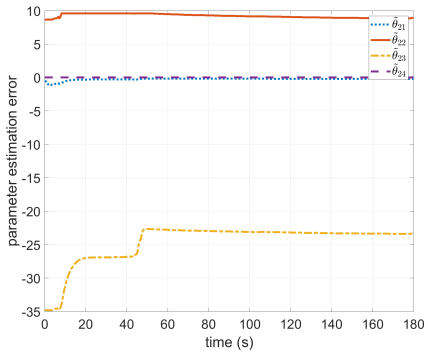


(c) Trajectory tracking in in y-axis.

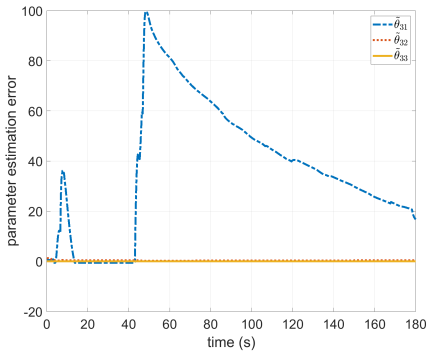
FIGURE 5: Comparison between the adaptive approach (*adap*) and the non-adaptive SAP controller (*n-adap*) for trajectory tracking using a tail-actuated robotic fish. The percentage indicates the deviation of the model parameters from their true values. For the adaptive controller, this indicates the initial deviation of the parameter estimates from their true values.



(a) Parameter estimation error for $\theta_{11} \dots \theta_{14}$.



(b) Parameter estimation error for $\theta_{21} \dots \theta_{24}$.



(c) Parameter estimation error for $\theta_{31} \dots \theta_{33}$.

FIGURE 6: Error in parameter estimation for the tail-actuated robotic fish for the case where the parameters where initialized with $\pm 90\%$ of their true values.

the parameters improves the system's performance. More importantly, comparing the tracking performance with 90% wrong parameters, the adaptive structure results in much improved tracking performance over using the non-adaptive controller.

Figure 6 shows the parameter estimation errors $\theta_1^* - \theta_1(t)$, $\theta_2^* - \theta_2(t)$, and $\theta_3^* - \theta_3(t)$. Similarly to the underwater glider system, only partial parameter convergence was achieved, as some parameters did not converge to their true values. This issue is commonly encountered when the system is overparametrized to allow for linear parameterization of the system. However, the adaptive controller was still able to track the desired trajectory tracking with high accuracy.

Conclusion and Future Work

To ensure successful operation, model-based algorithms require adequate description of the system's model and its underlying parameters. In some applications this poses a real limitation, as it might be difficult to measure some parameters directly, or because of the variability of these parameters depending on the environment the system is operating in. To overcome these challenges, it is desirable to estimate these parameters online while ensuring successful operation. In this paper we present a new adaptive scheme for nonlinear systems that combines a second-order single action policy controller with an online parameter estimator to update the system parameters used for prediction and control synthesis. This scheme offers a simple, yet effective, approach to controlling nonlinear systems with unknown parameters, and further improves the robustness of the SAP controller.

Simulation results are shown for applying the proposed adaptive structure to trajectory tracking for an underwater glider in the sagittal plane and tail-actuated robotic fish, and demonstrate the advantages of utilizing this structure when the model parameters are deviate from their true values, or are completely unknown. Future work will focus on the stability of the closed loop system under the proposed approach, with experimental work to further validate the applicability of this approach.

REFERENCES

[1] Ansari, A. R., and Murphey, T. D., 2016. "Sequential action control: Closed-form optimal control for nonlinear and nonsmooth systems.". *IEEE Trans. Robotics,* ***32***(5), pp. 1196–1214.

[2] Mamakoukas, G., MacIver, M. A., and Murphey, T. D., 2016. "Sequential action control for models of underactuated underwater vehicles in a planar ideal fluid". In 2016 American Control Conference (ACC), pp. 4500–4506.

[3] Mamakoukas, G., MacIver, M. A., and Murphey, T. D., 2018. "Feedback synthesis for underactuated systems using sequential second-order needle variations". *The International Journal of Robotics Research,* ***37***(13-14), pp. 1826–1853.

[4] Allgöwer, F., and Zheng, A., 2012. *Nonlinear model predictive control*, Vol. 26. Birkhäuser.

[5] Tassa, Y., Erez, T., and Smart, W. D., 2008. "Receding horizon differential dynamic programming". In Advances in neural information processing systems, pp. 1465–1472.

[6] Tassa, Y., Erez, T., and Todorov, E., 2012. "Synthesis and stabilization of complex behaviors through online trajectory optimization". In 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, pp. 4906–4913.

[7] Andrievsky, B., Peaucelle, D., and Fradkov, A. L., 2007. "Adaptive control of 3DOF motion for LAAS helicopter benchmark: Design and experiments". In 2007 American Control Conference, pp. 3312–3317.

[8] Pereira, A. R., Hsu, L., and Ortega, R., 2009. "Globally stable adaptive formation control of Euler-Lagrange agents via potential functions". In 2009 American Control Conference, pp. 2606–2611.

[9] Mingji, L., Zhongqin, C., Ximing, C., and Minggao, O., 2004. "Adaptive position servo control of permanent magnet synchronous motor". In Proceedings of the 2004 American Control Conference, Vol. 1, pp. 84–89 vol.1.

[10] Tzorakoleftherakis, E., Ansari, A., Wilson, A., Schultz, J., and Murphey, T. D., 2016. "Model-based reactive control for hybrid and high-dimensional robotic systems". *IEEE Robotics and Automation Letters,* ***1***(1), pp. 431–438.

[11] Ioannou, P., and Fidan, B., 2006. *Adaptive control tutorial*, Vol. 11. Siam.

[12] Krstic, M., Kanellakopoulos, I., Kokotovic, P. V., et al., 1995. *Nonlinear and adaptive control design*, Vol. 222. Wiley New York.

[13] Qu, Z., Hull, R. A., and Wang, J., 2006. "Globally stabilizing adaptive control design for nonlinearly-parameterized systems". *IEEE Transactions on Automatic Control,* ***51***(6), pp. 1073–1079.

[14] Lozano, R., and Zhao, X.-H., 1994. "Adaptive pole placement without excitation probing signals". *IEEE Transactions on Automatic Control,* ***39***(1), pp. 47–58.

[15] Limanond, S., and Tsakllis, K., 2000. "Model reference adaptive and nonadaptive control of linear time-varying plants". *IEEE Transactions on Automatic Control,* ***45***(7), pp. 1290–1300.

[16] Fukushima, H., Kim, T.-H., and Sugie, T., 2007. "Adaptive model predictive control for a class of constrained linear systems based on the comparison model". *Automatica,* ***43***(2), pp. 301–308.

[17] Sastry, S. S., and Isidori, A., 1989. "Adaptive control of linearizable systems". *IEEE Transactions on Automatic Control,* ***34***(11), pp. 1123–1131.

[18] Zarchi, H. A., Soltani, J., and Markadeh, G. A., 2010. "Adaptive input–output feedback-linearization-based torque control of synchronous reluctance motor without mechanical sensor". *IEEE Transactions on Industrial Electronics,* ***57***(1), pp. 375–384.

[19] Zhou, J., and Wen, C., 2008. *Adaptive backstepping control of uncertain systems: Nonsmooth nonlinearities, interactions or time-variations*. Springer.

[20] Matthew, J. S., Knoebel, N. B., Osborne, S. R., Beard, R. W., and Eldredge, A., 2006. "Adaptive backstepping control for miniature air vehicles". In 2006 American Control Conference, pp. 6–pp.

[21] Mamakoukas, G., MacIver, M. A., and Murphey, T. D., 2018. "Superlinear convergence using controls based on second-order needle variations". In 2018 IEEE Conference on Decision and Control (CDC), IEEE, pp. 4301–4308.

[22] Zhang, F., 2014. "Modeling, design and control of gliding robotic fish". PhD thesis, Michigan State University.

[23] Castaño, M. L., and Tan, X., 2016. "Model predictive control of a tail-actuated robotic fish". In ASME 2016 Dynamic Systems and Control Conference, American Society of Mechanical Engineers, pp. V001T03A006–V001T03A006.

[24] Castaño, M., and Tan, X., 2019. "Model predictive control-based path following for tail-actuated robotic fish". *Journal of Dynamic Systems, Measurement, and Control*.