Data-Driven, Non-Parametric Model Reference Adaptive Control Methods for Autonomous Underwater Vehicles

Derek I. Oesterheld

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Electrical Engineering

Daniel J. Stilwell, Chair
Andrew Kurdila
Stefano Brizzolara

September 15, 2023
Blacksburg, Virginia

Keywords: Adaptive Control, Kernel methods, Autonomous Underwater Vehicles

Data-Driven, Non-Parametric Model Reference Adaptive Control Methods for Autonomous Underwater Vehicles

Derek I. Oesterheld

(ABSTRACT)

This thesis details the implementation of two adaptive controllers on autonomous underwater vehicle(AUV) attitude dynamics starting from the standard six degree-of-freedom dynamic model. I apply two model reference adaptive control (MRAC) algorithms which make use of kernel functions for learning functional uncertainty present in the system dynamics. The first method extends recent results on model reference adaptive control using reproducing kernel Hilbert space (RKHS) learning techniques for some general cases of multi-input systems. The first controller design is a model reference adaptive controller (MRAC) based on a vector-valued RKHS that is induced by operator-valued kernels. This paper formulates a model reference adaptive control strategy based on a dead zone robust modification, and derives conditions for the ultimate boundedness of the tracking error in this case. The second controller is an implementation of the Gaussian Process MRAC developed by Chowdhary, et al. I discuss the method of each of these algorithms before contrasting the underlying theoretical structure of each algorithm. Finally, I provide a comparison of each algorithm's performance on the six degree-of-freedom dynamic model of the Virginia Tech 690 AUV and provide field trial results for the RKHS based MRAC implementation.

Data-Driven, Non-Parametric Model Reference Adaptive Control Methods for Autonomous Underwater Vehicles

Derek I. Oesterheld

(GENERAL AUDIENCE ABSTRACT)

This thesis details the implementation of two algorithms which control the attitude of an autonomous underwater vehicle. Rather than developing detailed dynamic models of the vehicles as is performed in classical control methods, each of these implementations only makes assumptions that the unknown portions of the dynamic models can be represented by a broad class of functions defined by a mathematical structure called a reproducing kernel Hilbert Space. Each algorithm implements learning techniques using the theory of reproducing kernel Hilbert spaces to bound the error between the vehicle attitude and the commanded vehicle attitude. One algorithm, called RKHS MRAC, implements an adaptive update law based on the attitude error to improve the controller performance. The second algorithm, called GP MRAC, uses estimated vehicle rotational accelerations and statistical learning methods to approximate the unknown function. Each of these methods is compared in theory and in a vehicle simulation. The RKHS MRAC is additionally demonstrated in field trial results.

Contents

List of Figures

List of Abbreviations

$\mathbb{R}$	The set of all real numbers
$\delta(t)$	control vector
$\langle \cdot, \cdot \rangle$	inner product operator
$\mathbb{N}$	The set of natural numbers
ϕ	Roll: the rotation about the x-axis of the vehicle BRF with respect to the IRF
ψ	Yaw: the rotation about the z-axis of the vehicle BRF with respect to the IRF
θ	Pitch: the rotation about the y-axis of the vehicle BRF with respect to the IRF
$\| \cdot \|$	norm operator
D_δ	Control space domain
D_x	State space domain
p	Roll rate: the rotation rate about the x-axis of the vehicle BRF with respect to the IRF
q	Pitch rate: the rotation rate about the y-axis of the vehicle BRF with respect to the IRF
r	Yaw rate: the rotation rate about the z-axis of the vehicle BRF with respect to the IRF
u	Surge: the velocity of the vehicle in the forward direction with respect to the BRF

v Sway: the velocity of the vehicle in the starboard direction with respect to the BRF

w Heave: the velocity of the vehicle in the downward direction with respect to the BRF

x The position of the vehicle in latitude with respect to an earth-fixed origin

$x(t)$ state vector

y The position of the vehicle in longitude with respect to an earth-fixed origin

z The position of the vehicle in altitude with respect to an earth-fixed origin

AUV Autonomous Underwater Vehicle

BRF Body Reference Frame

DPS Distributed Parameter System

IRF Inertial Reference Frame

Chapter 1

Introduction

In this work I evaluate two approaches to model reference adaptive control (MRAC) that both learn non-parametric representations of modeling errors and nonlinearities using kernel-based approaches. This is in contrast to classic model reference adaptive control approaches (e.g. see chapter 9 of [26]) where the designer must specify a parametric representation that may not necessarily be well-suited for a specific application.

To accomplish attitude control, in Chapter 2 I transform the general six degree of freedom (6-DOF) body reference frame AUV dynamics to inertial reference frame dynamic equations and formulate the dynamics such that I can use the GP-MRAC scheme. Traditionally the AUV dynamic modeling and control problem is formulated using the body reference frame dynamics as seen in [30],[29], and [40] to list several examples. The body reference frame AUV dynamics are typically preferred as they provide a more intuitive understanding of vehicle movement, are more readily linearizable, and admit simpler control designs[13]. While performing control in the inertial reference frame is not unique, see for example [35], adaptive 6-DOF control methods have typically been applied to the body reference frame dynamics [11],[20].

Chapter 3 of this work proposes a model-reference adaptive controller that leverages properties a reproducing kernel Hilbert space (RKHS) to model functional uncertainty in the plant. This work builds directly on a sequence of works [3, 17, 18, 19, 31], among others, that address native space RKHS embedding for adaptive estimation and control of uncertain ODE

systems. I specifically build on [7] that addresses model reference adaptive control embedded in native RKHS spaces for scalar-valued uncertainty and control signals. The techniques in [7] rely on conventional scalar reproducing kernels. I generalize to the case of a vector-valued uncertainty and vector-valued control signals using a native space RKHS induced by operator-valued kernel functions. The approach in this paper also extends the work on nonlinear observers in [6] and [16] to express the model-reference adaptive control problem using operator-valued kernel functions. I establish sufficient conditions for boundedness of the state of the controlled plant and that of the reference model, and articulate a specific upper bound. Furthermore, I illustrate the results on a realistic case-study of attitude control of a stream-lined tailed-controlled autonomous underwater vehicle. Such systems are often characterized by large uncertainties in hydrodynamics parameters, for which non-parametric data-driven models such as I use herein are well-suited, and also characterized by coupling between control channels for which a multi-input control approach is required.

The use in control theory of reproducing kernel Hilbert space (RKHS) embedding methods, also known as native space embedding, has a rich history in the related fields of machine learning, model estimation, and statistical inference (e.g. [34],[33], [2], [9], [37], [14]). In these fields approaches typically address nonlinear regression using discrete data sets consisting of input and output observations. In contrast, RKHS methods for designing continuous-time adaptive controllers and observers leverage the reproducing and self-adjoint properties of the associated kernel to develop continuous update laws for which I can determine convergence rates for functional estimates and apply Lyapunov analysis techniques to the study of system stability [3],[19].

While the theoretical ground work for scalar-valued RKHS methods has existed since the mid-twentieth century [1], the study of vector-valued RKHS methods is less mature, generally dating to the work of Michelli and Pontil in the context of machine learning theory [28].

More recently, significant effort has been devoted to the study of the properties of RKHS embedding methods in the continuous time control setting. References [3] and [19] establish conditions required for functional estimate convergence and an RKHS analog of the persistent excitation condition for adaptive estimation using scalar-valued RKHS embedding. The work in [6] extends the persistent excitation and convergence results to vector-valued functions and presents uniform ultimate boundedness guarantees for observer error using RKHS embedding methods.

In Chapter 4, I evaluate an MRAC approach which uses a kernel based approach in a Bayesian setting to yield a Gaussian process (GP) representation of functional uncertainty. This approach shares many features of the RKHS approach, since the GP representation implicitly sits in an RKHS, and possesses a very similar mathematical framework as the RKHS MRAC. The major difference being that the RKHS-MRAC assumes a deterministic and infinite-dimensional model of uncertainty, while the GP-MRAC assumes a stochastic model of uncertainty. For this evaluation I implement the GP-MRAC algorithm of [8],[27].

Chapter 5 gives numerical simulation results of each MRAC method and demonstrates validation of the RKHS-MRAC method of chapter 3 through field trial data from AUV operations in Claytor Lake near Dublin, VA.

Chapter 6 provides a comparison of the theoretical basis of each of the two MRAC algorithms and a side-by-side comparison of each algorithm's performance in a simulation of the VT-690 AUV.

Chapter 2

Vehicle Modeling

2.1 Vehicle Coordinate system

Following [30], I use the north-east-down coordinate system. The vehicle's position with respect to the inertial reference frame is described by the vector $\eta = \begin{bmatrix} x & y & z & \phi & \theta & \psi \end{bmatrix}^\mathsf{T} \in \mathbb{R}^6$. It is also useful to separately define $\eta_1 = \begin{bmatrix} x & y & z \end{bmatrix}^\mathsf{T}$ the position of the center of buoyancy of the vehicle, and $\eta_2 = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^\mathsf{T}$, the attitude of the body reference frame with respect to the inertial reference frame. The vehicle's motion is described in the BRF by the vector $\nu = \begin{bmatrix} u & v & w & p & q & r \end{bmatrix}^\mathsf{T} \in \mathbb{R}^6$. The velocity vector is related to the position vector by [13]

$$\dot{\eta} = J(\eta)\nu \tag{2.1}$$

where $J(\eta)$ is the rotation matrix

$$J(\eta) = \begin{bmatrix} J_1(\eta) & 0_{3\times3} \\ 0_{3\times3} & J_2(\eta) \end{bmatrix} \in \mathbb{R}^{6\times6}. \tag{2.2}$$

Explicit expressions for $J_1(\eta)$ and $J_2(\eta)$ may be found in [30].

2.2 Vehicle Dynamics

In this section, I consider the dynamics of a streamlined tail-controlled AUV. The dynamics are modeled using the standard six degree of freedom vehicle dynamics [30],[13]

$$M\dot{\nu} + C(\nu)\nu + D(\nu)\nu + g(\eta) = \tau \tag{2.3}$$

Where $M = M_{rb} - M_A \in \mathbb{R}^{6\times6}$ is the mass matrix, $C(\nu) = C_{rb}(\nu) - C_A(\nu) \in \mathbb{R}^{6\times6}$ is the coriolis matrix, $D(\nu) \in \mathbb{R}^6$ is the friction and damping vector, $g(\eta) \in \mathbb{R}^6$ is the gravitation force vector, $\tau \in \mathbb{R}^6$ is the forces and moments vector, which includes control forces and external forces or disturbances. Using (2.1), (2.3) can also be expressed solely in the inertial reference frame as [13]

$$\begin{gathered} M_\eta(\eta)\ddot{\eta} + C_\eta(\dot{\eta},\eta)\dot{\eta} + D\eta(\dot{\eta},\eta)\dot{\eta} + g(\eta) = \tau \\ M_\eta = MJ^{-1}(\eta) \\ C_\eta(\dot{\eta},\eta) = \Big(C(\dot{\eta}) - MJ^{-1}(\eta)\dot{J}(\eta)\Big)J^{-1}(\eta) \\ D_\eta(\dot{\eta},\eta) = D(\dot{\eta})J^{-1}(\eta) \end{gathered} \tag{2.4}$$

Because this work is concerned with attitude control, the attitude parameters are designated as the system states $x = \begin{bmatrix} x_1^\mathsf{T} & x_2^\mathsf{T} \end{bmatrix}^\mathsf{T} = \begin{bmatrix} \eta_2^\mathsf{T} & \dot{\eta}_2^\mathsf{T} \end{bmatrix}^\mathsf{T}$ Using this notation, consider only the dynamics of (2.4) associated with the rotational moments. Decomposing the dynamics of (2.4) into linear forces and rotational moments

$$\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} = \begin{bmatrix} M_{\eta_{11}} & M_{\eta_{12}} \\ M_{\eta_{21}} & M_{\eta_{22}} \end{bmatrix} \begin{bmatrix} \ddot{\eta}_1 \\ \dot{x}_2 \end{bmatrix} + \begin{bmatrix} C_{\eta_{11}} & C_{\eta_{12}} \\ C_{\eta_{21}} & C_{\eta_{22}} \end{bmatrix} \begin{bmatrix} \dot{\eta}_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} D_{\eta_{11}} & D_{\eta_{12}} \\ D_{\eta_{21}} & D_{\eta_{22}} \end{bmatrix} \begin{bmatrix} \dot{\eta}_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} g_1 \\ g_2 \end{bmatrix}. \tag{2.5}$$

The rotational moment equations are grouped in the second row of (2.5).

$$\tau_2 = M_{\eta_{21}}\ddot{\eta}_1 + M_{\eta_{22}}\dot{x_2} + C_{\eta_{21}}\dot{\eta}_1 + C_{\eta_{22}}x_2 + D_{\eta_{21}}\dot{\eta}_1 + D_{\eta_{22}}x_2 + g_2(x_1). \tag{2.6}$$

Vehicle control is accomplished using a two loop architecture. The outer loop controls the linear horizontal position and depth using a standard proportional-integral-differential (PID) controller which generates attitude commands for the attitude controller in the inner loop of the control system. Rewriting (2.5) in standard state space form, assuming that $M_{\eta_{22}}$ is invertible

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= M_{\eta_{22}}^{-1}\tau_2 - M_{\eta_{22}}^{-1}\Big(M_{\eta_{21}}\ddot{\eta}_1 + C_{\eta_{21}}\dot{\eta}_1 + C_{\eta_{22}}x_2 + D_{\eta_{21}}\dot{\eta}_1 + D_{\eta_{22}}x_2 + g_2(x_1)\Big). \end{aligned} \tag{2.7}$$

There are conditions under which M_η is singular, which are discussed in the sequel. From (2.4) I can write:

$$M_\eta^{-1} = (MJ^{-1}(\eta))^{-1} = J(\eta)M^{-1} \tag{2.8}$$

Following the approximations of [13], I assume that $M = M_{rb} + M_A$ is constant, $M_{rb} = M_{rb}^T \succ 0$ and $M_A = M_A^T \succ 0$ therefore $M = M^T \succ 0$, where $\succ$ indicates positive-definiteness, which also implies that M^{-1} exists. From [30], the rotation matrix has the form:

$$J(\eta) = \begin{bmatrix} J_1(\eta) & 0_{3\times 3} \\ 0_{3\times 3} & J_2(\eta) \end{bmatrix} \tag{2.9}$$

Therefore to assure that $M_{\eta_{22}}^{-1}$ exists I need only consider the matrix $J_2(\eta) \in \mathbb{R}^{3\times 3}$. Again

from [30]:

$$J_2(\eta) = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -sin(\phi) \\ 0 & \frac{\sin(\phi)}{\cos(\theta)} & \frac{\cos(\phi)}{\cos(\theta)} \end{bmatrix} \tag{2.10}$$

From (2.10) it is apparent that $M_{\eta_{22}}^{-1}$ exists provided that $\theta \in (-\frac{\pi}{2}, \frac{\pi}{2})$ which is an example of the classic gimbal lock condition when using Euler angles to describe vehicle attitude.

In subsequent chapters (2.7) is used to show that the AUV attitude dynamics can be formulated according to the dynamic requirements of each of the control algorithms.

2.3 Attitude Control

For control inputs, the Virginia Tech 690 (VT-690) AUV, shown in Figure 2.1 has a single rear thruster and four stern control surfaces in a cruciform configuration. The attitude controller considers these 4 control surfaces to be three virtual control surfaces: δ_e, the elevator angle, δ_r, the rudder angle, and δ_{roll}, the roll offset. For the purposes of attitude control, I ignore speed control and assume a constant propeller rate, thus the control input vector

$$\delta(t) = \begin{bmatrix} \delta_{roll} & \delta_e & \delta_r \end{bmatrix}^\mathsf{T} \in \mathbb{R}^3 \tag{2.11}$$

consists of only virtual control surface deflections.

Figure 2.1: Three VT-690 vehicles on a dock at Claytor Lake near Dublin, VA

From the assumptions on $\delta(t)$ of (2.11), τ_2 from (2.7) is modeled,

$$\tau_2 = u^2 \underbrace{\begin{bmatrix} K_{uu\delta_{roll}} & 0 & 0 \\ 0 & M_{uu\delta_e} & 0 \\ 0 & 0 & N_{uu\delta_r} \end{bmatrix}}_{\Lambda} \begin{bmatrix} \delta_{roll} \\ \delta_e \\ \delta_r \end{bmatrix} = u^2 \Lambda \delta. \tag{2.12}$$

where u is the surge velocity, $K_{uu\delta_{roll}}$, $M_{uu\delta_e}$, $N_{uu\delta_r}$ are the control surface effectiveness coefficients for each control surface. Although surge is a body reference frame velocity, it is retained in the notation for simplicity.

2.4 Reference Model

In the following sections several model reference adaptive control (MRAC) algorithms are implemented. These are variations on the standard MRAC algorithm as discussed in, for

example, [26]. For each algorithm a reference model

$$\dot{x}_r(t) = A_r x_r(t) + B_r r(t) \tag{2.13}$$

is designed that specifies the desired closed-loop behavior of the system. The goal of MRAC is to generate a control signal, $\delta(t)$ which results in $\|x(t) - x_r(t)\| \leq \bar{\epsilon}$ for an input signal $r(t)$ and $t \geq T$ for some T sufficiently large. Throughout this paper the error between the reference model states and the actual states is defined as

$$e(t) = x(t) - x_r(t) \tag{2.14}$$

In the ideal case the control signal would drive $\|e(t)\|$ to zero in finite time, however this is typically not achievable in real world control systems in the presence of non-linear and/or stochastic uncertainty.

The design of the reference model should incorporate the desired transient response characteristics of the closed-loop system and the physical limitations of the plant. As an example, the VT-690 AUV control surface actuators are limited to $\pm 20\,\text{deg}$ which in turn limits the angular rates achievable with control surface actuation. The reference model should incorporate this information and limit the transient behavior to those angular rates that are achievable by the system.

Using the above considerations and (2.7), the reference model is chosen to have the structure

$$\dot{x}_r(t) = \underbrace{\begin{bmatrix} 0_{3\times 3} & I \\ -K_{r_p} & -K_{r_d} \end{bmatrix}}_{A_r} \begin{bmatrix} x_{r_1}(t) \\ x_{r_2}(t) \end{bmatrix} + \underbrace{\begin{bmatrix} 0_{3\times 3} \\ K_p \end{bmatrix}}_{B_r} r(t) \tag{2.15}$$

which matches the structure of the closed-loop system of (2.7) where the control input only directly influences the angular velocity states. $K_p \in \mathbb{R}^{3\times3}$ and $K_d \in \mathbb{R}^{3\times3}$ are diagonal, positive-definite matrices designed to have the desired transient characteristics and to ensure that A_r is Hurwitz.

Chapter 3

Reproducing Kernel Hilbert Space MRAC

In this chapter, I propose a model-reference adaptive controller for the case of vector-valued uncertainty and control signals using a native space induced from an operator-valued kernel function. This work builds directly from recent results on model-reference adaptive control for scalar-valued functional uncertainty and control signals using native RKHS spaces induced from more typical scalar-valued kernel functions. Note that taking the Cartesian product of scalar-valued RKHS $\mathcal{H}$ to form a vector-valued space $\mathcal{H} \times \mathcal{H} \times \cdots \times \mathcal{H}$ is one way to formulate estimation and control problems for vector-valued functional uncertainty, and this seems a natural approach to extend initial efforts using scalar-valued spaces to more general control problems. But many vector-valued native spaces cannot be generated in this way. From first principles the approach in this paper enables MRAC formulations for any operator-valued kernel, which includes the Cartesian product as a very special case. In this way I define a very general strategy that is applicable to a much wider class of native spaces. This work addresses the case when a dead-zone modification is employed, for which I generate an ultimate bound between the state of the plant and the state of the reference model. I show in Proposition 3.1 that the ultimate bound is dependent on selection of the operator-valued kernel function as well as the number and location of kernel centers, which provide the control designer opportunities to reduce the bound. It is an explicit, known

function that depends on the location and number of centers of the approximating subspace used to represent the functional uncertainty. It is shown following Corollary 3.2 that this explicit ultimate bound holds for all functional uncertainty f in the (generally infinite dimensional) uncertainty class $\mathcal{B}_R := \{f \in \mathbb{H} \mid \|f\|_{\mathbb{H}} \leq R\}$. In this sense it is a wider robustness guarantee than those stated for uncertainty associated with ranges of real parameters from a fixed uncertainty model of finite dimension.

3.1 Problem Description

3.1.1 Background

I consider a nonlinear system with matched uncertainty

$$\dot{x}(t) = Ax(t) + B(u(t) + f(x)) \tag{3.1}$$

where $x(t) \in \mathbb{R}^n$ is the state, and $u(t) \in \mathbb{R}^m$ is a control signal. I assume that $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ are known constant matrices, and that $f : \mathbb{R}^n \to \mathbb{R}^m$ is an unknown function. I further assume that $f \in \mathbb{H}$, which is a native reproducing kernel Hilbert space induced by the operator-valued kernel $\mathbb{K}$ that maps $\mathbb{X} \times \mathbb{X}$ to the set of linear bounded functions $\mathcal{L}(\mathbb{Y})$. Throughout, I specialize to the case that $\mathbb{X} := \mathbb{R}^n$ and that $\mathbb{Y} := \mathbb{R}^m$.

Thorough treatments of RKHS and operator-valued kernels can be found in [32] and [16]. I provide herein only the background necessary for analysis of this specific model reference adaptive controller.

For any $x \in \mathbb{R}^n$, the evaluation operator is defined by $\mathbb{E}_x f = f(x) \in \mathbb{R}^m$. The vector-valued

analog of the reproducing property for the vector-valued RKHS is the identity

$$\langle \mathbb{K}_x u, f \rangle_{\mathbb{H}} = \langle u, \mathbb{E}_x f \rangle_{\mathbb{R}^m} , \tag{3.2}$$

which is satisfied for each $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, and $f \in \mathbb{H}$. Here the operator $\mathbb{K}_x \in \mathcal{L}(\mathbb{R}^m, \mathbb{H})$ is defined by the identity

$$(\mathbb{K}_x u)(z) := \mathbb{K}(z, x)u.$$

From the reproducing property (3.2) and properties of the adjoint, I note that the adjoint of the evaluation operator satisfies

$$\mathbb{E}_x^* u = \mathbb{K}_x u$$

for all $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$. Therefore the adjoint of the operator $\mathbb{K}_x$ is in fact the evaluation operator. Finally, given the operator-valued kernel $\mathbb{K}$, the vector-valued native space is defined as the closed linear span

$$\mathbb{H} = \overline{\text{span}\{\mathbb{K}_x u | x \in \mathbb{R}^n, u \in \mathbb{R}^m\}}. \tag{3.3}$$

3.1.2 Finite Approximations

To construct a finite-dimensional approximation of $f \in \mathbb{H}$, I assume that a finite set of distinct points $x_i \in \mathbb{R}^n$ have been selected $\Omega_N = \{x_i \in \mathbb{R}^n | 1 \leq i \leq N\}$. The points in Ω_N are referred to as centers, and I define the space of finite-dimensional approximants

$$\mathbb{H}_N := \text{span}\{\mathbb{K}_{x_i} e_j | x_i \in \Omega_N, 1 \leq j \leq m\} \tag{3.4}$$

where $\{e_j\}_{j=1}^m$ is any basis for $\mathbb{R}^m$. I assume that the kernel function $\mathbb{K}$ that defines $\mathbb{H}$ is *strictly* positive definite. This implies that the dimension of $\mathbb{H}_N$ is always mN. An

approximant $\hat{f}_N \in \mathbb{H}_N$ of $f \in \mathbb{H}$ is expressed

$$\hat{f}_N(\cdot) = \sum_{j=1}^{m} \sum_{i=1}^{N} \hat{\alpha}_k \mathbb{K}_{x_i}(\cdot) e_j, \ k = i + (j-1)N \tag{3.5}$$

for some set of scalar coefficients $\{\hat{\alpha}_k\}_{k=1}^{mN}$. In this work I consider time-varying approximants, and write

$$\hat{f}_N(t, \cdot) = \sum_{j=1}^{m} \sum_{i=1}^{N} \hat{\alpha}_k(t) \mathbb{K}_{x_i}(\cdot) e_j$$

for time-varying coefficients $\{\hat{\alpha}_k(t)\}_{k=1}^{mN}$. I assume that the set of centers Ω_N are selected only once. However, the choice of centers affects approximation accuracy, and selecting centers in real-time is an on-going research challenge that arises in many streaming (real-time) applications of RKHS.

For simplicity of notations, I omit dependency of functions on the spatial variable if there is no confusion (e.g. $f := f(\cdot)$ and $\hat{f}(t) := \hat{f}(t, \cdot)$). The orthogonal projection operator $\Pi_N : \mathbb{H} \to \mathbb{H}_N$ is characterized by [39]

$$\langle f - \Pi_N f, g \rangle_{\mathbb{H}} = 0, \ \forall \ g \in \mathbb{H}_N. \tag{3.6}$$

Define $f_N := \Pi_N f \in \mathbb{H}_N$. The estimation error between the function, f, and its finite dimensional approximant $\hat{f}_N$ is defined as

$$\tilde{f}(t, \cdot) := \hat{f}_N(t, \cdot) - f(\cdot),$$

and I note that $\tilde{f}(t, \cdot) \in \mathbb{H}$ for each fixed t. In this analysis, I find useful a specific decom-

position of the estimation error into mutually orthogonal parts

$$\begin{aligned}\tilde{f}(t) &= \hat{f}_N(t) - (f - \Pi_N f + \Pi_N f) \\ &= \underbrace{(\hat{f}_N(t) - \Pi_N f)}_{\tilde{f}_N(t)} - \underbrace{(f - \Pi_N f)}_{\tilde{f}_R}. \end{aligned} \tag{3.7}$$

The first term

$$\tilde{f}_N(t) := \hat{f}_N(t) - \Pi_N f \in \mathbb{H}_N$$

is the error between the approximant $\hat{f}_N(t, \cdot)$ and the projection of the function f into the finite-dimensional subspace $\mathbb{H}_N$, and

$$\tilde{f}_R := f - \Pi_N f \perp \mathbb{H}_N$$

is the residual error due to projecting the function onto the subspace.

3.1.3 MIMO MRAC Using RKHS

I consider a nonlinear system in (3.1), expressed as

$$\dot{x}(t) = Ax(t) + B(u(t) + \mathbb{E}_{x(t)} f) \tag{3.8}$$

where $f \in \mathbb{H}$. This approach follows a standard model-reference adaptive control(MRAC) formulation (see for example [26]). A reference model following the principles discussed in Section 2.4 is proposed. Again, the challenge is to generate a control signal $u(t)$ for (3.8) such that $\|x(t) - x_r(t)\| \leq \bar{\varepsilon}$ for an input signal $r(t)$ and $t \geq T$ for T sufficiently large, and where I have some influence over the value of $\bar{\varepsilon}$. For the reference model, $A_r \in \mathbb{R}^{n \times n}$ is

Hurwitz, and $B_r \in \mathbb{R}^{n\times l}$. The control law has the form

$$u(t) = \hat{K}_x^{\mathsf{T}}(t)x(t) + \hat{K}_r^{\mathsf{T}}(t)r(t) - \hat{f}_N(t, x(t)) \tag{3.9}$$

where $\hat{K}_x^{\mathsf{T}}(t) \in \mathbb{R}^{m\times n}$, $\hat{K}_r^{\mathsf{T}}(t) \in \mathbb{R}^{m\times l}$, and $\hat{f}_N(t,\cdot) \in \mathbb{H}_N$ are updated online according to the adaptation laws. Specifically, the function $\hat{f}_N \in \mathbb{H}_N$ is an online estimate of the unknown dynamics f.

I explicitly consider the dead zone modification method of the standard approach to MRAC (see Section 11.2.1 in [26], Section 4.6.4 in [12], among many others). The tracking error is denoted by (2.14) and the following parameter update laws are proposed

$$\dot{\hat{K}}_x(t) = \begin{cases} -M_x x(t) e^{\mathsf{T}}(t) PB & \text{if } \|e(t)\| \geq \bar{\epsilon} \\ 0 & \text{otherwise} \end{cases}, \tag{3.10}$$

$$\dot{\hat{K}}_r(t) = \begin{cases} -M_r r(t) e^{\mathsf{T}}(t) PB & \text{if } \|e(t)\| \geq \bar{\epsilon} \\ 0 & \text{otherwise} \end{cases}, \tag{3.11}$$

$$\dot{\hat{f}}_N(t,\cdot) = \begin{cases} \gamma_f \Pi_N \mathbb{E}^*_{x(t)} B^{\mathsf{T}} P e(t) & \|e(t)\| \geq \bar{\epsilon} \\ 0 & \text{otherwise} \end{cases}. \tag{3.12}$$

In these equations, the positive definite matrices $M_x \in \mathbb{R}^{n\times n}$ and $M_r \in \mathbb{R}^{l\times l}$, as well as the positive constant $\gamma_f \in \mathbb{R}$ affect the adaptation rates, and $P \in \mathbb{R}^{n\times n}$ is the solution to the algebraic Lyapunov equation

$$A_r^{\mathsf{T}} P + P A_r = -Q \tag{3.13}$$

where $Q \in \mathbb{R}^{n \times n}$ is a positive definite matrix that is selected as part of the control design process.

3.1.4 Tracking error dynamics and matching conditions

The tracking error dynamics with the control law (3.9) satisfy

$$\begin{aligned}\dot{e}(t) = Ax(t) + B\left(\hat{K}_x^{\mathsf{T}} x(t) + \hat{K}_r^{\mathsf{T}} r(t) - \mathbb{E}_{x(t)} \hat{f}_N(t, \cdot)\right) \\ + B\mathbb{E}_{x(t)} f(\cdot) - A_r x_r(t) - B_r r(t).\end{aligned}$$

By imposing the classical *matching conditions*

$$\begin{aligned}A + BK_x^{\mathsf{T}} &= A_r, \\ BK_r^{\mathsf{T}} &= B_r,\end{aligned} \tag{3.14}$$

where K_x and K_r are the so-called ideal adaptation gains, the tracking error dynamics can be written

$$\dot{e}(t) = A_r e(t) + B\tilde{K}_x^{\mathsf{T}}(t) x(t) + B\tilde{K}_r^{\mathsf{T}}(t) r(t) - B\mathbb{E}_{x(t)} \tilde{f}_N(t, \cdot), \tag{3.15}$$

where the error variables are

$$\begin{aligned}\tilde{K}_x(t) &:= \hat{K}_x(t) - K_x, \\ \tilde{K}_r(t) &:= \hat{K}_r(t) - K_r, \\ \tilde{f}(t, \cdot) &:= \hat{f}_N(t, \cdot) - f(\cdot).\end{aligned}$$

3.2 Analysis of Closed-Loop Performance

The tracking error dynamics (3.15) combined with the update laws of (3.10)-(3.12) define a distributed parameter system (DPS) whose state $\{e, \tilde{K}_x, \tilde{K}_r, \tilde{f}\}$ evolves in

$$\mathcal{A} := \mathbb{R}^n \times \mathbb{R}^{n\times m} \times \mathbb{R}^{l\times m} \times \mathbb{H}.$$

In this work I always assume that the error equations that define this DPS are forward complete. That is, for any initial condition I assume that the maximal interval of existing is $[0, \infty)$. Existence and uniqueness of solutions have been discussed for a similar DPS in native space embedding in [3, 17, 18, 19, 31].

Asymptotic behavior of the closed-loop system is characterized by ultimate boundedness of the tracking error $e(t)$.

Proposition 3.1. *Suppose that there is a compact set $\Omega \supset \cup_{\tau\geq 0} x(\tau)$ that contains the closed loop trajectory and the deadzone $\bar{\varepsilon}$ in the update laws (3.10)-(3.12) satisfies*

$$\bar{\varepsilon} \geq \frac{2\|PB\|C}{\lambda_{\min}(Q)} \tag{3.16}$$

where $C = \sup_{x\in\Omega} \|\mathbb{E}_x(f - \Pi_N f)\|$. Then there exists T such that the tracking error $e(t)$ of (3.15) combined with the update laws (3.10)-(3.12) satisfies

$$\|e(t)\| \leq \bar{\varepsilon} \tag{3.17}$$

for all $t \geq T$.

Proof. To establish ultimate boundedness of $e(t)$, I propose the Lyapunov function

$$v(e, \tilde{K}_x, \tilde{K}_r, \tilde{f}) = e^\mathsf{T} P e + \gamma_f^{-1} \langle \tilde{f}, \tilde{f} \rangle_{\mathbb{H}} + \text{trace}[\tilde{K}_x^\mathsf{T} M_x^{-1} \tilde{K}_x + \tilde{K}_r^\mathsf{T} M_r^{-1} \tilde{K}_r] \tag{3.18}$$

where $\tilde{f} = \hat{f}_N - f$. I first consider the case that $\|e(t)\| \geq \bar{\varepsilon}$. Differentiating (3.18) along the trajectory of error equations and substituting

$$\begin{aligned} \dot{\hat{K}}_x &= -M_x x(t) e^\mathsf{T} P B, \\ \dot{\hat{K}}_r &= -M_r r(t) e^\mathsf{T} P B, \\ \dot{\hat{f}}_N(t, \cdot) &= \gamma \Pi_N \mathbb{E}^* B^\mathsf{T} P e, \end{aligned}$$

yields

$$\begin{aligned} \dot{v} = &- e^\mathsf{T} Q e + 2 e^\mathsf{T} P B \tilde{K}_x^\mathsf{T} x + 2 e^\mathsf{T} P B \tilde{K}_r^\mathsf{T} r - 2 e^\mathsf{T} P B \mathbb{E}_x \tilde{f} \\ &+ 2 \gamma_f^{-1} \langle \dot{\hat{f}}_N, \tilde{f} \rangle \\ &- 2 e^\mathsf{T} P B \tilde{K}_x^\mathsf{T} x - 2 e^\mathsf{T} P B \tilde{K}_r^\mathsf{T} r. \end{aligned}$$

This simplifies to

$$\dot{v} = -e^\mathsf{T} Q e + 2 \gamma_f^{-1} \left\langle \dot{\hat{f}}_N, \tilde{f} \right\rangle - 2 e^\mathsf{T} P B \mathbb{E}_x \tilde{f}. \tag{3.19}$$

Recalling the decomposition of $\tilde{f} = \tilde{f}_N - \tilde{f}_R$ in (3.7), the relationship in (3.6), and the function estimation dynamics in (3.12) noting that $\dot{\hat{f}}_N(t, \cdot) \in \mathbb{H}_N$, the inner product in (3.19) can be

expressed

$$\begin{aligned}\langle \dot{\tilde{f}}_N, \tilde{f}\rangle &= \left\langle \tilde{f}_N - \tilde{f}_R, \dot{\tilde{f}}_N(t,\cdot)\right\rangle_{\mathbb{H}},\\ &= \left\langle \tilde{f}_N, \gamma_f \mathbb{E}_x^* B^{\mathsf{T}} P e\right\rangle_{\mathbb{H}} - \left\langle \tilde{f}_R, \dot{\tilde{f}}_N(t,\cdot)\right\rangle_{\mathbb{H}},\\ &= \left\langle \mathbb{E}_x \tilde{f}_N, \gamma_f B^{\mathsf{T}} P e\right\rangle_{\mathbb{R}^m} - 0,\\ &= \gamma_f e^{\mathsf{T}} P B \mathbb{E}_x \tilde{f}_N.\end{aligned}$$

Again from (3.7), $\tilde{f}_R = f - \Pi_N f$, and I can write

$$\begin{aligned}\dot{v} &= -e^{\mathsf{T}} Q e + 2 e^{\mathsf{T}} P B \mathbb{E}_x \tilde{f}_R\\ &\leq \|e\| \left(-\lambda_{\min}(Q)\|e\| + 2\|P\|\|B\|\|\mathbb{E}_x \left(I - \Pi_N\right) f\|\right).\end{aligned} \tag{3.20}$$

Thus $\dot{v} \leq 0$ whenever

$$\|e\| \geq \frac{2\|PB\|}{\lambda_{\min}(Q)} \sup_{x\in\Omega} \|\mathbb{E}_x(f - \Pi_N f)\| \tag{3.21}$$

where the right-hand side is a lower bound for the deadzone $\bar{\varepsilon}$. Thus (3.21) is satisfied whenever $\|e\| \geq \bar{\varepsilon}$. Following a standard argument in [12] or [26], for example, this is sufficient to ensure that $\|e\| \geq \bar{\varepsilon}$ for finite time. □

Proposition 3.1 is intuitively satisfying: if the deadzone $\bar{\epsilon}$ is scaled properly so that it provides a pointwise bound for the worst case approximation error $\|\mathbb{E}_x(f - \Pi_N f)\|$ for the functional uncertainty f, then the ultimate bound holds. One of the powerful properties of projection or interpolation operators in a native space is that such upper bounds are often readily available. In many of the recent papers [3, 17, 18, 19, 31] such bounds are obtained using the power function for a scalar-valued RKHS. In the paper [6], related ultimate bounds are derived for a class of nonlinear observers in vector-valued native spaces induced by an

operator-valued kernel. These bounds make use of a generalization of the power function for operator-valued kernels described recently in [39]. These define the operator-valued power function given by

$$(P_N^\alpha(x))^2 := \langle (\mathbb{K}(x,x) - \mathbb{K}_N(x,x))\alpha, \alpha \rangle_{\mathbb{R}^m}$$

for each $x \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}^m$. In this equation $\mathbb{K}_N(x,x)$ is the reproducing kernel of $\mathbb{H}_N$. Let $K \in \mathbb{R}^{mN \times mN}$ be the Gram block matrix defined by $K := \{\mathbb{K}(x_i, x_j)\}_{i,j=1}^N$ and let $k(x, \Xi) := \{\mathbb{K}(x, \xi_i)\}_{i=1}^N = \begin{bmatrix} \mathbb{K}(x,\xi_1) & \cdots & \mathbb{K}(x,\xi_n) \end{bmatrix} \in \mathbb{R}^{m \times mN}$. By definition K is positive semi-definite, therefore from Corollary 2.7 of [39] the reproducing kernel of $\mathbb{H}_N$ is given by

$$\mathbb{K}_N(x_i, x_j) = k(x_i, \Xi) K^\dagger k(x_j, \Xi)^\mathsf{T} \in \mathbb{R}^{m \times m} \tag{3.22}$$

where $K^\dagger$ is the Moore-Penrose pseudo-inverse of K. If K is positive definite then $K^\dagger = K^{-1}$. In the remainder of this work, K is specifically constructed to be positive-definite.

The operator-valued power function is of use in the present context since from Corollary 2.11 of [39] I have

$$|\langle \mathbb{E}_x (I - \Pi_N) f, \alpha \rangle_{\mathbb{R}^m}| \leq P_N^\alpha(x) \|f\|_{\mathbb{H}}$$

for all $x \in \mathbb{R}^n, \alpha \in \mathbb{R}^m$, and $f \in \mathbb{H}$. This gives an immediate corollary that ties the placement of centers to a bound that is known in closed form.

Corollary 3.2. *Let the hypotheses of Proposition 3.1 hold. Then the tracking error satisfies the ultimate bound*

$$\|e(t)\| \leq O\left(\sup_{x \in \Omega} \sqrt{\|\mathbb{K}(x,x) - \mathbb{K}_N(x,x)\|} \|f\|_{\mathbb{H}} \right).$$

Proof. The proof of this corollary follows immediately from the comments above and the proof of Proposition 3.1. See [6] for the details. □

It should be emphasized that Corollary 3.2 can be understood as a statement of robust adaptive control. I define the functional uncertainty class

$$\mathcal{B}_R := \{f \in \mathbb{H} \mid \|f\|_{\mathbb{H}} \leq R\},$$

and replace C in Equation 3.16 by

$$C := \sup_{x \in \Omega} \sqrt{\|\mathbb{K}(x,x) - \mathbb{K}_N(x,x)\|} R.$$

Then the ultimate bound of the Corollary holds for all functional uncertainty $f \in \mathcal{B}_R$ over the uncertainty class $\mathcal{B}_R$. Note that in contrast to approaches in real parametric adaptive control, the guarantee is over a (generally infinite dimensional) class of functional uncertainty, and not just over real parametric uncertainty in coordinate representations for a fixed dimensional model. In this sense the robustness guarantee is over a broader class of uncertain models than usually encountered in real parametric adaptive control.

Remark 3.3. We note that while the expressions of $\mathbb{K}$ and $\mathbb{K}_N$ are known in closed forms, computating centers Ω_N that attain the supremum on the right hand side involves combinatorial optimization. For scalar-valued case, greedy algorithms have been shown to reach near-optimal rate of convergence [15, 25].

3.3 Practical Implementation

As discussed in 3.1.2, using (3.5), I approximate the uncertain function f using a set of N kernel centers. Learning the approximate function, $\hat{f}_N$ amounts to learning the vector of coefficients $\alpha = \begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_{3N} \end{bmatrix}^\mathsf{T} \in \mathbb{R}^{3N}$ with each set of three coefficients associated with a kernel centered at one of the N centers. The set of kernel functions is given by $\bigcup_{i=1}^{N} \{\mathbb{K}_{\xi_i} e_1, \mathbb{K}_{\xi_i} e_2, \mathbb{K}_{\xi_i} e_3\}$ where $e_j \in \mathbb{R}^3$ is the standard basis vector with a one at entry j and zeros elsewhere. To develop an update law for α from (3.12), following a similar process to that described in [7], I take the inner product of $\dot{\hat{f}}_N$ with each basis function

$$\langle \mathbb{K}_{\xi_i} y_i, \dot{\hat{f}}_N \rangle_{\mathbb{H}} = \langle \mathbb{K}_{\xi_i} y_i, \sum_{k=1}^{3N} \dot{\hat{\alpha}}_k(t) \mathbb{K}_{\xi_k} y_k \rangle_{\mathbb{H}}$$

Using the reproducing property and the linearity properties of the inner product results in an update law with the form

$$\dot{\hat{\alpha}} = \gamma_f G^{-1} \Phi(x) \tag{3.23}$$

where $\{G\}_{jk} = y_j^\mathsf{T} \mathbb{K}(\xi_j, \xi_k) y_k \in \mathbb{R}^{3N \times 3N}$ and

$$\Phi(x) = \begin{bmatrix} e^\mathsf{T} P B \mathbb{K}(\xi_1, x) e_1 \\ e^\mathsf{T} P B \mathbb{K}(\xi_1, x) e_2 \\ \vdots \\ e^\mathsf{T} P B \mathbb{K}(\xi_N, x) e_3 \end{bmatrix}.$$

For the purposes of the numerical simulation in this paper the kernel centers are placed in a uniformly spaced grid throughout a subset of the function's input space ($\Omega \subset \mathbb{R}^6 \times \mathbb{R}^3$) and

I use a simple kernel function:

$$\mathbb{K}(x_1, x_2) = k_{5/2}(x_1, x_2) \begin{bmatrix} 1 & 0 & \frac{1}{2} \\ 0 & 1 & 0 \\ \frac{1}{2} & 0 & 1 \end{bmatrix} \tag{3.24}$$

where $k_{5/2}(x_1, x_2)$ is the scalar 5/2 Matern kernel parameterized by amplitude $a = 0.1$ and length scale $l = 0.3$ (see [34]). This kernel was chosen to capture the closely coupled nature of the roll and yaw behavior of the vehicle model.

The adaptation rate matrices and gains $M_x,\ M_r, \gamma_f$ were set empirically based on observations of the adaptive parameter behaviors.

Starting from (2.7) I make the additional simplifying assumptions that the center of gravity and center of buoyancy of the vehicle are collocated and that the vehicle's mass matrix is diagonal. The first assumption results in $g_2(\eta) = 0\ \forall\ \eta$. The second assumption results in $M_{\eta_{21}} = 0\ \forall\ \eta$. $M_{\eta_{22}}$ is a function of the roll and pitch due to the presence of a rotation matrix. Therefore, I additionally apply the small angle assumption which results in the rotation matrix being approximately equivalent to the identity matrix and reduces $M_{\eta_{22}}^{-1} = M_{22}^{-1}$ which is constant. Using these assumptions simplifies (2.7) to

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= M_{22}^{-1}\Big((\tau_2 - (C_{\eta_{21}} + D_{\eta_{21}})\dot{\eta}_1 - (C_{\eta_{22}} + D_{\eta_{22}})x_2 \Big) \end{aligned} \tag{3.25}$$

which is in the form required to apply the RKHS MRAC controller. Applying the model for τ_2, the functional uncertainty is given by

$$f(x, \dot{\eta}_1) = -\frac{1}{u^2}\Lambda^{-1}\big((C_{\eta_{21}} + D_{\eta_{21}})\dot{\eta}_1 + (C_{\eta_{22}} + D_{\eta_{22}})x_2\big). \tag{3.26}$$

For the purposes of expressing the functional uncertainty I convert the earth reference frame linear velocities, $\dot{\eta}_1$ in (3.25) to body reference frame linear velocities, surge, sway and heave, denoted collectively ν_1. Under the assumption of constant forward speed the body reference frame velocities are also near constant which requires us to use fewer basis centers to cover the range of expected vehicle velocities in the numerical illustration below. Thus I can express the AUV dynamics in the required form

$$\dot{x} = Ax + B\big(\delta + f(x, \nu_1)\big) \tag{3.27}$$

where $A \in \mathbb{R}^{6\times 6}$ and $B \in \mathbb{R}^{6\times 3}$ are constant matrices and B is based on the vehicle moments of inertia and control surface effectiveness coefficients. The control surface effectiveness coefficients are estimated via hydrodynamic modeling discussed in [30]. For the purposes of this experiment, I neglect measurement error in the vehicle moments of inertia which results in a known B matrix. Additionally, the vehicle body reference frame linear velocities are included in the input space of the functional uncertainty, $f : \mathbb{R}^9 \to \mathbb{R}^3$.

Chapter 4

Gaussian Process MRAC

In this chapter I describe the application of the Gaussian Process Model Reference Adaptive Control (GP-MRAC) algorithm of [8],[27] to a tail-controlled AUV, specifically the VT-690 described in Chapter 2. GP-MRAC leverages the tools of Gaussian Process Regression (GPR) to estimate an unknown, non-linear function present in the system dynamics. GPR is one of many Bayesian Learning methods, which have become increasingly popular in recent years with the increased interest in statistical machine learning. GPR has several desirable properties when compared to other machine learning methods. Namely, it rigorously quantifies model uncertainty at test points, and the underlying statistical inference can be readily analyzed using the tools of Bayesian statistics [34]. GPR seeks to estimate an unknown function in a stochastic setting. The unknown function, $h(x)$, is assumed to be sampled from a GP which is described by a probability distribution over a (typically infinite) set of functions. This is noted as

$$h(x) \sim \mathcal{GP}(m(x), k(x, x')) \tag{4.1}$$

where $m(x)$ is the mean function and $k(x, x')$ is the covariance function. A GP is completely described by its mean and covariance function. The goal of GPR therefore is to learn the mean and covariance function. GPR is a supervised machine learning method in that the algorithm requires training data points with (potentially noisy) labels to perform the desired estimation [34].

4.1 Problem Description

Following the algorithm developed by [8] and [27], this section considers systems with dynamics of the form

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= g(x) + b(x)\delta(t) \end{aligned} \tag{4.2}$$

where $x_1 \in \mathbb{R}^n$, $x_2 \in \mathbb{R}^n$, $\delta(t) \in \mathbb{R}^n$ and $g : \mathbb{R}^{2n} \to \mathbb{R}^n$ and $b : \mathbb{R}^{2n} \to \mathbb{R}^{n \times n}$ are unknown, potentially nonlinear, functions.

Following [8] the control law is formulated by computing a desired acceleration which consists of a feed-forward term based on the reference model acceleration, a feed-back term based on the state error, and an adaptive term based on GPR:

$$\rho = \rho_{ff} + \rho_{fb} - \rho_{gp} \tag{4.3}$$

$$\rho_{gp} \sim \mathcal{GP}(m(x), \Sigma(x, x')). \tag{4.4}$$

The ideal control law

$$\delta(t) = b^{-1}(x)(\rho - g(x)). \tag{4.5}$$

is given by [27]. Because the functions $b(x)$ and $g(x)$ are unknown, estimates of each function are required resulting in the practical control law

$$\delta(t) = \hat{b}^{-1}(x)(\rho - \hat{g}(x)). \tag{4.6}$$

From (4.6), the estimated function $\hat{b}(x)$ must be invertible over the domain of interest $D_x \subset \mathbb{R}^{2n}$. The purpose of the adaptive element, ρ_{gp} is the cancellation of error between the the approximate functions $\hat{b}$ and $\hat{g}$ and the true functions b and g.

4.2 Background

In this section, I summarize the relevant aspects of the GP-MRAC algorithm developed by [8] and [27]. The desired acceleration of (4.3) is generated using (4.4) for the adaptive element, the feed-forward and feedback terms are given by [27]

$$\rho_{ff} = \dot{x}_{r_2} \tag{4.7}$$

$$\rho_{fb} = -\begin{bmatrix} K_p & K_d \end{bmatrix} e(t) = -Ke(t) \tag{4.8}$$

where $K \in \mathbb{R}^{m \times n}$. Substituting (4.3) into (4.6) and (4.2) results in the following error dynamics

$$\dot{e}(t) = \underbrace{\begin{bmatrix} 0_{m\times m} & I_{m\times m} \\ -K_p & -K_d \end{bmatrix}}_{A} e(t) + \begin{bmatrix} 0_{m\times m} \\ I_{m\times m} \end{bmatrix} (\tilde{g}(x) + \tilde{b}(x)\delta(t) - \rho_{gp}) \tag{4.9}$$

where $\tilde{g}(x) = g(x) - \hat{g}(x)$ and $\tilde{b}(x) = b(x) - \hat{b}(x)$. K is selected such that the A-matrix, as shown in (4.9), is Hurwitz. If the adaptive element can be made to cancel the error between the true functions and the estimated functions (i.e. $\rho_{gp}(x, \delta) = \tilde{g}(x) + \tilde{b}(x)\delta(t) \; \forall \; x \in D_x, \; \delta \in D_\delta$) then the system becomes exponentially stable [27]. To accomplish this, Gaussian Process Regression (GPR) is employed, see Chapter 2 of [34] for further details on GPR.

Let $D_z \subset \mathbb{R}^k$ be the domain of an unknown, scalar-valued function $h : \mathbb{R}^k \to \mathbb{R}$. Additionally

let $Z = \{z_1, \cdots, z_i, \cdots, z_N\}$ be a set of input observations for which $Y = \{y_1 \cdots, y_i, \cdots y_N\}$ are corresponding output observations of the unknown function $h(z)$. The output observations are assumed to be corrupted by independent identically distributed (i.i.d) Gaussian random noise

$$y_i = h(z_i) + \gamma, \;\; \gamma \sim \mathcal{N}(0, \sigma_n^2). \tag{4.10}$$

In order to perform gaussian process regression, a kernel function is selected which defines the covariance of the GP. For the purposes of this work I continue to use the scalar matern-5/2 kernel, $k_{5/2}(z_1, z_2)$, from 3.3. The process of model selection for gaussian process regression includes the choice of kernel function, and the hyper-parameters which characterize it. Model selection represents its own area of study which is well-summarized in Chapter 5 of [34]. The GP is trained using the observed input and output data, Z and the Y. In the bayesian setting, any information about the noise in the output data is encoded in the prior distribution given by equation 2.21 of [34]

$$\begin{bmatrix} y_1 \\ \vdots \\ y_N \\ h^* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K(Z,Z) + \sigma_n^2 I & K(Z, z^*) \\ K(Z, z^*)^\intercal & k(z^*, z^*) \end{bmatrix}\right) \tag{4.11}$$

where z^* is the test input, $K(Z, z^*)$ is the column vector whose i^{th} entry is the covariance function, $k_{5/2}(z_i, z^*)$, and $K(Z, Z)$ is the matrix with $k_{5/2}(z_i, z_j)$ at the i^{th} row and j^{th} column. From the prior distribution and the training data and associated observations a posterior distribution is developed. The posterior probability distribution characterizes both the expected value of the unknown function at a given test point and the model's uncertainty. From equation 2.25 and 2.26 of [34] these values for a single test point are given

by

$$\bar{h}(z^*) = \mathbb{E}[h(z^*) \mid Z, Y, z^*] = K(Z, z^*)^\mathsf{T}(K(Z, Z) + \sigma_n^2 I)^{-1} y \tag{4.12}$$

$$\text{var}[h(z^*)] = k(z^*, z^*) - K(Z, z^*)^\mathsf{T}(K(Z, Z) + \sigma_n^2 I)^{-1} k(Z, z^*) \tag{4.13}$$

where $y = \begin{bmatrix} y_1 & \cdots & y_i & \cdots & y_N \end{bmatrix}^\mathsf{T}$ is the vector of training observations.

Using the algorithm developed by Csato and Opper in [10], a GP can be learned in real-time (on-line) as the control algorithm executes. Loosely speaking, the GPR algorithm of [10] evaluates each new input data point using a measure of linear independence from the current library of basis centers and adds it to the library if it is sufficiently independent. If the library exceeds the user-defined budget size then the basis centers are evaluated for relevance and the point with the lowest score is removed and the GP's associated parameters are updated accordingly.

Under the assumptions of the GP-MRAC algorithm of [8] and [27], the closed loop system is mean-squared uniformly ultimately bounded. Stability results and proofs for the GP-MRAC algorithm are given by [9].

4.3 Practical Implementation

In order to implement GP-MRAC, estimates of the unknown functions, denoted $\hat{f}(x)$ and $\hat{b}(x)$, are required. These are generated using the 6-DOF model described in [13] and developed for the 690 AUV using the methods of [30]. Additionally, the estimated function $\hat{b}(x)$ must be invertible for all x in the domain of interest, $D_x \subset \mathbb{R}^{2n}$. These are constructed

using (2.7) and (2.12) as follows

$$\hat{g}(x) = -M_{\eta 22}^{-1}\Big(M_{\eta 21}\ddot{\eta}_1 + C_{\eta 21}\dot{\eta}_1 + C_{\eta 22}x_2 + D_{\eta 21}\dot{\eta}_1 + D_{\eta 22}x_2 + g_2(x_1)\Big) \tag{4.14}$$

$$\hat{b}(x)\delta(t) = M_{\eta 22}^{-1}\tau_2 = u^2 J_2(x_1) M_{22}^{-1}\Lambda\delta(t)$$

$$\hat{b}(x) = u^2 J_2(x_1) M_{22}^{-1}\Lambda. \tag{4.15}$$

The function $\hat{b}(x)$ has the inverse

$$\hat{b}^{-1}(x) = \frac{1}{u^2}\Lambda^{-1} M_{22} J_2^{-1}(x_1). \tag{4.16}$$

From (2.12), Λ is a diagonal matrix with non-zero entries on the diagonal, and thus it is non-singular by construction. Therefore (4.16) is well defined under two conditions. Firstly, surge, u must be non-zero. Physically this condition is equivalent to a minimum steerageway requirement for the vehicle attitude to be controllable using the control surfaces. This condition is enforced in the control algorithm by setting control surfaces to a default angle if vehicle surge drops below the threshold for minimum steerageway. Secondly, the rotation matrix $J_2(x_1)$ must be non-singular. From (2.10) it is clear that $J_2(x_1)$ is non-singular over the set $\{x_1 \in \mathbb{R}^3 : \ -\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2}\}$. This condition amounts to the example of gimbal lock referenced in Chapter 2.

As shown in (4.14) and (4.15), both the estimates and the true functions are dependent upon the linear accelerations and velocities in addition to the attitude and rotational velocities. The VT-690 measures both linear velocity and linear acceleration using an Inertial Measurement Unit (IMU) and these signals are treated as an exogenous input to the control algorithm.

The standard GP regression problem assumes that the GP has a scalar-valued output. The

simplest way to use this method for attitude control requires three separate GPs one for each attitude variable: roll, pitch, and yaw.

The inputs to each GPR algorithm are training data points consisting of $(\eta_2, \dot{\eta}, \ddot{\eta}_1, \delta)$ where η, η_1, and η_2, are defined in Section 2.1 and the output observations are given by

$$y_i = \dot{x}_2(t_i) - \rho(t_i) = \tilde{g}(x(t_i)) + \tilde{b}(x(t_i))\delta(t_i) \tag{4.17}$$

The angular acceleration of the vehicle is not directly measured, but is estimated from angular velocity data using a non-causal Savitsky-Golay filtering process [36] to reduce estimation noise. While the estimation requirement is undesirable, the non-causality of this method does not significantly impact the performance of the GP-MRAC algorithm as the GP is simply trained on data several time steps in the past.

Chapter 5

Results

5.1 Simulation Scenario

For each of the two controllers described in chapters 3 and 4, the first experiment simulates an AUV traveling at a constant speed of 1.5 m/sec, performing repeated step changes in pitch and yaw. The pitch commands alternate between ± 5 deg and the yaw commands alternate between ± 10 deg as shown by the dotted lines in the figures below. The second experiment repeats the steps of the first simulation but add an instantaneous change in vehicle buoyancy from 1% positively buoyant to 1% negatively buoyant at $t = 650$ sec.

The AUV is modeled using the 6-DOF dynamics of [30]. The GP-MRAC also makes use of the 6-DOF AUV dynamic model to create the estimated functions required for this algorithm (see section 4.3). Therefore, in the GP-MRAC simulation (section 5.3) and the comparative simulations (section 6.2), the simulated AUV model has uncertainty injected into each of the hydrodynamic coefficients. Each coefficient is modified by a random multiplier sampled from a uniform distribution between $0.75 - 1.25$ (i.e. 25 percent uncertainty in hydrodynamic coefficients)

5.2 RKHS MRAC Numerical Simulation Results

The controller is designed according to the RKHS MIMO MRAC formulation described in chapter 3.

I performed the first experiment multiple times with increasing numbers of kernel centers and recorded the average function error norm, $\|\tilde{f}(t)\|_2$ and the average state error norm, $\|e(t)\|_2$. Figure 5.1 shows the improvement in performance as kernel centers are added and demonstrates that the magnitude of performance improvement is additionally dependent on kernel location. Above 1458 kernel centers the computational intensity became prohibitive. Figures 5.2-5.4 show the simulation with 1458 kernel centers.

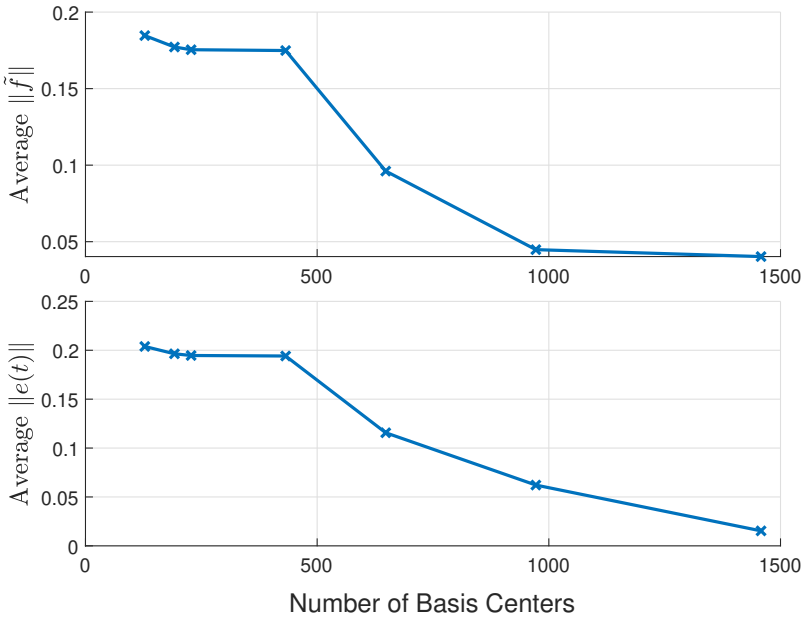

Figure 5.1: RKHS-MRAC Experiment 1: Comparison of error norm and functional error norm versus number of kernel centers

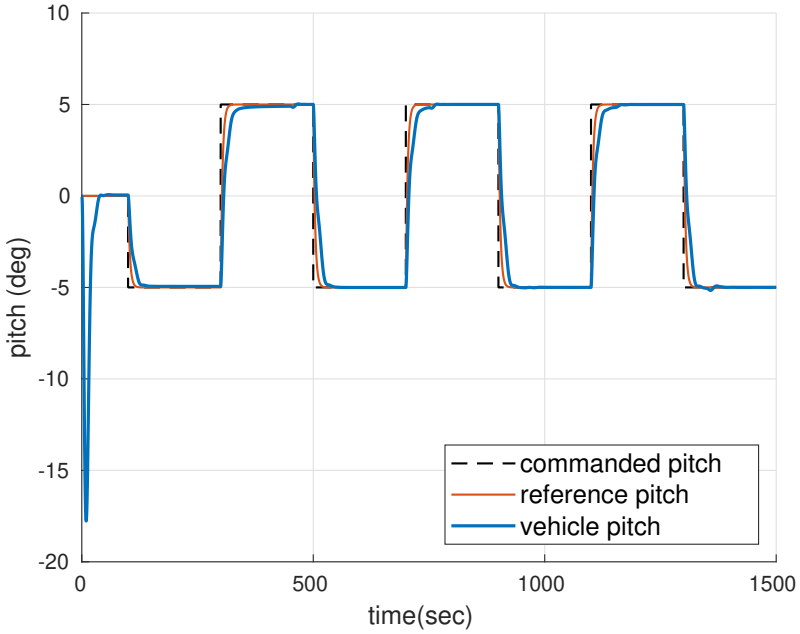

Figure 5.2: RKHS-MRAC Experiment 1: Vehicle Pitch

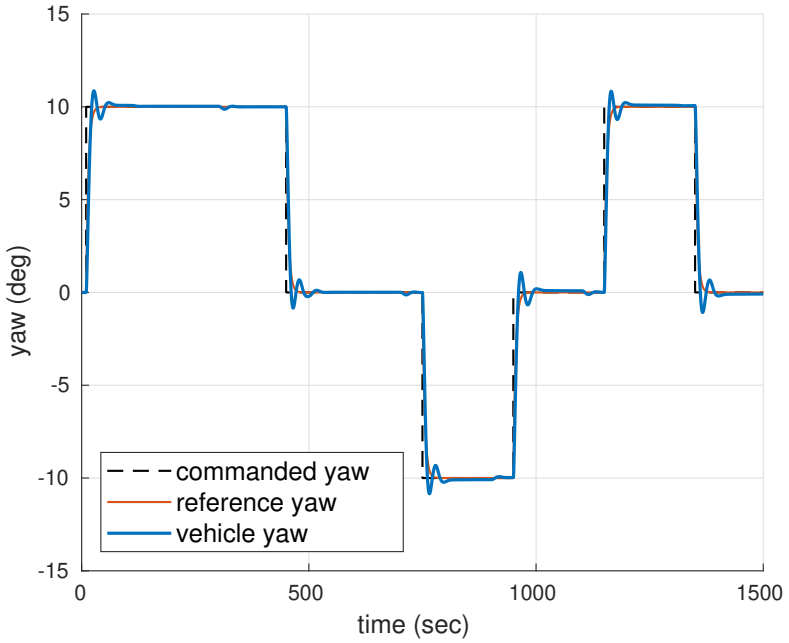

Figure 5.3: RKHS-MRAC Experiment 1: Vehicle Yaw

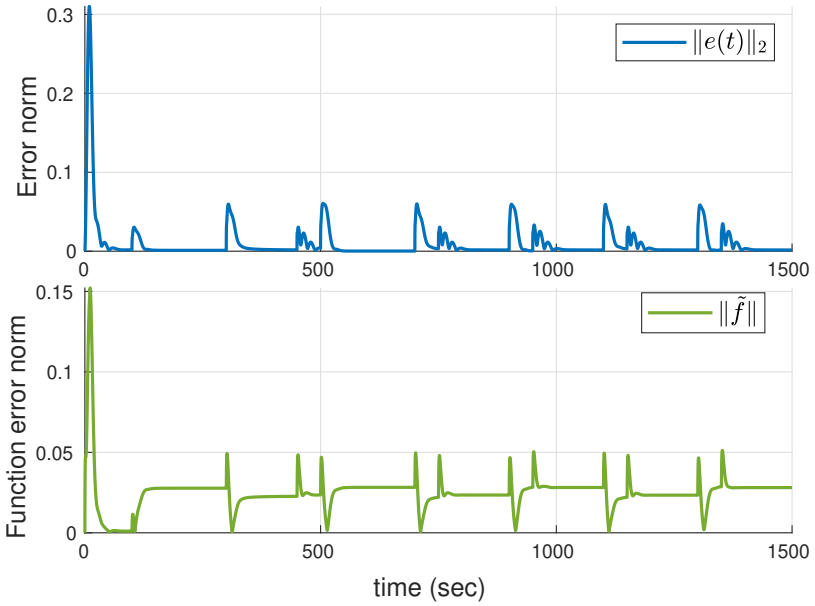

Figure 5.4: RKHS-MRAC Experiment 1: Norm of State Error and Function Error

In the second experiment I repeat the steps of the first simulation but add an instantaneous change in vehicle buoyancy from 1% positively buoyant to 1% negatively buoyant at $t = 650$ sec. Figures 5.5-5.7 show the attitude results of experiment 2.

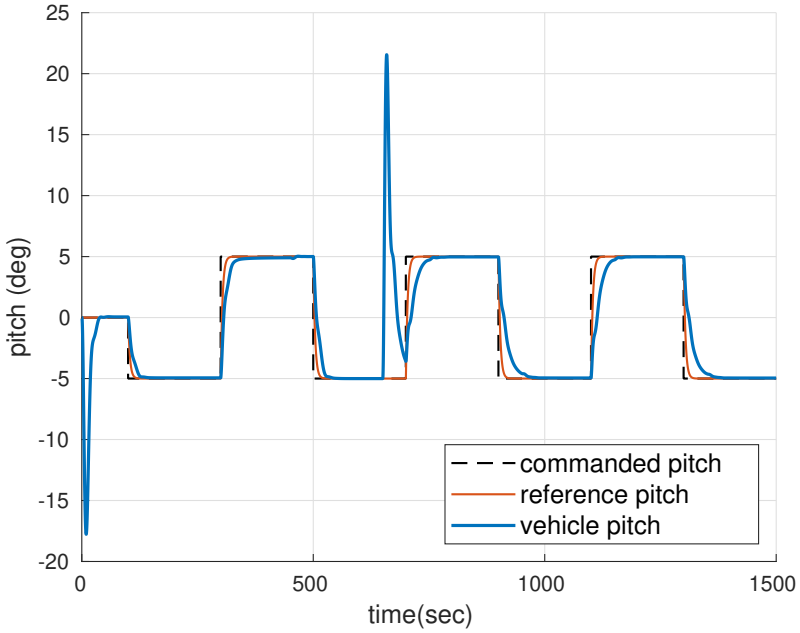

Figure 5.5: RKHS-MRAC Experiment 2: Vehicle Pitch

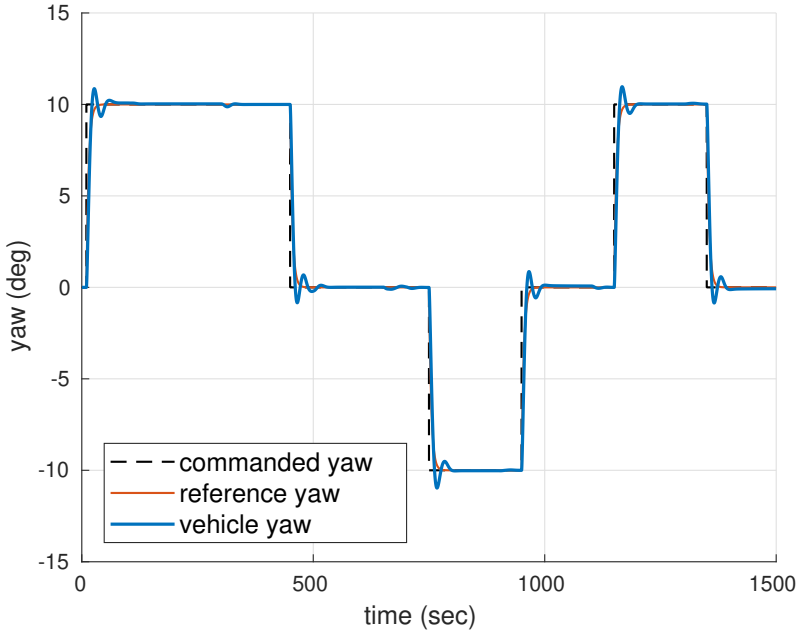

Figure 5.6: RKHS-MRAC Experiment 2: Vehicle Yaw

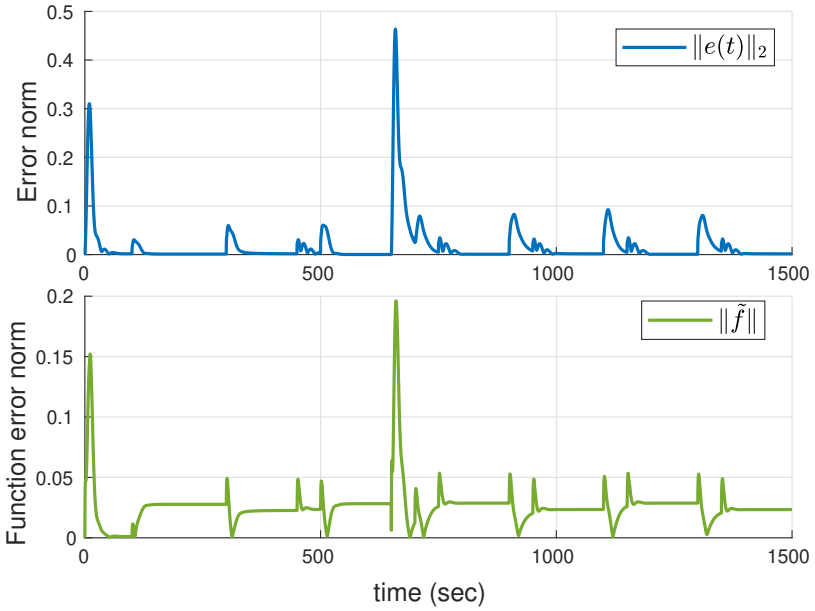

Figure 5.7: RKHS-MRAC Experiment 2: Norm of State Error and Function Error

5.3 GP-MRAC Numerical Simulation Results

The controller is designed according to the GP-MRAC formulation of [8] and described in chapter 4. Figures 5.2-5.4 show the simulation with a budget of 1000 kernel centers.

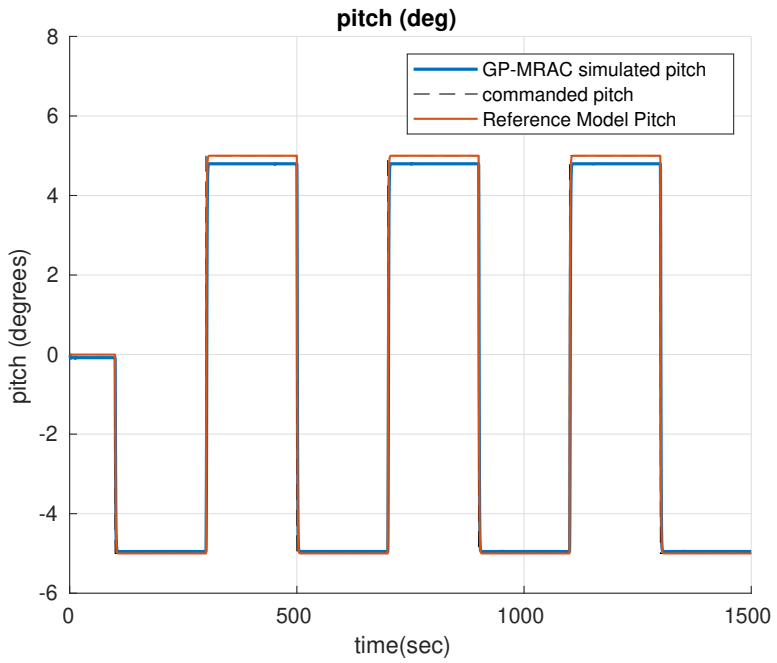

Figure 5.8: GP-MRAC Experiment 1: Vehicle Pitch

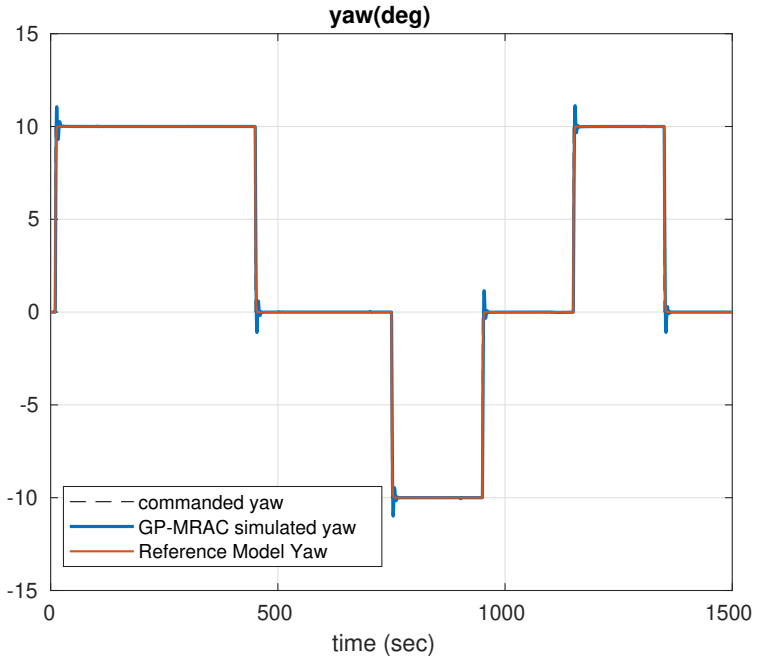

Figure 5.9: GP-MRAC Experiment 1: Vehicle Yaw

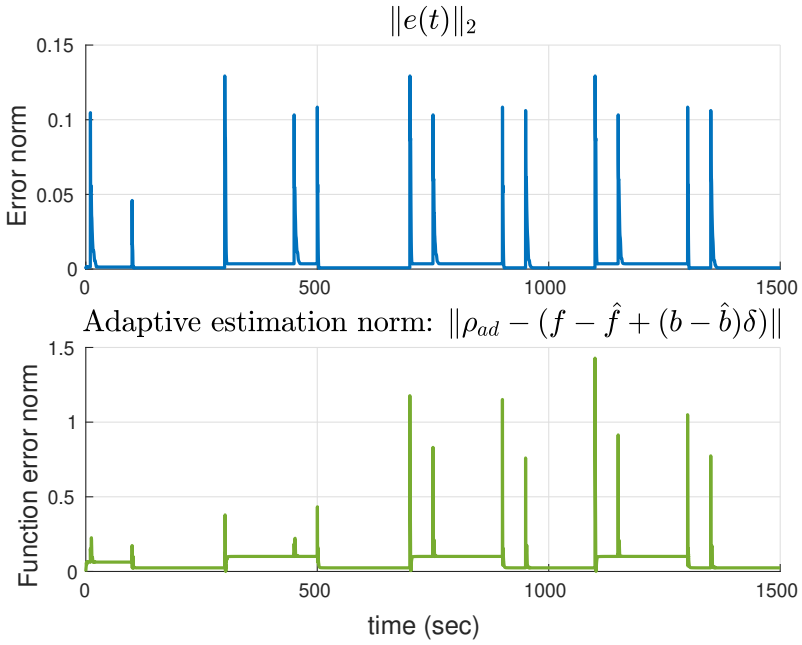

Figure 5.10: GP-MRAC Experiment 1: Norm of State Error and Function Error

In the second experiment I repeat the steps of the first simulation but add an instantaneous change in vehicle buoyancy from 1% positively buoyant to 1% negatively buoyant at $t = 650$ sec. Figures 5.11-5.13 show the attitude results of experiment 2.

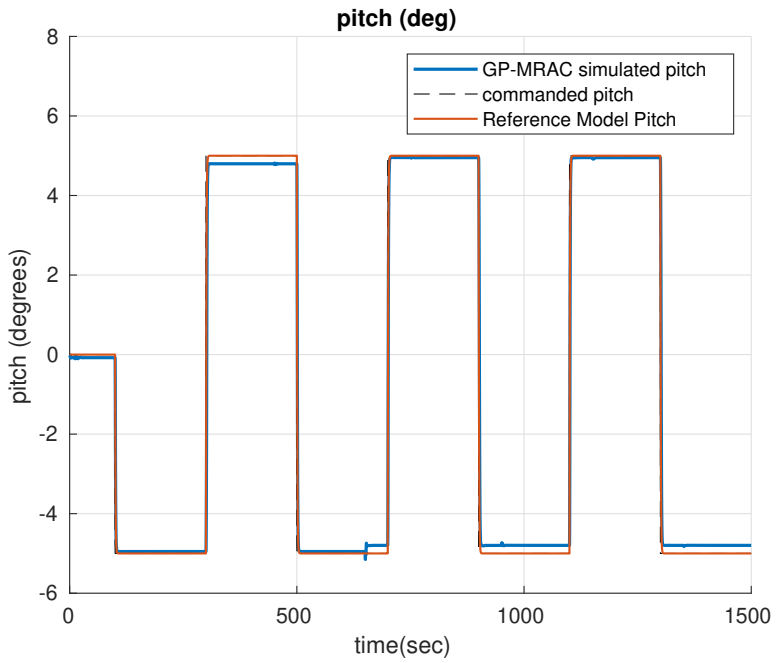

Figure 5.11: GP-MRAC Experiment 2: Vehicle Pitch

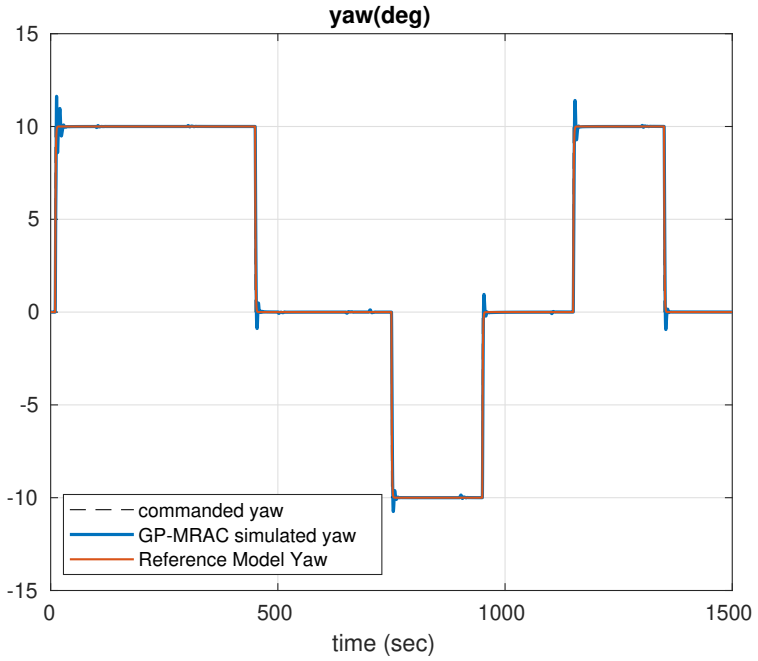

Figure 5.12: GP-MRAC Experiment 2: Vehicle Yaw

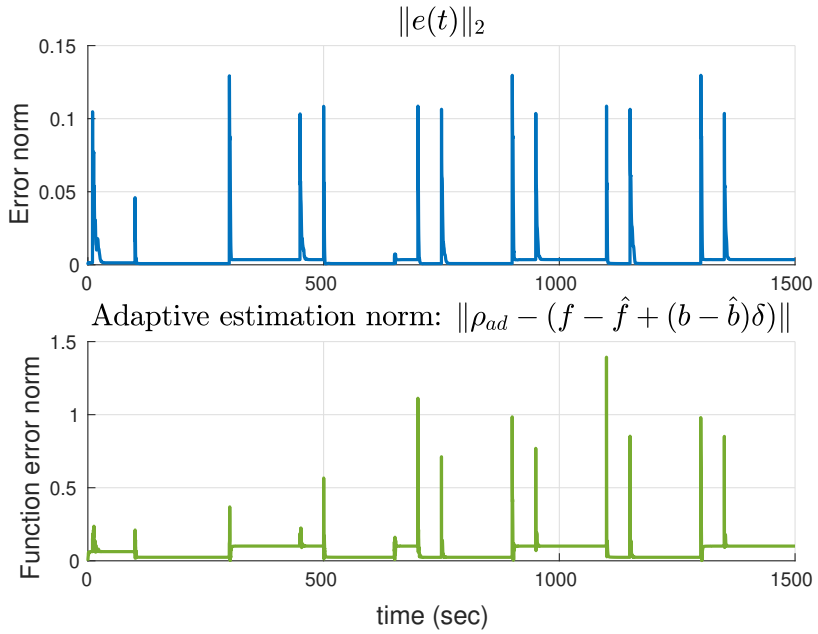

Figure 5.13: GP-MRAC Experiment 2: Norm of State Error and Function Error

5.4 RKHS MRAC Field Trial Results

5.4.1 Initial Field Trial Results

The initial field trials of the vehicle consist of performing a circular maneuver on the surface for the purpose of navigation calibration, a dive maneuver to a depth of 1m, and a constant yaw command at a depth of 5m for 360 sec. The constant yaw command was performed without an outer-loop position controller. For simplicity, only the maneuvers after the circular maneuver are shown in the following figures.

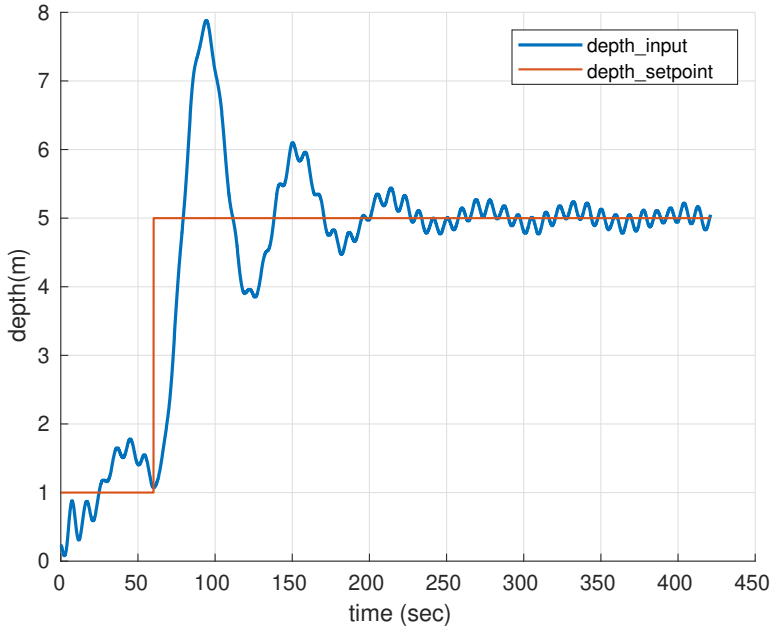

Figure 5.14: Vehicle depth during initial field trials

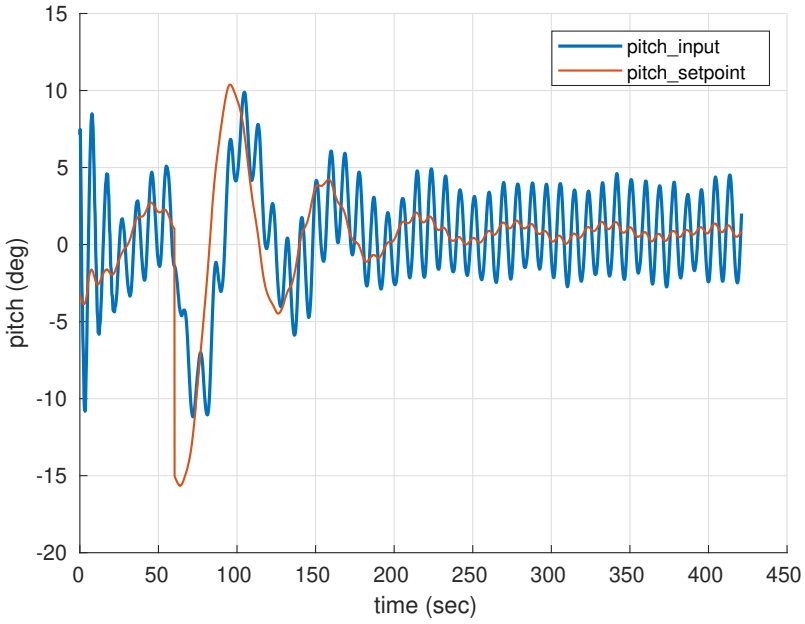

Figure 5.15: Vehicle pitch during initial field trials

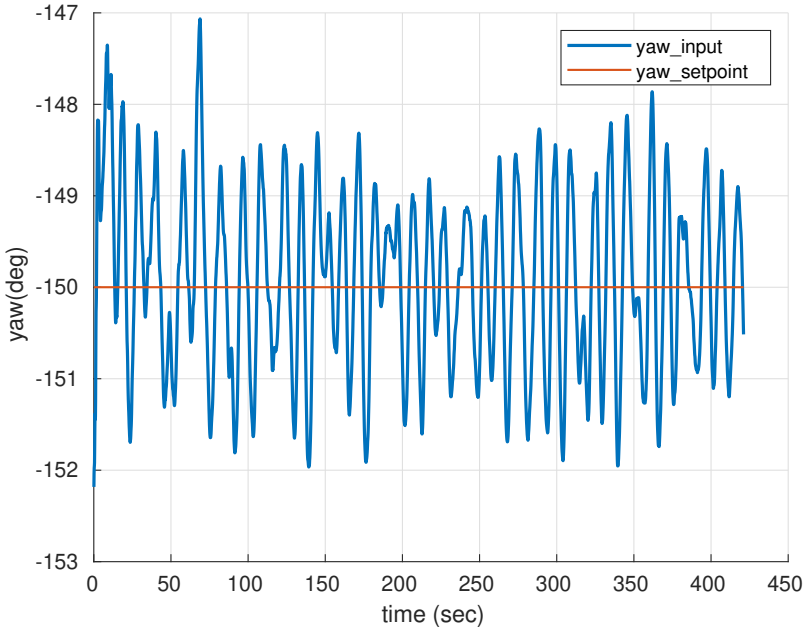

Figure 5.16: Vehicle yaw during initial field trials

The full closed loop system performed adequately though with more oscillation than desired during steady-state. In steady-state level flight the depth error was ±25 cm. The inner loop attitude controller exhibited significant, undesired oscillations specifically in the pitch axis. Pitch error during steady-state level flight was ±3.5 deg and yaw error was ±2 deg.

5.4.2 Modifications

ADP-PI

Building on the work in [21], an adaptive augmented PI (ADP-PI) controller is implemented. The algorithm of [21] is modified to apply to a multi-input, multi-output control system and the adaptive element used is that described in chapter 3. For this algorithm I consider

dynamics of the form

$$\dot{x}_1 = x_2 \tag{5.1}$$
$$\dot{x}_2 = A_{21}x_1 + A_{22}x_2 + B(\delta(t) + f(x))$$

where A_{21} and A_{22} are unknown, square, constant matrices and B is a known, square, invertible matrix, and $f(x)$ is an unknown function. Alternatively if B is not invertible then A_{21} and A_{22} must be known. Let $r(t) \in \mathbb{R}^3$ be the attitude command signal and define three additional states as

$$x_I(t) = \int_0^t x_1(\tau) - r(\tau)d\tau. \tag{5.2}$$

Adding the additional states to the overall state dynamics $x_a(t) = \begin{bmatrix} x_1^\mathsf{T}(t) & x_2^\mathsf{T}(t) & x_I^\mathsf{T}(t) \end{bmatrix}^\mathsf{T}$ results in the following system

$$\dot{x}_a = \begin{bmatrix} 0_3 & I_3 & 0_3 \\ A_{21} & A_{22} & 0_3 \\ I_3 & 0_3 & 0_3 \end{bmatrix} x_a + \begin{bmatrix} 0_3 \\ B \\ 0_3 \end{bmatrix} (\delta(t) + f(x)) + \begin{bmatrix} 0_3 \\ 0_3 \\ -I_3 \end{bmatrix} r(t) \tag{5.3}$$

The control law is formulated using a linear component and an adaptive component

$$\delta(t) = \delta_{\text{lin}}(t) + \delta_{\text{ad}}(t) \tag{5.4}$$

The adaptive component is given by $\delta_{\text{ad}}(t) = -\hat{f}(x)$ where $\hat{f}$ is constructed in the same manner as in chapter 3. The linear component is given by

$$\delta_{\text{lin}}(t) = -K_p(x_1(t) - r(t)) - K_I x_I(t) = -K^{\mathsf{T}}\left(x_a(t) - \begin{bmatrix} r(t) \\ 0 \\ 0 \end{bmatrix}\right) \tag{5.5}$$

where $K_p, K_I \in \mathbb{R}^{3\times 3}$ are diagonal gain matrices and $K^{\mathsf{T}} = \begin{bmatrix} K_{p_1} & K_{p_2} & K_I \end{bmatrix} \in \mathbb{R}^{3\times 9}$. This results in the following closed-loop system

$$\dot{x}_a = \underbrace{\begin{bmatrix} 0_3 & I_3 & 0_3 \\ A_{21} - BK_{p_1} & A_{22} - BK_{p_2} & -BK_I \\ I_3 & 0_3 & 0_3 \end{bmatrix}}_{A_r} x_a + \underbrace{\begin{bmatrix} 0_3 \\ BK_{p_1} \\ I_3 \end{bmatrix}}_{B_r} r - \begin{bmatrix} 0_3 \\ B \\ 0_3 \end{bmatrix} (\tilde{f}(x)) \tag{5.6}$$

where $\tilde{f}(x) := \hat{f}_N(x) - f(x)$ as in section 3.2. Following the practices described in [21], the reference model is selected to be Hurwitz and to exhibit the desired transient behavior. From there the gains are tuned to achieve reference model tracking using simulation. The error dynamics for the system of equation (5.6) are given by

$$\dot{e} = \dot{x}_a - \dot{x}_r = A_r e(t) - \begin{bmatrix} 0_3 \\ B \\ 0_3 \end{bmatrix} \tilde{f}(x). \tag{5.7}$$

Let P be the solution to the algebraic Lyapunov equation $A_r P + P A_r^T + Q = 0$. Using the Lyapunov function:

$$v(e, \tilde{f}) = e^T P e + \gamma_f^{-1} \langle \tilde{f}, \tilde{f} \rangle_{\mathbb{H}} \tag{5.8}$$

stability and boundedness results similar to those in proposition 3.1 can be proven using the same general method.

Projection Operator

Following [26], the RKHS-MRAC controller was modified to implement a projection operator on the adaptive law to prevent adaptive parameter drift due to disturbance inputs. First let g be a convex function which maps the function estimation coefficient vector alpha (see section 3.3) to the real numbers, defined as in equation 11.57 of [26]:

$$g(\alpha) = \frac{(1+\epsilon_\alpha)\|\alpha\|^2 - \alpha_{\max}^2}{\epsilon_\alpha \alpha_{\max}^2} \tag{5.9}$$

The gradient of equation (5.9) with respect to α is given by

$$\nabla g(\alpha) = \frac{2(1+\epsilon_\alpha)}{\epsilon_\alpha \alpha_{\max}^2}\alpha. \tag{5.10}$$

The projection operator (see sections 11.4 and 11.5 of [26]) is designed such that if the vector of coefficients, α, at time $t = 0$ is in the sub-level set $\{\alpha : g(\alpha) \leq 0\}$ then it remains in the set $\{\alpha : g(\alpha) \leq 1\} = \{\alpha : \|\alpha\| \leq \alpha_{\max}\}$ for all time $t \geq 0$ [26]. The projection operator is given as (equation 11.37 of [26])

$$\text{Proj}(\alpha, y) = \begin{cases} y - \frac{\nabla g(\alpha)(\nabla g(\alpha))^\intercal}{\|\nabla g(\alpha)\|^2} y g(\alpha), & g(\alpha) > 0 \text{ and } \nabla g(\alpha)^\intercal y > 0 \\ y, & \text{otherwise} \end{cases}. \tag{5.11}$$

For the RKHS-MRAC controller the projection operator is used to modify equation (3.23) to

$$\dot{\hat{\alpha}} = \text{Proj}(\hat{\alpha}, \gamma_f G^{-1}\Phi(x)) \tag{5.12}$$

where G and $\Phi(x)$ are defined as in equation (3.23).

5.4.3 Subsequent field trial results

The field trials of the vehicle consist of performing a circular maneuvers on the surface for the purpose of navigation calibration, a dive maneuver to a depth of 3m, a constant yaw command at a depth of 3m for 150 sec, and a lawn-mower survey path at a depth of 6m. The constant yaw command at 3m depth was performed without an outer-loop position controller similar to initial field trials while the survey path was performed with an outer-loop position controller which set the yaw command. Again, only the maneuvers after the circular maneuvers are shown in the following figures.

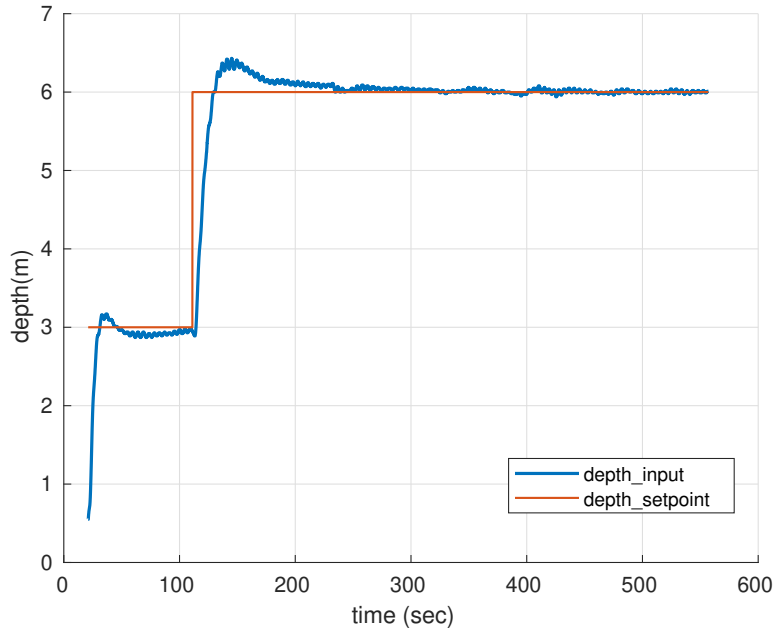

Figure 5.17: Vehicle depth during subsequent field trials

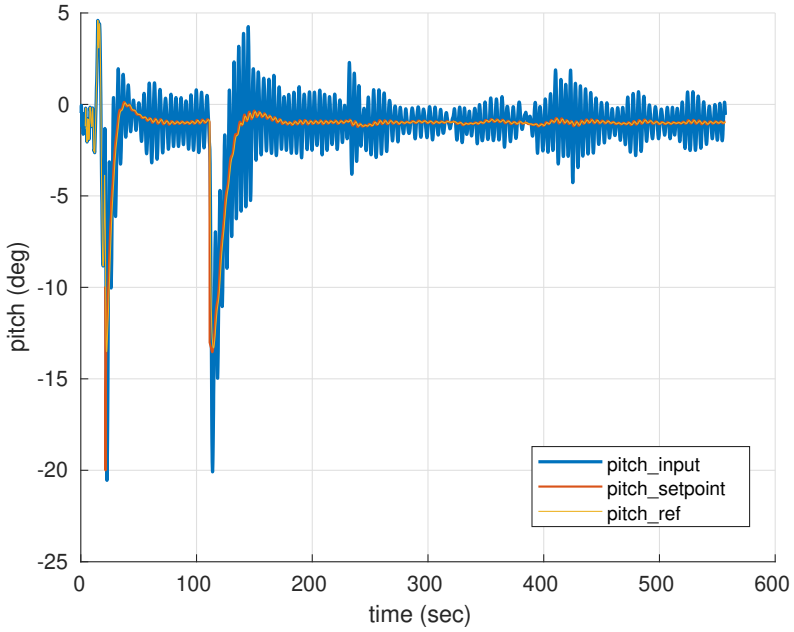

Figure 5.18: Vehicle pitch during subsequent field trials

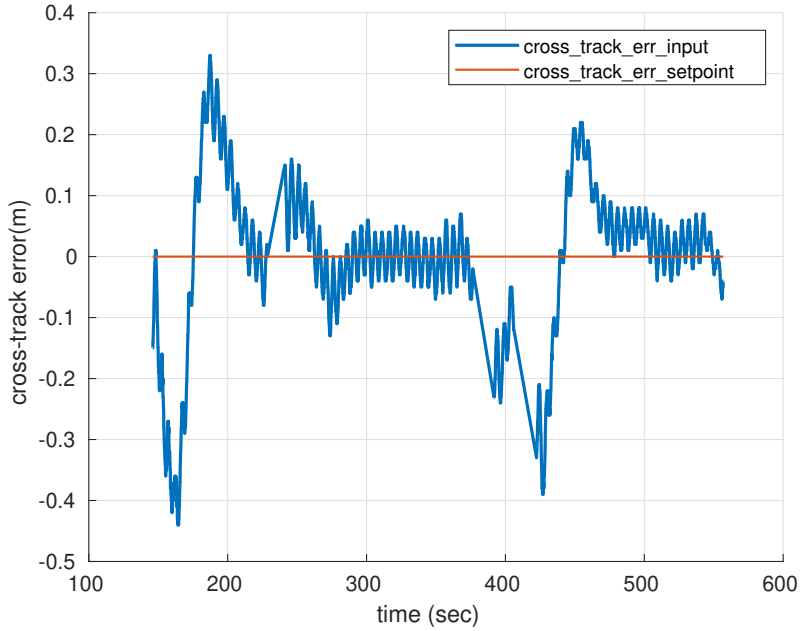

Figure 5.19: Vehicle cross track error during subsequent field trials

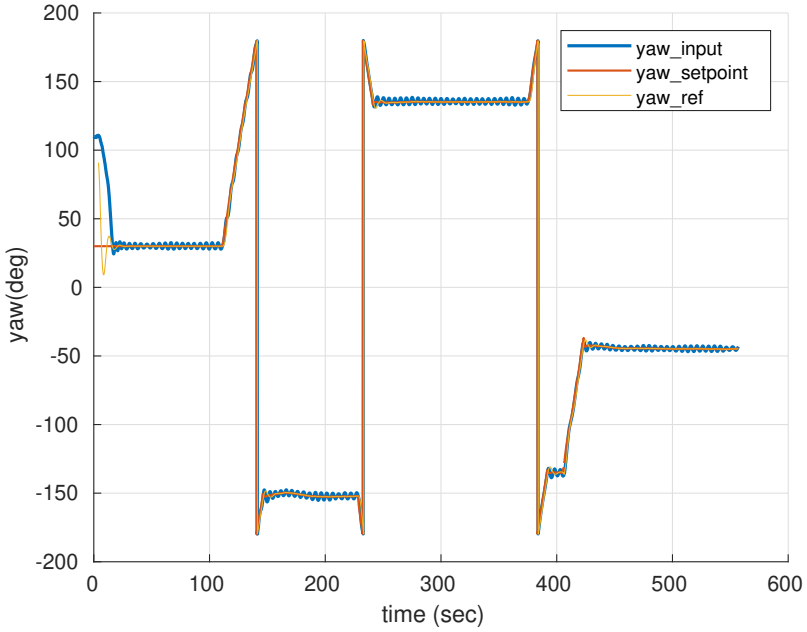

Figure 5.20: Vehicle yaw during subsequent field trials

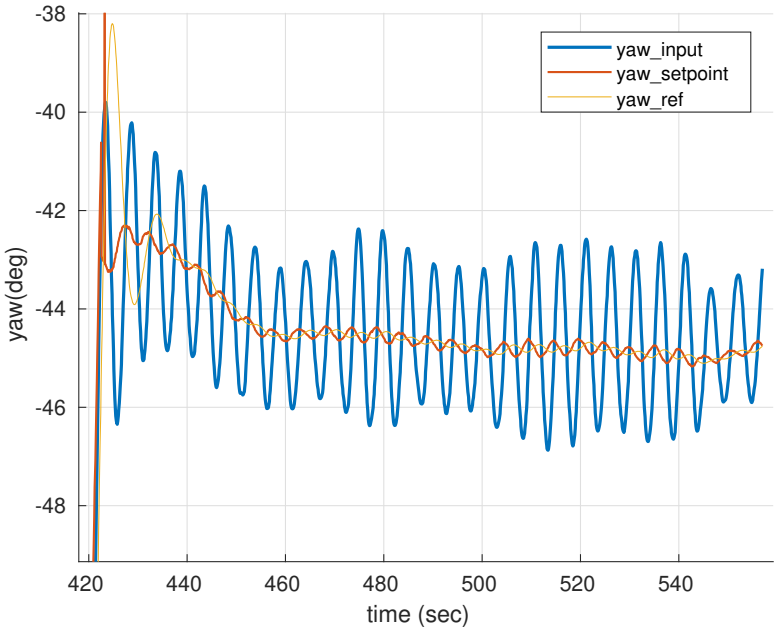

Figure 5.21: Vehicle yaw during final survey leg of subsequent field trials

In general, the full closed loop system performed reasonably well. In steady-state level flight the depth error was less than ±5 cm which is a significant improvement over initial field trials. The steady-state position cross track error, which was not measured during the initial field trials, was less than ±10 cm. The inner loop attitude controller continued to exhibit undesired oscillations although of lower amplitude than those present during initial field trials. Pitch error during steady-state level flight was ±2 deg and yaw error was similar to intial field trials when executed open loop at ±2 deg, during the survey with an outer-loop position controller yaw oscillations were slightly worse at ±2.5 deg.

Chapter 6

Algorithm Comparison

6.1 Comparison of algorithm structure

The theoretical basis for Gaussian Process Regression closely relates to that of functional estimation in an RKHS framework. This relationship is thoroughly treated in chapter 12 of [32], Chapter 6 of [34] and [24] to name a few. The remainder of this section summarizes the similarities as they pertain to the controllers described in Chapters 3 and 4.

In the GP-MRAC algorithm it is assumed that the unknown function is drawn from a Gaussian process. Let Ω be the sample space of possible functions. Consistent with chapter 4 let $h(z)$ denote the unknown function with $z \in \mathcal{Z} \subset \mathbb{R}^d$. To say that $h(z) \sim GP$ is to say that $h : \Omega \times \mathcal{Z} \to \mathbb{R}$ and thus h is a random function which follows a Gaussian process. By definition, a GP is completely defined by its mean function and its covariance function and any finite subset of the range of h will exhibit a multi-variate Gaussian distribution [34]

$$\begin{bmatrix} h(z_1) & \cdots & h(z_N) \end{bmatrix}^{\mathsf{T}} \sim \mathcal{N}(m, \Sigma). \tag{6.1}$$

As described in 4.2, GP regression relies on bayesian inference of a posterior probability distribution from a prior distribution and a set of input, output observations. Let $Z = \{z_1, \cdots, z_N\}$ denote the set of input points associated with the output observations which are assumed to be corrupted by independent identically distributed (i.i.d) Gaussian random

noise in (4.10). Under this assumption and a Gaussian prior, the posterior distribution $p(h \mid Z, y)$ is Gaussian

$$p(h \mid Z, y) \propto \underbrace{p(y \mid h, Z)}_{\text{likelihood}} \underbrace{p(h)}_{\text{prior}} \tag{6.2}$$

For the GP-MRAC algorithm the function estimation is performed by taking the expectation of the posterior distribution in equation (6.2) evaluated at the test point. From section 2.2 of [34] the posterior mean function can be expressed in the form

$$\bar{h}(z^*) = \sum_{i=1}^{N} \alpha_i k(z_i, z^*) \tag{6.3}$$

where $k(z_i, z^*)$ is the kernel function, in this case the matern-5/2 kernel. Using equation (3.4), the posterior mean function clearly lies in the finite dimensional subspace of a RKHS of scalar valued ($m = 1$) functions defined by

$$\bar{h} \in \mathcal{H}_N = \text{span}\{k_{5/2}(z_i, \cdot) \mid z_i \in Z,\ 1 \leq i \leq N\} \tag{6.4}$$

which is itself an RKHS (see Lemma 2.5 of [39]). Although the posterior mean function is in an RKHS, the sample functions from the GP are not necessarily in the RKHS. Further details can be found in Chapter 6 of [34], Chapter 1 of [38], and [23].

When performing GPR using the algorithm of Csató and Opper [10], the posterior GP is built iteratively with a series of single observation Bayesian updates up to a user-defined maximum number of observations. Each time a new observation is added, a new finite-dimensional subspace of an RKHS is constructed. The dimension of these finite-dimensional subspaces increases monotonically, that is $\mathcal{H}_1 \subset \mathcal{H}_2 \subset \cdots \subset \mathcal{H}_N \subset \mathcal{H}$.

In the implementation of GP-MRAC for AUV attitude control, three separate GP's are

trained, one for each rotational axis (roll, pitch, and yaw). The finite dimensional RKHS subspaces induced by each of these posterior mean functions can be equivalently represented as a vector valued RKHS subspace which is the cartesian product of the three scalar valued RKHS subspaces

$$\mathbb{H}_D = \mathcal{H}_M^{(1)} \times \mathcal{H}_N^{(2)} \times \mathcal{H}_P^{(3)} \tag{6.5}$$

where the dimension of the product space is $D \leq M + N + P$ depending on the individual scalar kernel functions and the number of basis points which are shared between the scalar-valued RKHS subspaces. This type of construction limits the type of vector-valued native spaces which are generated compared to those generated by the more general operator-valued kernel function used in the RKHS-MRAC algorithm. The obvious solution is multi-output Gaussian Process regression, but this introduces additional computational intensity and an increasingly complex model selection problem. Several methods of performing multi-output GP regression are discussed in [5] and [4].

The key architectural differences are summarized by

1. Stochastic versus deterministic setting:

   The GP-MRAC method includes the ability to compute the posterior variance at a given test point, which rigorously quantifies the uncertainty associated with the model constructed by the GP. This quantitative measure of uncertainty associated with the prediction has no analog in the RKHS-MRAC framework as this framework performs functional regression in a deterministic setting.

2. The need to measure or estimate angular acceleration:

   The RKHS-MRAC method uses deterministic Lyapunov stability analysis to determine the update law for the estimated function while the GP-MRAC method, which

is a supervised learning algorithm, requires observations of the output of the unknown function. In practice acquiring these observations is often non-trivial and can introduce another significant source of noise into the control loop. In the AUV attitude control problem, this is accomplished by estimating angular acceleration which requires numerical differentiation of noisy angular rate measurements. [22] acknowledges the acceleration estimation problem in the GP-MRAC algorithm and proposes an alternative method for generating GP training labels using a fast-trained neural network.

3. Reliance on dynamic modeling:

   The performance bound of the GP-MRAC algorithm relies on the quality of the estimated functions in equation (4.14) and (4.15) (see theorem S1 of [8]), therefore achieving better performance could require more detailed and complex modeling of the system dynamics. Meanwhile only the control influence matrix of the system dynamics need be accurately modeled to support the RKHS-MRAC algorithm.

6.2 Simulation comparison

For each controller, I perform a similar experiment to that described in section 5.1. The AUV is modeled with 25 percent uncertainty in each of the hydrodynamic coefficients as in section 5.1 and the same model is used for each controller. The attitude commands alternate between $\pm 10\,\text{deg}$ for both pitch and yaw. As these controllers utilize slightly different architectures, the meaningful metrics for comparison are the norm of the error signal, $\|e(t)\|$ and the functional estimation error, $\|\tilde{f}\|$.

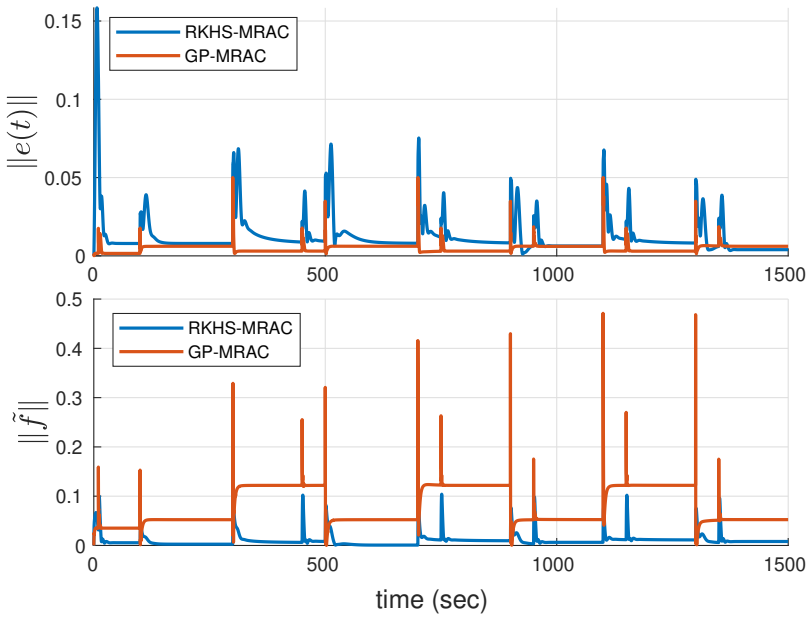

Figure 6.1: Comparison of MRAC algorithm error and functional error

As shown in figure 6.1, the GP-MRAC algorithm generally exhibits better performance in error norm, but the steady state error norm is comparable for each algorithm. The RKHS-MRAC algorithm performs better in functional estimation in this simulation.

Chapter 7

Conclusions and Summary

This paper demonstrates the modeling of vehicle dynamics in the inertial reference frame for the purpose of attitude control of autonomous underwater vehicles. A general MRAC controller based on vector-valued RKHS embedding methods is developed and implementation is presented in the context of attitude control of AUVs. The adaptive update laws yield ultimate boundedness of the reference signal tracking error under that the dead-zone modification. A benefit of non-parametric function estimation for adaptive control in an RKHS setting, rather than parametric function estimation that arises is conventional adaptive control, is that I am able to influence the estimation performance (e.g., error) by choice of kernel, location of kernel centers, and the number of kernel centers. For the attitude control example, it is explicitly demonstrated that increasing the number of kernels and kernel density yields reduced function estimation error.

Further I formulate the AUV attitude dynamics such that the GP-MRAC controller can be applied to the attitude control problem. I compare the two MRAC controllers in theory and on a simulated VT-690 AUV and demonstrate that the RKHS MRAC controller can achieve similar performance to the GP-MRAC controller without the need for the attitude acceleration estimates that are required for the GP-MRAC controller.

Finally, I demonstrate the performance of the RKHS MRAC controller on a VT-690 vehicle in field trials in both its originally developed form and with several modifications which demonstrably improved the control algorithm's performance.

Bibliography

[1] N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68:337–404, 1950.

[2] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[3] Parag Bobade, Suprotim Majumdar, Savio Pereira, Andrew J. Kurdila, and John B. Ferris. Adaptive estimation in reproducing kernel hilbert spaces. In *2017 American Control Conference (ACC)*, pages 5678–5683, 2017. doi: 10.23919/ACC.2017.7963839.

[4] Edwin V. Bonilla, Kian Ming Adam Chai, and Christopher K. I. Williams. Multi-task gaussian process prediction. In *NIPS*, 2007.

[5] Phillip Boyle and Marcus Frean. Dependent gaussian processes. In *NIPS*, 2004. URL https://api.semanticscholar.org/CorpusID:6491673.

[6] John A. Burns, Jia Guo, Andrew J. Kurdila, Sai Tej Paruchuri, and Haoran Wang. Error bounds for native space embedding observers with operator-valued kernels. *2023 American Control Conference (ACC), to appear*, 2023.

[7] John A. Burns, Andrew J. Kurdila, Derek Oesterheld, Daniel Stilwell, and Haoran Wang. Learning theory convergence rates for observers and controllers in native space embedding. *2023 American Control Conference (ACC), to appear*, 2023.

[8] Girish Chowdhary, Hassan A. Kingravi, Jonathan P. How, and Patricio A. Vela. A bayesian nonparametric approach to adaptive control using gaussian processes. In *52nd IEEE Conference on Decision and Control*, pages 874–879, 2013. doi: 10.1109/CDC.2013.6759992.

[9] Girish Chowdhary, Hassan A Kingravi, Jonathan P How, and Patricio A Vela. Bayesian nonparametric adaptive control using gaussian processes. *IEEE transactions on neural networks and learning systems*, 26(3):537–550, 2014.

[10] L. Csató and Manfred Opper. Sparse on-line gaussian processes. *Neural Computation*, 14:641–668, 2002.

[11] Rongxin Cui, Chenguang Yang, Yang Li, and Sanjay Sharma. Adaptive neural network control of auvs with control input nonlinearities using reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(6):1019–1029, 2017. doi: 10.1109/TSMC.2016.2645699.

[12] Jay Farrell and Marios Polycarpou. *Adaptive Approximation Based Control: Unifying Neural, Fuzzy and Traditional Adaptive Approximation Approaches*. John Wiley & Sons, 2006.

[13] Thor I. Fossen. *Guidance and Control of Ocean Vehicles*. John Wiley, 1999.

[14] Aditya Gahlawat, Pan Zhao, Andrew Patterson, Naira Hovakimyan, and Evangelos Theodorou. L1-gp: L1 adaptive control with bayesian learning. In *Learning for Dynamics and Control*, pages 826–837. PMLR, 2020.

[15] Tingran Gao, Shahar Z Kovalsky, and Ingrid Daubechies. Gaussian process landmarking on manifolds. *SIAM Journal on Mathematics of Data Science*, 1(1):208–236, 2019.

[16] J. Guo. *Convergence of kernel methods for modeling and estimation of dynamical systems*. PhD thesis, Virginia Tech, 2021.

[17] Jia Guo, Sai Tej Paruchuri, and Andrew J. Kurdila. Persistence of excitation in uniformly embedded reproducing kernel hilbert (rkh) spaces. In *2020 American Control Conference (ACC)*, pages 4539–4544, 2020. doi: 10.23919/ACC45564.2020.9147851.

[18] Jia Guo, Michael E. Kepler, Sai Tej Paruchuri, Hoaran Wang, Andrew J. Kurdila, and Daniel J. Stilwell. Adaptive estimation of external fields in reproducing kernel hilbert spaces. *International Journal of Adaptive Control and Signal Processing*, 36:1931–1957, 2022. doi: 10.1002/acs.3442. URL https://doi.org/10.1002/acs.3442.

[19] Jia Guo, Sai Tej Paruchuri, and Andrew J. Kurdila. Partial persistence of excitation in RKHS embedded adaptive estimation. *IEEE Transactions on Automatic Control*, pages 1–13, 2022. doi: 10.1109/TAC.2022.3226710. URL https://doi.org/10.1109/TAC.2022.3226710.

[20] Liwei Guo, Weidong Liu, Zeyu Li, and Linfeng Li. An adaptive sliding mode control strategy for the heading control of autonomous underwater vehicles. In *Global Oceans 2020: Singapore – U.S. Gulf Coast*, pages 1–4, 2020. doi: 10.1109/IEEECONF38699.2020.9389459.

[21] Paola Jaramillo Cienfuegos, Adam Shoemaker, Robert W. Grange, Nicole Abaid, and Alexander Leonessa. Classical and adaptive control of ex vivo skeletal muscle contractions using functional electrical stimulation (fes). *PLOS ONE*, 12(3):1–29, 03 2017. doi: 10.1371/journal.pone.0172761. URL https://doi.org/10.1371/journal.pone.0172761.

[22] Girish Joshi and Girish Chowdhary. Adaptive control using gaussian-process with model reference generative network. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 237–243, 2018. doi: 10.1109/CDC.2018.8619431.

[23] T. Kailath. Rkhs approach to detection and estimation problems–i: Deterministic signals in gaussian noise. *IEEE Transactions on Information Theory*, 17(5):530–549, 1971. doi: 10.1109/TIT.1971.1054673.

[24] Motonobu Kanagawa, Philipp Hennig, Dino Sejdinovic, and Bharath K Sriperumbudur. Gaussian processes and kernel methods: A review on connections and equivalences, 2018.

[25] Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9(2), 2008.

[26] E. Lavretsky and K. Wise. *Robust and Adaptive Control: With Aerospace Applications*. Advanced Textbooks in Control and Signal Processing. Springer London, 2012. ISBN 9781447143956. URL https://books.google.com/books?id=cRefvQEACAAJ.

[27] Miao Liu, Girish Chowdhary, Bruno Castra da Silva, Shih-Yuan Liu, and Jonathan P. How. Gaussian processes for learning and control: A tutorial with examples. *IEEE Control Systems Magazine*, 38(5):53–86, 2018. doi: 10.1109/MCS.2018.2851010.

[28] Charles A. Micchelli and Massimiliano Pontil. On learning vector-valued functions. *Neural Computation*, 17(1):177–204, 2005. doi: 10.1162/0899766052530802.

[29] Muhammad Mudassir and Attaullah Y. Memon. Steering control of an autonomous underwater vehicle using smc techniques. In *2020 3rd International Conference on Control and Robots (ICCR)*, pages 154–158, 2020. doi: 10.1109/ICCR51572.2020.9344394.

[30] Taylor Njaka, Stefano Brizzolara, and Daniel J Stilwell. Technical and research bulletin 1-45: Guide for CFD-informed AUV maneuvering models, Jan 2022.

[31] Sai Tej Paruchuri, Jia Guo, and Andrew J. Kurdila. Kernel center adaptation in the reproducing kernel hilbert space embedding method. *International Journal of Adaptive Control and Signal Processing*, 36:1562–1583, 2022. doi: 10.1002/acs.3407. URL https://doi.org/10.1002/acs.3407.

[32] Vern I. Paulsen and Mrinal Raghupathi. *An Introduction to the Theory of Reproducing Kernel Hilbert Spaces*. Cambridge University Press, 1 edition, Apr 2016. ISBN 978-1-107-10409-9. doi: 10.1017/CBO9781316219232. URL https://www.cambridge.org/core/product/identifier/9781316219232/type/book.

[33] Gianluigi Pillonetto, Francesco Dinuzzo, Tianshi Chen, Giuseppe De Nicolao, and Lennart Ljung. Kernel methods in system identification, machine learning and function estimation: A survey. *Automatica*, 50(3):657–682, 2014. ISSN 0005-1098. doi: https://doi.org/10.1016/j.automatica.2014.01.001. URL https://www.sciencedirect.com/science/article/pii/S000510981400020X.

[34] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 11 2005. ISBN 9780262256834. doi: 10.7551/mitpress/3206.001.0001. URL https://doi.org/10.7551/mitpress/3206.001.0001.

[35] G. Rigatos, G. Raffo, and P. Siano. Auv control and navigation with differential flatness theory and derivative-free nonlinear kalman filtering. *Intelligent Industrial Systems*, 3 (1):29–41, Feb 2017. doi: 10.1007/s40903-017-0068-y.

[36] Abraham. Savitzky and M. J. E. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry*, 36(8):1627–1639, 1964. doi: 10.1021/ac60214a047. URL https://doi.org/10.1021/ac60214a047.

[37] Adam J Thorpe, Cyrus Neary, Franck Djeumou, Meeko MK Oishi, and Ufuk Topcu. Physics-informed kernel embeddings: Integrating prior system knowledge with data-driven control. *arXiv preprint arXiv:2301.03565*, 2023.

[38] Grace Wahba. *Spline Models for Observational Data*. Society for Industrial and Applied Mathematics, 1990. doi: 10.1137/1.9781611970128. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611970128.

[39] Dominik Wittwar, Bernard Haasdonk, and Gabriele Santin. Interpolation with uncoupled separable matrix-valued kernels. *Dolomites Research Notes on Approximation*, 11(11/2018):23–39, 2018. ISSN 2035-6803. doi: 10.14658/pupj-drna-2018-3-4. URL https://doi.org/10.14658/pupj-drna-2018-3-4.

[40] Niankai Yang, Dongsik Chang, Matthew Johnson-Roberson, and Jing Sun. Energy-optimal control for autonomous underwater vehicles using economic model predictive control. *IEEE Transactions on Control Systems Technology*, 30(6):2377–2390, 2022. doi: 10.1109/TCST.2022.3143366.