Adaptive-Optimal Control of Nonstationary Dynamical Systems

Ali Abdollahi, Rakshit Allamraju and Girish Chowdhary

Abstract An adaptive-optimal control architecture is presented for adaptive control of constrained aerospace systems that are subject to dynamic change. The architecture brings together three key elements, model predictive control based reference command shaping, Gaussian Process (GP) based Bayesian nonparametric model reference adaptive control, and online GP clustering over nonstationary (time-varying) GPs. The key salient feature of our architecture is that not only can it detect changes, but it uses online GP clustering to enable the controller to utilize past learning of similar models to significantly reduce learning transients. Stability of the architecture is argued theoretically and performance is validated empirically.

1 Introduction

Control of aerospace vehicles that are subject to dynamic change is a challenging problem. Such change could happen due to faults, reconfiguration of the vehicle, or due to different operating environments. A widely employed method in aerospace controls is to first formulate a first-principles based model of the system, identify its parameters through system identification, and then optimize the control design utilizing these models. However, in presence of nonstationarity induced due to dynamic change, a single parameterized model may not be sufficient to describe the dynamics of the system. In fact, when the change is unexpected or sudden, such as due to a fault, a-priori models may not be applicable at all. One way to deal with change is to estimate the parameters of a model online, and then utilize online optimization techniques, such as Model Predictive Control (MPC) ([5, 24]) to learn new control policies. The attractiveness of utilizing online MPC is in MPC's ability to optimize over state and input constraints. These methods, often referred to as indi-

Ali Abdollahi and Rakshit Allamraju are Graduate Research Assistants, and Girish Chowdhary is an Assistant Professor at Oklahoma State University, Stillwater OK e-mail: ali.abdollahi, rakshit.allamraju,girish.chowdhary@okstate.edu

rect adaptive control mostly assume that the online generated estimates are equal to the real parameters (an assumption known as certainty equivalence [3]). Several authors have studied adaptive-MPC architectures that rely on variants of the certainty equivalence principle [4, 1, 15]. However, it is difficult to guarantee stability of such methods, especially during parameter estimation transients [21].

On the other hand, direct adaptive controllers such as Model Reference Adaptive Control (MRAC) can guarantee stability, even during harsh transients. These techniques have been successfully employed for flight vehicle control [18]. However, MRAC methods do not typically provide stability under state constraints. Adaptive control literature also consists of hybrid direct-indirect control architectures [19, 14, 9]. These methods seek to employ learning in a direct adaptive control framework. For example, in Chowdhary et al.'s concurrent learning method instantaneous data is used concurrently with specifically online recorded data to speed up the learning of unknown system parameters [11]. The learning capability of concurrent learning has been utilized with MPC architectures to yield adaptive optimal controllers [10, 22]. However, the main limitation of the concurrent learning approach, and in fact many other adaptive control and online system identification approaches, is that they utilize an a-priori fixed structure for modeling the unknown system dynamics, the parameters of which are learned online. On the other hand, the Gaussian Process MRAC (GP-MRAC) approach does not need to assume an a-priori model structure, rather, it utilizes online data to simultaneously infer both the structure and the parameters of the unknown system dynamics [27, 12].

The main challenge this paper seeks to address is of adaptive control in presence of unforeseen changes in the system dynamics. Our architecture is inspired by Muhlegg et al.'s concurrent learning adaptive-optimal control architecture [22] since it also relies on optimizing the reference model commands. This paper extends MPC based reference command shaping technique [22] to the Bayesian Nonparametric (BNP) adaptive control of Gaussian processes [8]. Another contribution of this paper is in showing that by optimizing the reference model commands using MPC, the system performance can be optimized without losing the desirable stability guarantees of GP-MRAC. An online clustering method is utilized to identify and leverage similarities between online learned GP models of the unknown dynamics. Therefore, the key salient feature of our architecture is that not only can it detect changes, but it also uses online GP clustering [16] to enable the controller to utilize past learning of similar models to significantly reduce learning transients.

1.1 Related Work

Model Predictive Control (MPC), also referred to as Receding Horizon Control and Moving Horizon Optimal Control, has been widely adopted in industry as an effective means to deal with multivariable constrained control problems [20, 25]. With advent in compact computing resources, authors have recently proposed and employed MPC architectures for real-time control of aerospace vehicles [6]. However,

the performance of MPC can depend on how the accuracy of the employed predictive model of the system dynamics is. Several authors have proposed learning based MPC algorithms [4, 1, 15], however the presence of learning transients prevents a general non-conservative solution to be formed. The CL-MPC architecture brings together learning based MRAC and MPC and utilizes an online threshold test to switch between the two [10]. One of the main challenges in implementing MPC based architectures is to guarantee computational feasibility of obtaining an optimal solution online on resource constrained aircraft. In Muhlegg et al.'s approach this problem is tackled by solving offline the optimal control problem for a linear reference model that the adaptive controller is guaranteed to track [22].

Most existing GP inference algorithms assume that the underlying generative model is stationary. Grande et al.'s Gaussian Process Non-Bayesian Clustering (GP-NBC) algorithm [16, 2] decouples the problem of changepoint detection, regression and model reuse, resulting in efficient online learning in the presence of changepoints. GP-NBC is highly computationally efficient and orders of magnitude faster than other GP clustering methods, such as Dirichlet Process based GP clustering [17]. We utilized GP-NBC in an adaptive-optimal control framework. We show that the resulting control architecture is guaranteed to be stable, and the likelihood of wrong clustering as well as the worst case tracking error due to misclassification are bounded.

2 Problem Formulation

We begin by describing the class of switching nonlinear dynamical systems our approach is applicable to. Let $x = [x_1 \ x_2]^\top \in D_x \subset \mathbb{R}^n$ be defined as the state vector of the nonlinear system where D_x is a subset of $\mathbb{R}^n$. Let $u(t) \in \mathbb{R}^m$ be an admissible control input to the system, then the class of dynamical systems considered here have the form:

$$\dot{x}(t) = Ax(t) + Bu(t) + B\Delta_i(x(t)) \tag{1}$$

where the state and input matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ are the linear known components, and $\Delta_i(x(t)) \in \mathbb{R}^m$ is the unknown, stochastic, and switching component in the system with the switching index i. The solution to (1) induces a nonstationary continuous time Markovian process. The process is nonstationary, i.e. it does not have a time invariant generating distribution, because the stochastic component Δ_i switches. We assume that the system (A,B) is fully controllable and the control input $u(t) \in \mathbb{R}^m$ is constrained due to limited control energy available to the system. It is further assumed that Δ_i is a stochastic process, and in particular a GP. Here we leverage the idea of modeling stochastic uncertainties $\Delta(x)$ in MRAC using Gaussian Processes [8]. That is,

$$\Delta_i(x) \sim \mathcal{GP}(m_i(x), K_i(x,x')), \tag{2}$$

where $m(\cdot)$ is the mean function of the GP and $K(\cdot,\cdot)$ is a real valued covariance function. Note that the mean function of a GP is globally Lipschitz continuous.

It is assumed that Δ_i switches in an unknown manner between different models of Δ_i each with a different mean m_i and kernel k_i. Furthermore, none of m_i are known a-priori. Hence, Δ_i can also be modeled as a draw from a function generating distribution, such as Dirichlet Process - Gaussian Process BNP that can be used to build a generative model for Δ_i [17]. In which case, the inference over Δ_i can be conducted in a Bayesian manner using sampling based techniques such as Gibbs sampling. However, we avoid this formulation, since Gibbs sampling is not amenable to online real-time implementation. Instead, the GP-NBC method described in this paper learns and clusters together elements of a set Υ that contains the different means m_i and the associated hyperparameters of the kernels k_i. Hence, we assume that m_i are defined to be elements of a function inclusion map Υ:

$$m_i \sim \Upsilon(x). \tag{3}$$

The above equation indicates that each GP Δ_i is based on a unique mean m_i from a set over functions Υ.

The problem we are interested in solving is tracking the states of a reference model x_{rm} in presence of the switching stochastic nonlinear uncertainty Δ_i. Let $x_{rm}(t) \in D_x \subset \mathbb{R}^n$ be the state of the reference model which characterizes the desired response of the system. The dynamics of the reference model are assumed to take on the form:

$$\dot{x}_{rm}(t) = A_{rm}x_{rm}(t) + B_{rm}u_{rm}(x(t), r(t)) \tag{4}$$

where $A_{rm} \in \mathbb{R}^{n\times n}$ and $B_{rm} \in \mathbb{R}^{n\times m}$ and the reference model input $u_{rm} \in \mathbb{R}^m$. The objective is to design a controller which can achieve a closed-loop performance such that the plant model defined by (1) tracks the reference model given by (4). Therefore a control law with a feedback term defined as $u_{fb} = K_m^\top x; K \in \mathbb{R}^{n\times m}$, a linear feedforward term, $u_{ff} = K_b^\top r; K_b \in \mathbb{R}^{m\times m}$, and an adaptive element, u_{ad} is chosen to have the desired behavior.

$$u(t) = u_{ff}(t) + u_{fb}(t) - u_{ad}(t) \tag{5}$$

The gain matrices K_m and K_r are chosen such that $A_{rm} = A + BK_m^\top$ and $B_{rm} = BK_b$ [See Section 4, (18)]. Define the tracking error as $e(t) = x(t) - x_{rm}(t)$. Then by using (1) and (4) and then substituting (5) into (1), we can obtain the tracking error dynamics as:

$$\dot{e}(t) = A_{rm}e(t) + B_{rm}(\Delta(x(t)) - u_{ad}(t)). \tag{6}$$

We are now in a position to present an overview of the presented architecture shown in figure 1. The architecture consists of three key elements, a reference command shaping MPC, GP-MRAC based adaptive control, and GP-NBC based clustering. The synthesis of u_{ad} is accomplished using the GP-MRAC method [8]. However, GP-MRAC is a tracking architecture that tracks a given reference command, it does not guarantee the optimality of the reference command, especially in pres-

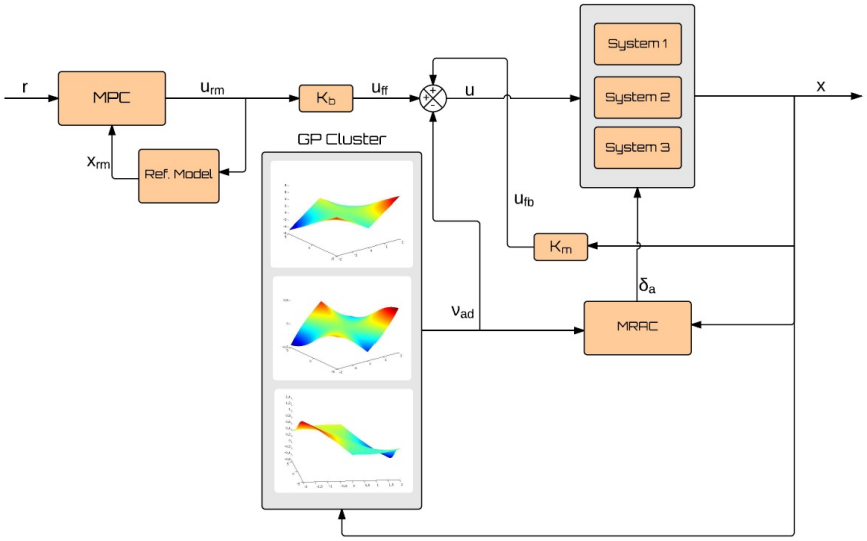


Fig. 1 The proposed solution architecture combines three key elements: a reference model shaping MPC, a GP-MRAC based adaptive reference tracking architecture, and GP-NBC based computationally efficient model clustering.

ence of state and input constraints. To accommodate this, we utilize the MPC based reference command shaping technique proposed by Muhlegg et al. [22]. Hence, in our architecture, the reference command is shaped using MPC on the linear reference model (4) in presence of state and input constraints. The optimized reference command is tracked by the GP-MRAC which simultaneously tracks the reference commands and learns a model of the underlying uncertainty Δ_i. The GP-NBC clustering algorithm utilizes online data to learn an estimate of the set Υ by learning new GPs and clustering together those GPs from which samples have been previously seen. The stability of the entire architecture is argued in Section 5 and results are presented in Section 6.

3 Review of Control Architecture

We review the basic notations and formulations of the control architecture in this section.

3.1 Model Predictive Control (MPC)

A brief overview of MPC for constrained discrete systems is presented (See [5, 7] for further details). Our focus is on illustrating how MPC can be used to obtain optimal solutions for the reference command shaping. The discretized version of the reference model dynamics is formulated as:

$$\begin{aligned} x_{rm}(k+1) &= A_{rm}x_{rm}(k) + B_{rm}u_{rm}(k) \\ y(k) &= C_{rm}x_{rm}(k) \end{aligned} \tag{7}$$

by taking a difference operation on both sides we have:

$$\begin{aligned} \Delta x_{rm}(k+1) &= A_{rm}\Delta x_{rm}(k) + B_{rm}\Delta u_{rm}(k) \\ \Delta y(k) &= C_{rm}\Delta x_{rm}(k) \end{aligned} \tag{8}$$

Defining $\bar{x}_{rm}(k) = [\Delta x_{rm}^{\top}(k) \quad y(k)]^{\top}$ results in the following state-space model.

$$\begin{aligned} \bar{x}_{rm}(k+1) &= \bar{A}_{rm}\bar{x}_{rm}(k) + \bar{B}_{rm}\Delta u_{rm}(k) \\ y(k) &= \bar{C}_{rm}\bar{x}_{rm}(k) \end{aligned} \tag{9}$$

where the triplet $(\bar{A}_{rm}, \bar{B}_{rm}, \bar{C}_{rm})$ which is called the augmented model, is as follows:

$$\bar{A}_{rm} = \begin{bmatrix} A_{rm} & o_{rm}^{\top} \\ C_{rm}A_{rm} & 1 \end{bmatrix} \bar{B}_{rm} = \begin{bmatrix} B_{rm} \\ C_{rm}B_{rm} \end{bmatrix} \bar{C}_{rm} = \begin{bmatrix} o_{rm} & 1 \end{bmatrix}$$

Upon formulation of the mathematical model, the next step in design of a predictive controller is to calculate the predicted plant output with the future control signal as the adjustable variables [30]. Assuming that at the sampling instant $k_i > 0$, the future control trajectory is denoted by,

$$\Delta u(k_i), \Delta u(k_i+1), \ldots, \Delta u(k_i+N_c-1)$$

Also, the future state variables are,

$$x(k_i+1 \mid k_i), x(k_i+2 \mid k_i), \ldots, x(k_i+N_p \mid k_i)$$

where N_p and N_c are prediction and control horizon, respectively. Based on the state-space model (9), the future state and output variables are calculated sequentially using the set of future control parameters and form the compact matrix equation (10) by defining,

$$Y = \begin{bmatrix} y(k_i+1 \mid k_i) \\ y(k_i+2 \mid k_i) \\ \vdots \\ y(k_i+N_p \mid k_i) \end{bmatrix} \Delta U = \begin{bmatrix} \Delta u((k_i) \\ \Delta u((k_i+1) \\ \vdots \\ \Delta u((k_i+N_c-1) \end{bmatrix}$$

$$Y = Fx(k_i) + \phi\Delta U \tag{10}$$

where,

$$F = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{N_p} \end{bmatrix} \phi = \begin{bmatrix} CB & 0 & 0 & \dots & 0 \\ CAB & CB & 0 & \dots & 0 \\ CA^2B & CAB & CB & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ CA^{N_p-1}B & CA^{N_p-2}B & CA^{N_p-3}B & \dots & CA^{N_p-N_c}B \end{bmatrix}$$

For a given set-point signal $r(k_i)$ at sample time k_i, the optimization objective is to find the best control parameter vector ΔU such that an error function between the set-point and the predicted output is minimized.

Assuming that the data vector which contains the set-point information is $R_s^\top = \begin{bmatrix} 1 & 1 & \dots & 1 \end{bmatrix} r(k_i)$, the cost function J that reflects the control objective is,

$$J = (R_s - Y)^\top Q(R_s - Y) + \Delta U^\top \bar{R} \Delta U \tag{11}$$

Here, Q denotes a positive definite diagonal matrix and $\bar{R} = r_w I_{Nc}$ which r_w is a tuning parameter for desired closed-loop performance. If we insert (10) into (11), we could form the MPC problem as minimizing the cost function.

$$\begin{aligned} J = {} & (R_s - Fx(k_i))^\top Q(R_s - Fx(k_i)) \\ & - 2\Delta U^\top \phi^\top Q(R_s - Fx(k_i)) \\ & + \Delta U^\top (\phi^\top Q \phi + \bar{R}) \Delta U \end{aligned} \tag{12}$$

which is subject to constraints on the incremental control ΔU and the output Y, that are collected together in equation (13) as a matrix compact form.

$$\begin{bmatrix} -C_2 \\ C_2 \\ -\phi \\ \phi \end{bmatrix} \Delta U \leq \begin{bmatrix} -U_{min} + C_1 u(k_i - 1) \\ U_{max} - C_1 u(k_i - 1) \\ -Y_{min} + Fx(k_i) \\ Y_{max} - Fx(k_i) \end{bmatrix} \tag{13}$$

3.2 Model Reference Adaptive Control (MRAC)

In this section we briefly review MRAC (See [23, 29] for further details). We begin first by describing model reference control architecture for systems without any uncertainty. Let $x(t) \in \mathbb{R}^n$ be the state vector, let $u(t) \in \mathbb{R}^m$ denote the control input and consider the following linear system,

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{14}$$

where $A \in \mathbb{R}^{n \times n}, B \in \mathbb{R}^{n \times m}$. We assume that the pair (A, B) is controllable and that B has full column rank. We assume $u(t)$ is restricted to the class of admissible

control inputs consisting of measurable functions and $x(t)$ is available for full state feedback.

A designer chosen linear reference model is used to characterize the desired closed-loop response of the system.

$$\dot{x}_{rm}(t) = A_{rm}x_{rm}(t) + B_{rm}r(t) \tag{15}$$

The reference command $r \in \mathbb{R}^m$ is assumed to be bounded and piecewise continuous, the matrix $A_{rm} \in \mathbb{R}^{n\times n}$ is chosen to be Hurwitz and $B_{rm} \in \mathbb{R}^{n\times m}$ is such that steady state accuracy is ensured. An adaptive control law including a linear feedback part, $u_{fb}(t) = K_m^\top x(t)$; $K_m \in \mathbb{R}^{n\times m}$, and a linear feedforward part, $u_{ff}(t) = K_b^\top r(t)$ with $K_b \in \mathbb{R}^{1\times m}$ is proposed to have the following form:

$$u(t) = u_{ff}(t) + u_{fb}(t) \tag{16}$$

Substituting (16) into (14), we have,

$$\dot{x}(t) = (A + BK_m^\top)x(t) + BK_b^\top r(t) \tag{17}$$

The design objective is to have the model in (14) behave as the reference model in (15). To that effect, we introduce the following model matching conditions,

$$\begin{aligned} A + BK_m^\top &= A_{rm} \\ BK_b^\top &= B_{rm} \end{aligned} \tag{18}$$

Defining the tracking error to be $e(t) = x(t) - x_{rm}(t)$ yields,

$$\dot{e}(t) = A_{rm}e(t) \tag{19}$$

which shows that the tracking error converges to zero exponentially fast since A_{rm} is Hurwitz. It follows from Lyapunov theory that there exists a unique positive definite matrix $P \in \mathbb{R}^{n\times n}$ satisfying the Lyapunov equation,

$$A_{rm}^\top P + PA_{rm} + Q = 0 \tag{20}$$

for any positive definite matrix $Q \in \mathbb{R}^{n\times n}$.

4 Adaptive Control under Switching Models

In this section we provide a method for modeling the nonlinear uncertainty $\Delta(x)$ which is assumed to be a smooth deterministic function in the Hilbert space $\mathcal{H}$. We look at a particular class of nonstationary switching functions $\bar{\Delta} = \{\Delta_1, \Delta_2, \ldots, \} \in \mathcal{H}$ that can be acted upon the system defined by (1). Our algorithm uses an adaptive control architecture which can learn the various models of uncertainties and detect change points in the underlying model to actively cancel them out.

4.1 Gaussian Process MRAC

In the presented architecture, the stochastic uncertainty Δ_i in (1) is modeled as a Gaussian process (GP) [see (2)]. A GP is a distribution over functions and a sample drawn from a GP will be a function completely characterized by the mean and variance of the GP [26]. A common choice for defining the covariance K is the squared exponential function:

$$k(x,y) = exp(-\frac{\| x-y \|^2}{2\sigma^2}), \tag{21}$$

although the presented method can accommodate other kernels. The kernel function generates a mapping ψ to an infinite dimensional Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$ such that k is an inner product defined by $k(x,y) = \langle \psi(x), \psi(y) \rangle_H$. In this RKHS, the mean m_i of the data generating stochastic process Δ_i is a linear combination of nonlinear bases:

$$m(x) = \sum_{j=1}^{\infty} \alpha_j k(x, x_j) \tag{22}$$

where $\alpha_i \in \mathcal{H}$ is a countably infinite dimensional vector, $k \in \mathcal{H}$ are the kernel functions evaluated over data points x_j vectors in the dual space. This rather general model can be learned directly from data, let $\mathcal{D} = \{(X_1, y_1)(X_2, y_2), \ldots, (X_n, y_n)\}$ be a set of state measurements sampled from an environment. Then assuming Normal distribution as the prior on the data, the GP posterior mean and variance of the GP given the measurements are estimated as:

$$\begin{bmatrix} y_t \\ y_{t+1} \end{bmatrix} \sim \mathcal{N}(0, \begin{bmatrix} K(\mathcal{D},\mathcal{D}) + \omega^2 I & k(\mathcal{D},x) \\ k(\mathcal{D},x)^\top & k(x,x) \end{bmatrix}) \tag{23}$$

The posterior distribution is computed by conditioning the joint distribution over the new measurement y_{t+1} to obtain,

$$p(y_{t+1} \mid y_t, x_t, \mathcal{D}) \sim \mathcal{N}(\hat{m}(x_{t+1}), \hat{\Sigma}(x_{t+1})) \tag{24}$$

where,

$$\begin{gathered} \hat{m}(x_{t+1}) = \alpha^\top k(x_{t+1}, \mathcal{D}) \\ \hat{\Sigma}(x_{t+1}) = k(x_{t+1}, x_{t+1}) - k(\mathcal{D}, x_{t+1})^\top [K(\mathcal{D},\mathcal{D}) + \omega^2 I]^{-1} k(\mathcal{D}, x_{t+1}) \end{gathered} \tag{25}$$

are the predictive mean and covariance respectively and $\alpha = K(\mathcal{D},\mathcal{D}) + \omega^2 I]^{-1} \bar{y}$. Hence, the mean is directly estimated from the set of available data. In essence, at any given point in time, an instantiation of the GP can be thought of as a Gaussian Radial Bases Function Network (RBFN). However, the main strength of the GP is that it does need to assume an a-priori allocation of RBF centers.

However, the main disadvantage of using the traditional GP regression techniques is that the covariance matrix increases in size as the size of $\mathcal{D}$ increases. In online applications, this can quickly become intractable as computing the inverse in

(25) can become computationally intractable. It was shown in [8] that this problem can be alleviated using online sparsification techniques, and in particular Csato and Opper's budgeted online Sparse Gaussian Process regression technique [13] which only includes valuable data points in an active *Basis Vector set* ($\mathcal{BV}$). When new data is observed, the sparsification algorithm computes how well the new data point can be approximated by the existing basis vectors using a comparative test called the *kernel linear independence test* which $\forall \psi \in \mathcal{BV}$ is defined as:

$$\gamma_{t+1} = \| \sum_{i=1}^{t} \alpha_i \psi(x_i) - \psi(x_{t+1}) \|_{\mathcal{H}} \tag{26}$$

The γ_{t+1} gives the residual distance between $\psi(x_i)$ and the GP generated by elements in $\mathcal{BV}$. An existing element ψ_m in the basis vector set which minimizes $D(\mathcal{GP} \parallel \mathcal{BV}) - D(\mathcal{GP} \parallel \mathcal{BV} \backslash \{\psi_m\})$ is removed and the new sample is added to the set. Given the basis vector set, the approximate mean and variance can be written from (25) as:

$$\begin{gathered} \hat{m}(x_{t+1}) = \alpha^\top k(x_{t+1}, \mathcal{BV}) \\ \hat{\Sigma}(x_{t+1}) = k(x_{t+1}, x_{t+1}) - k(\mathcal{BV}, x_{t+1})^\top [K(\mathcal{BV}, \mathcal{BV}) + \omega^2 I]^{-1} k(\mathcal{BV}, x_{t+1}) \end{gathered} \tag{27}$$

The reader is referred to [13] and [8] for details of the algorithmic implementation.

To achieve the tracking objective, the adaptive element is set to the online learned mean of the stochastic process Δ_i: $u_{ad}(x) = \hat{m}_i(x)$ at any state $x(t)$ from (6) to ensure that the tracking error goes to zero asymptotically.

$$\begin{gathered} u(t) = u_{ff}(t) + u_{fb}(t) - u_{ad}(t) \\ u_{ad} = \hat{m}(x) \end{gathered} \tag{28}$$

An online implementation of Gaussian Processes in MRAC is presented in [8].

4.2 Change Point Detection with Gaussian Process Non-Bayesian Clustering

In most existing GP inference techniques, including the ones discussed in 4.1, the data is assumed to be generated from a single GP. However, this technique is not applicable to cases when the underlying data generating distribution could be switching in time. In particular, an algorithm that can determine the current generative distribution does not model well the data being observed is required when Δ_i is switching. In [16, 2], the GP-NBC method is proposed for identifying separable GP models from sequentially observable outputs, under mild assumptions [16], on the class of functions $\mathcal{F}$ in which the various models exist. We present a brief overview of GP-NBC.

For a GP, the log-likelihood of a subset of points y can be evaluated as:

Algorithm 1 GP Clustering [16]

Input: Initial data (X,Y), lps size l, model deviation η
Initialize GP Model 1 from (X,Y).
Initialize set of least probable points $S = \emptyset$.
while new data is available **do**
  Denote the current model by M_c.
  If data is unlikely with respect to M_c, include it in S.
  if $|S| == l$ **then**
    for each model M_i **do**
      Calculate log-likelihood of data points S using having been generated from current model M_i (29) $\log(S|M_i)$, and find highest likelihood model M_h, making M_h current model.
      Create new GP M_S from S.
      if $\frac{1}{l}(\log(S|M_S) - \log(S|M_c)) > \eta$ **then**
        Add M_S as a new model.
      end if
    end for
  end if
end while

$$\log P(y \mid x, M) = -\frac{1}{2}(y-\mu(x))^\top \Sigma_{xx}(y-\mu(x)) - \log|\Sigma_{xx}|^{1/2} + C \tag{29}$$

where $\mu(x) = K(X,x)^\top (K(X,X)+\omega_n^2 I)^{-1} Y$ is the mean prediction of the model M and $\Sigma_{xx} = K(x,x)+\omega_n^2 I - K(X,x)^\top (K(X,X)+\omega_n^2 I)^{-1} K(X,x)$ is the conditional variance plus the measurement noise. The log-likelihood contains two terms which account for the deviation of points from the mean, $\frac{1}{2}(y-\mu(x))^\top \Sigma_{xx}(y-\mu(x))$, as well as the relative certainty in the prediction of the mean at those points $\log|\Sigma_{xx}|^{1/2}$. GP-NBC algorithm, at all times, maintains a set of points S which are considered unlikely to have arisen from the current GP model M_c. The set S is used to create a new GP M_S, which is tested against the existing models M_i using a non-Bayesian hypothesis test to determine whether the new model M_S merits instantiation as a new model, or should be clustered with an existing one. This test is defined as:

$$\frac{P(y \mid M_i)}{P(y \mid M_j)} \underset{\hat{M}_j}{\overset{\hat{M}_i}{\gtrless}} \eta \tag{30}$$

where $\eta = (1-p)/p$, and $p = P(M_1)$. If the quantity on the left-hand side is greater than η, then the hypothesis M_i (i.e. that the data y is better represented by M_i) is chosen, and vice versa. The GP NBC algorithm is reproduced in algorithm 1 (see [16, 2] for more details).

5 Stability Analysis

This section proves the stability of the presented adaptive-optimal control architecture. From the tracking error dynamics in (6), it follows that since A_{rm} is Hurwitz, if $\|\Delta - u_{ad}\|$ is bounded, the tracking error stays bounded. The main advantage of the reference command shaping architecture utilized here is that optimizing the output of the reference model using MPC will not change the tracking error dynamics in (6) [22].

Remark 1 *The tracking error dynamics can be derived as follows:*

$$\begin{aligned}\dot{e} &= \dot{x} - \dot{x}_{rm} = Ax + B(u+\Delta) - A_{rm}x_{rm} - B_{rm}r \\ &= Ax + B(K_m x + K_b r - u_{ad} + \Delta) - A_{rm}x_{rm} - B_{rm}r \\ &= (A + BK_m)x - A_{rm}x_{rm} + BK_b r - B_{rm}r - Bu_{ad} + B\Delta \\ &= A_{rm}e + B(\Delta - u_{ad})\end{aligned}$$

Note that due to the matching conditions in (18), the reference signal is canceled out, hence, the stability of the architecture is not affected directly by the MPC based reference command optimization. Hence, unlike previous learning based MPC architecture [4, 10], we do not need to explicitly account for the effect of including MPC in the closed loop in the stability analysis.

Let $\xi(t)$ be a zero-mean Wiener process and G_σ be a linear operator associated with the covariance kernel $k(z,z')$ [26], then following the analysis in [8] we let,

$$\Delta(z) = m(z(t)) + G_\sigma(t, z(t))d\xi \tag{31}$$

Hence, the tracking error dynamics can be written as [8],

$$\dot{e} = A_{rm}e + B(\varepsilon_m + G_\sigma) \tag{32}$$

where $\varepsilon_m = m - \hat{m}$ and $u_{ad} = \hat{m}$ is the estimation of the mean m. Now, to bound $\|\Delta - u_{ad}\|$, we can utilize the Global Approximation Theorem for GPs from [8], which yields the following bound on ε_m:

$$\| \varepsilon_m \| \leq \frac{2\kappa^2 M\sqrt{k_{max}}}{\omega^4} + \frac{\kappa k_{max} M}{\omega^2} \tag{33}$$

where k_{max} is the greatest kernel approximation error, κ is the maximum value that the kernel can take (1 for Gaussian kernel), ω is the process noise, M is the largest value of an outlying measurement [8].

The following theorem guarantees the stability of the presented architecture:

Theorem 1 *Assume the system in (1), control law in (5) and the uncertainty Δ which is representable by a Gaussian Process. Then, the outlined adaptive-optimal control algorithm and $u_{ad} = \hat{m}(z(t))$ guarantee that the system is mean square uniformly ultimately bounded a.s.*

Proof. The proof follows directly from [8]. In particular, by letting $V(e(t)) = \frac{1}{2}e^{\top}(t)Pe(t)$ be the Lyapunov candidate, where P is the positive definite solution to the Lyapunov equation (20), it can be shown that the Itô differential of the Lyapunov candidate is less than zero outside of the set,

$$\Theta_{\gamma} = \{\| e \| \geq \frac{c_5 M + \sqrt{(c_5 M)^2 + 2\lambda_{min}(Q)c_1}}{\lambda_{min}(Q)}\} \tag{34}$$

where $c_3 = \frac{2\kappa^2}{\omega^4}$, $c_4 = \frac{\kappa}{\omega^2}$, and $c_5 = c_3\sqrt{k_{\max}} + c_4 k_{\max}$.

Remark 2 *The bound in* (34) *gets tighter as the kernel density increases. The ability to handle higher kernel density directly depends on the available onboard processing power and memory, hence increasing processing power and memory can help reduce the error bound. The bound in* (34) *gets looser with large outliers M, but can be balanced with larger regularizing parameter* ω*. Finally, note that unlike RBF-NN based proofs [28], operation over a compact domain where kernels have been a-priori allocated need not be assumed, the domain can grow with data, but the kernel budget restricts attainable accuracy.*

Remark 3 *The above theorem guarantees that the system states will be driven inside a positively invariant set exponentially fast. Therefore if the switching interval between any two consecutive model* (Δ_i) *switches is sufficiently large, the stability of the switching nonlinear dynamical system in* (1) *can be guaranteed. Note that the required time interval between any two switches will always be finite, because of the exponential convergence guarantees.*

We now analyze the effect on the stability due to potential miss-classification by GP-NBC of the underlying model. Note first that GP-NBC is guaranteed to correctly classify the underlying model within a finite number of samples [16]. This property allows us to prove the following corollary:

Corollary 1 *Consider a system and the tracking error dynamics as described by* (1) *and* (6)*. If the worst case* ε_m *is defined as* $\varepsilon_w = \mid m - \hat{m}_{worst} \mid$ *where* $\hat{m}_{worst} = \sup_{i\in\Upsilon}(\hat{m}_{true} - \hat{m}_i)$ *and the probability of making an incorrect prediction and detecting a switch point are defined as* δ_L *and* δ_D*, respectively, then, the growth in the tracking error due to a potential miss-classification by GP-NBC is bounded with probability* $1 - \delta_L - \delta_D$*.*

Proof. The presented proof is based on the worst case ε_w which is calculated using the largest difference of ε and follows the analysis used in [16] to obtain an upper bound on the growth of the tracking error. By applying the triangle inequality to the solution of the tracking error dynamics (6), we obtain the bounds on $e(t)$ as,

$$\| e(t) \| \leq \| exp(A(t-t_0))e(t_0) \| + \| \int_{\tau}^{t} exp(A(t-\tau))B(\tau)(\Delta - u_{ad})d\tau \| \tag{35}$$

From [16] there exists an upper bound on the total number of time steps GP-NBC can make an incorrect prediction when the model has switched. This guarantees

that the integral defined in the transient response is finite. Since $\Delta - u_{ad}$ is upper bounded by ε_w the above equation reduces to

$$\| e(t) \| \leq \| exp(A(t-t_0))e(t_0) \| + \| A^{-1}[exp(A(t)) - I]B(t)\varepsilon_w \| \tag{36}$$

Since the total number of time steps GP-NBC makes an incorrect prediction is $n+(m-1)\mathcal{N}_c(U_E, d_x)$ with probability $1-\delta_L-\delta_D$ [16], the integral in the transient response is finite with the same probability. Here n is the worst case bound on the number of samples such that GP makes an incorrect prediction, m is the window size to sufficiently distinguish any two models and $\mathcal{N}_c(U_E, d_x)$ is the covering number [16].

6 Results

In this section we evaluate the presented architecture through simulation on a representative flight control problem. Consider the uncertain nonlinear dynamical system in the form (1):

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} (u + \Delta) \tag{37}$$

This class of models is applicable to the class of wingrock dynamics:

$$\begin{bmatrix} \dot{\phi} \\ \dot{p} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \phi \\ p \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} (\delta_a + \Delta) \tag{38}$$

where $\delta_a = u$ in (37) is the aileron deflection, ϕ is the roll angle, and p is the roll rate. The generic nonlinear and time-varying uncertainty Δ takes on the following form for wingrock dynamics $\Delta = W_0 + W_1\phi + W_2 p + W_3|\phi|p + W_4|p|p + W_5\phi^3$. Here we assume that the weights W are randomly switching between the three different sets of weights given below:

$$\begin{bmatrix} W_0 \\ W_1 \\ W_2 \\ W_3 \\ W_4 \\ W_5 \end{bmatrix} = \begin{bmatrix} 2 \\ 0.2314 \\ 0.1918 \\ -0.6245 \\ 0.0095 \\ 0.0214 \end{bmatrix}, \begin{bmatrix} -3 \\ -0.01859 \\ 0.01516 \\ -0.06245 \\ 0.0095 \\ 0.0214 \end{bmatrix}, \begin{bmatrix} 0 \\ 0.324 \\ -0.01516 \\ -0.06245 \\ 0.0095 \\ 0.0214 \end{bmatrix}$$

The natural frequency and the damping ratio of the chosen reference model are $\omega_{rm} = 2$ and $\zeta_{rm} = 0.5$, so that:

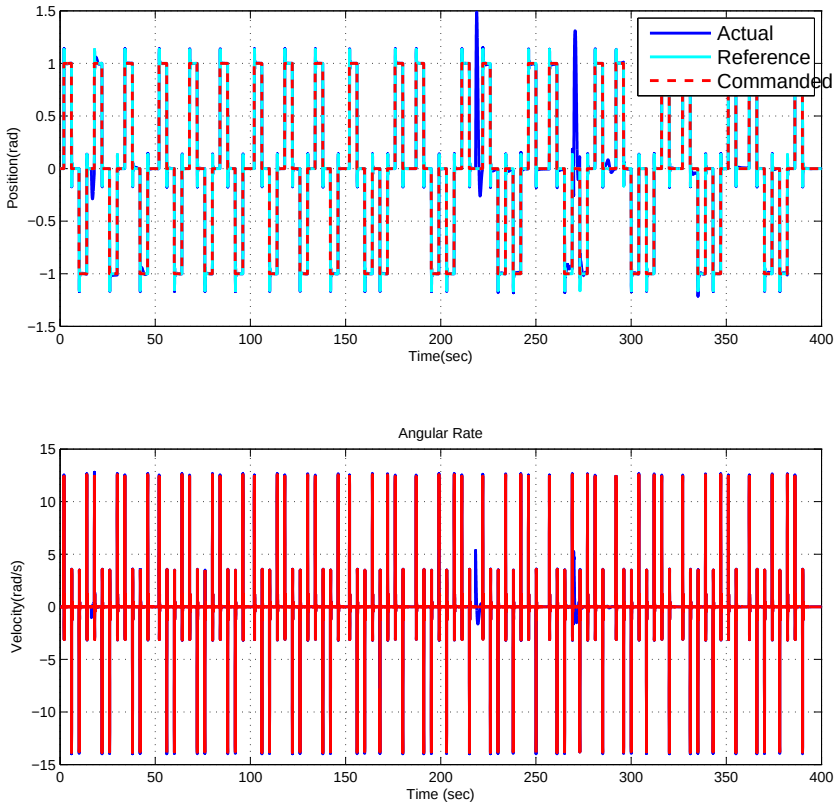
$$\begin{bmatrix} \dot{x}_{rm_1} \\ \dot{x}_{rm_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -4 & -2 \end{bmatrix} \begin{bmatrix} x_{rm_1} \\ x_{rm_2} \end{bmatrix} + \begin{bmatrix} 0 \\ 4 \end{bmatrix} u_{rm} \tag{39}$$

The feedback and feedforward gains are $K_m = [-4 \quad -2]$, $K_b = 4$, respectively. The simulation runs for 250 seconds with a time step of 0.01 seconds.

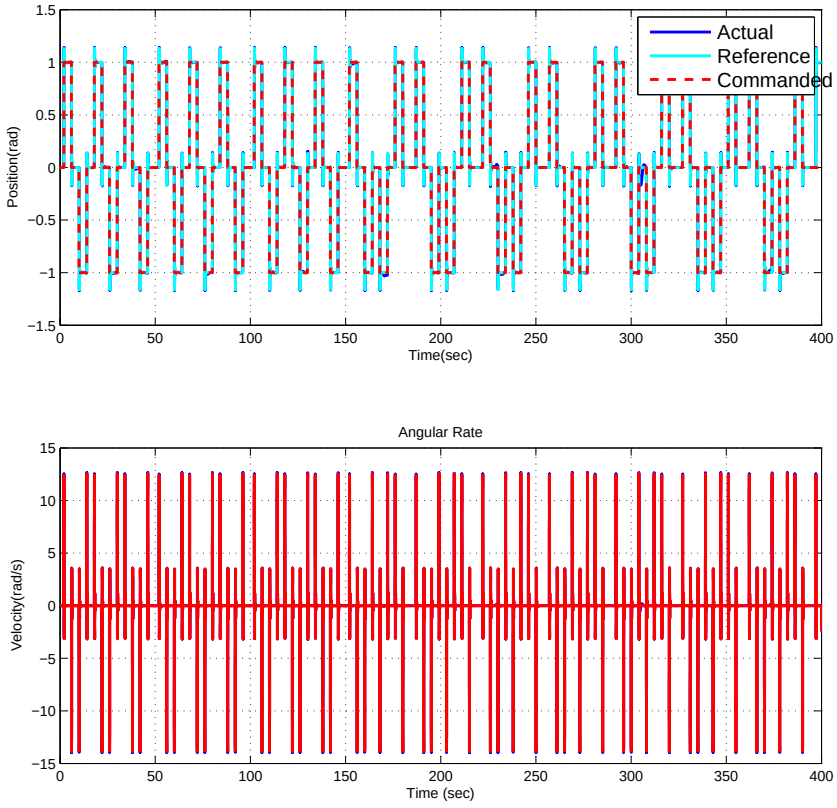
The weights are assumed to switch in a manner unknown to the controller, however, we assume that there is a finite time interval between any two switches. In order to test the performance of our method, we compare the presented GP-MRAC-NBC algorithm with a GP-MRAC architecture which learns the uncertainty model without clustering. We show comparisons between the positions and velocities between both the methods along with the error performances.

Figure 2 presents the tracking performance. The baseline GP-MRAC has suboptimal transient performance, since it can be seen that once the modeling error is switched to second set of weights, the positions are not optimally tracked. However in steady-state, when GP-MRAC does learn the new model, GP-MRAC's performance is similar to GP-MRAC-NBC.

Figure 3 compares the error performance of the baseline with the GP-MRAC-NBC. A degradation is noticed in the error performance when no clustering is performed. It can also be noticed that the performance of the clustering algorithm improves over time as the algorithm sees more and more data from a particular model. For example, the error of the system with the first set of weights has improved in performance when first model is encountered the second time. The same is true for second and third sets of weights. This indicates long-term learning and reuse of previous experience. Sharp peaks in the error plot indicate the points where the clustering algorithm is using the data to identify the right model and hence there is a small degradation in the performance, however, once identification is done, the tracking error significantly reduces. Figure 4 shows that the GP-NBC algorithm is able to correctly detect changes in the model and cluster together data from the same underlying GP.

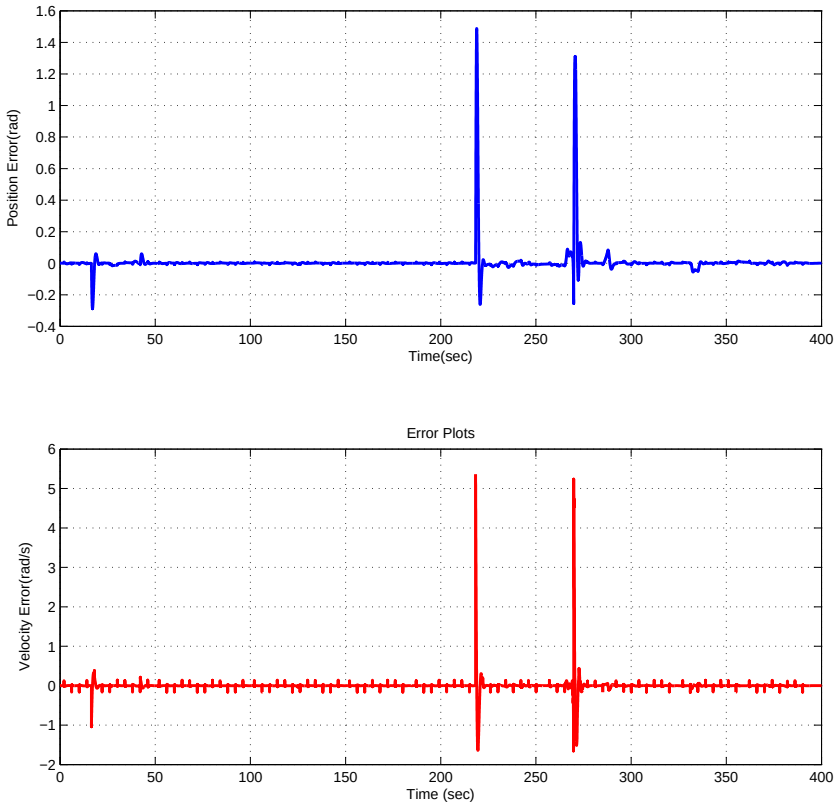


(a) Baseline Performance

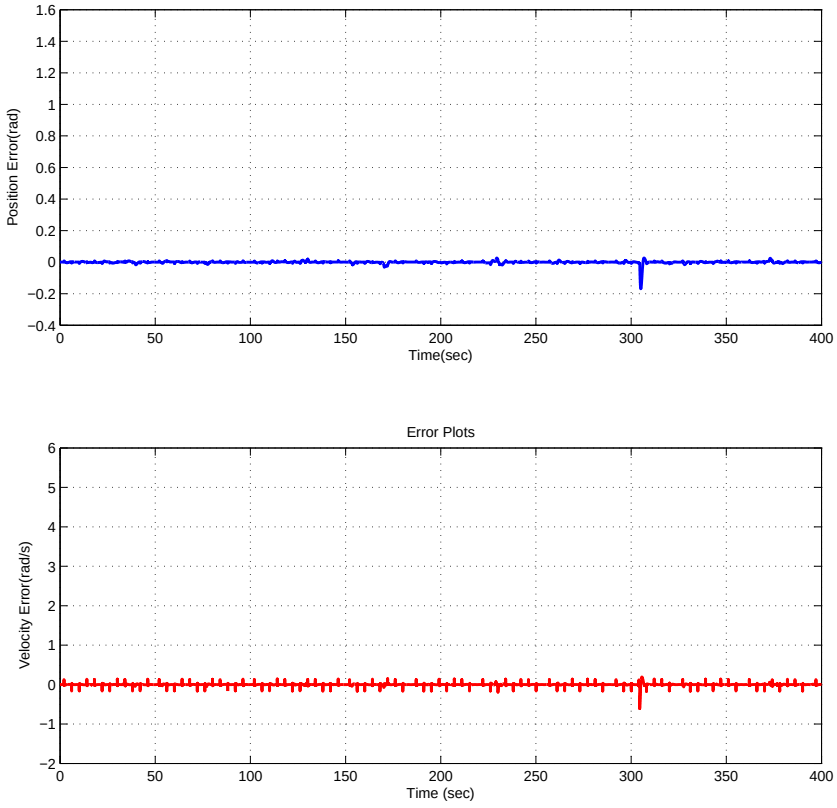


(b) Performance with Clustering

Fig. 2 Comparison in performance without and with clustering the uncertainty models.



(a) Baseline Error



(b) Error with Clustering

Fig. 3 Comparison in error performance without and with clustering the uncertainty models.

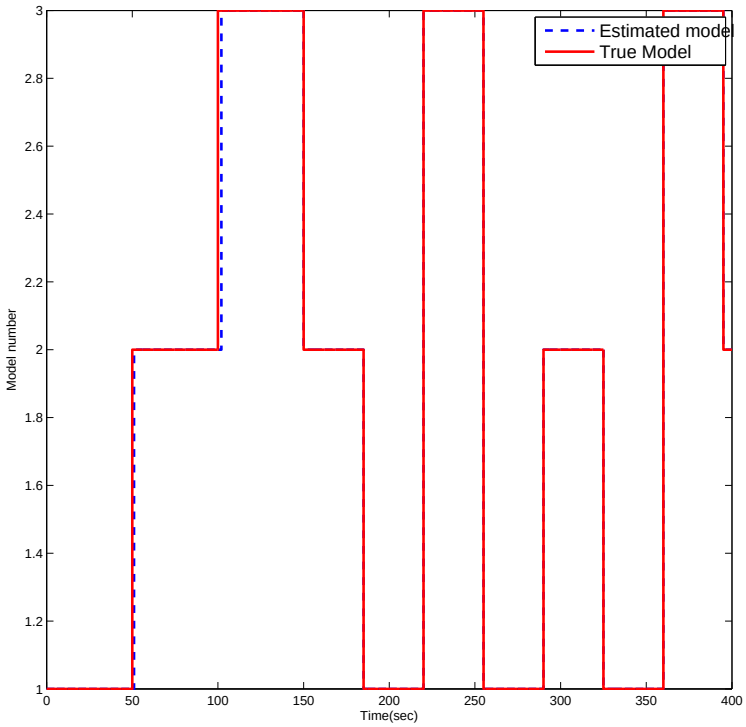


Fig. 4 Model estimation performance of the Clustering Algorithm

7 Conclusion

We presented a GP based adaptive-optimal control architecture for adaptive control of aerospace systems with dynamically changing stochastic modeling uncertainty. The key salient feature of our architecture is that not only can it detect changes, but it uses online GP clustering to enable the controller to utilize past learning of similar models to significantly reduce learning transients. The stability of the architecture was argued theoretically, and empirical results showed that it can be used in online settings. These results indicate the possibility of designing flight control and decision making systems that can adapt to unforeseen situations while being able to leverage past experience to minimize suboptimal performance due by reducing the time spent in learning.

8 Acknowledgment

This research was supported in part by by University of Oklahoma NASA Cooperative Agreement No. NNX13AB21A and Air Force Office of Scientific Research Award Number FA9550-14-1-0399.

References

1. Veronica Adetola, Darryl DeHaan, and Martin Guay. Adaptive model predictive control for constrained nonlinear systems. *Systems & Control Letters*, 58(5):320–326, 2009.
2. Rakshit Allamaraju, Hassan Kingravi, Allan Axelrod, Girish Chowdhary, Robert Grande, Christopher Crick, Weihua Sheng, and Jonathan How. Human aware path planning in urban environments with nonstationary mdps. In *International Conference on Robotics and Automation*, Hong Kong, China, 2014. IEEE.
3. Karl J Åström and Björn Wittenmark. *Adaptive control*. Courier Dover Publications, 2013.
4. Anil Aswani, Humberto Gonzalez, S Shankar Sastry, and Claire Tomlin. Provably safe and robust learning-based model predictive control. *Automatica*, 49(5):1216–1226, 2013.
5. Alberto Bemporad and Manfred Morari. Robust model predictive control: A survey. In *Robustness in identification and control*, pages 207–226. Springer, 1999.
6. P. Bouffard, A. Aswani, and C. Tomlin. Learning-based model predictive control on a quadrotor: Onboard implementation and experimental results. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, page 279284, may 2012.
7. Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer Science & Business Media, 2013.
8. G. Chowdhary, H.A. Kingravi, J.P. How, and P.A. Vela. Bayesian nonparametric adaptive control using gaussian processes. *Neural Networks and Learning Systems, IEEE Transactions on*, PP(99):1–1, 2014.
9. Girish Chowdhary and Eric Johnson. Concurrent learning for convergence in adaptive control without persistency of excitation. In *Decision and Control (CDC), 2010 49th IEEE Conference on*, pages 3674–3679. IEEE, 2010.
10. Girish Chowdhary, Maximilian Mühlegg, Jonathan P. How, and Florian Holzapfel. Concurrent learning adaptive model predictive control. In Qiping Chu, Bob Mulder, Daniel Choukroun, Erik-Jan Kampen, Coen Visser, and Gertjan Looye, editors, *Advances in Aerospace Guidance, Navigation and Control*, pages 29–47. Springer Berlin Heidelberg, 2013.
11. Girish Chowdhary, Tansel Yucelen, Maximillian Mühlegg, and Eric N Johnson. Concurrent learning adaptive control of linear systems with exponentially convergent bounds. *International Journal of Adaptive Control and Signal Processing*, 27(4):280–301, 2013.
12. Lehel Csató and Manfred Opper. Sparse on-line gaussian processes. *Neural Computation*, 14(3):641–668, 2002.
13. Lehel Csató and Manfred Opper. Sparse on-line gaussian processes. *Neural computation*, 14(3):641–668, 2002.
14. Manuel A Duarte and Kumpati S Narendra. Combined direct and indirect approach to adaptive control. *Automatic Control, IEEE Transactions on*, 34(10):1071–1075, 1989.
15. Hiroaki Fukushima, Tae-Hyoung Kim, and Toshiharu Sugie. Adaptive model predictive control for a class of constrained linear systems based on the comparison model. *Automatica*, 43(2):301–308, 2007.
16. Robert C. Grande. Computationally efficient Gaussian process changepoint detection and regression. Master's thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, Cambridge MA, June 2014.
17. Joshua Joseph, Finale Doshi-Velez, A. S. Huang, and N. Roy. A bayesian nonparametric approach to modeling motion patterns. *Autonomous Robots*, 31(4):383–400, 2011.

18. Suresh Kannan. *Adaptive Control of Systems in Cascade with Saturation*. PhD thesis, Georgia Institute of Technology, Atlanta Ga, 2005.
19. Eugene Lavretsky. Combined/composite model reference adaptive control. *IEEE Transactions on Automatic Control*, 54(11):2692, 2009.
20. Jay H Lee and Brian Cooley. Recent advances in model predictive control and other related areas. In *AIChE Symposium Series*, volume 93, pages 201–216. New York, NY: American Institute of Chemical Engineers, 1971-c2002., 1997.
21. David Mayne. Nonlinear model predictive control: Challenges and opportunities. In *Nonlinear model predictive control*, pages 23–44. Springer, 2000.
22. Maximilian Mühlegg, Girish Chowdhary, and Florian Holzapfel. Optimizing reference commands for concurrent learning adaptive-optimal control of uncertain dynamical systems. In *GNC*. AIAA, 2013.
23. Kumpati S. Narendra and Anuradha M. Annaswamy. *Stable Adaptive Systems*. Prentice-Hall, Englewood Cliffs, 1989.
24. GD Nicolao, L Magi, and R Scattolini. Nonlinear model predictive control, volume 26 of progress in systems and control theory, chapter stability and robustness of nonlinear receding horizon control, 2000.
25. S Joe Qin and Thomas A Badgwell. An overview of industrial model predictive control technology. In *AIChE Symposium Series*, volume 93, pages 232–256. New York, NY: American Institute of Chemical Engineers, 1971-c2002., 1997.
26. Carl Edward Rasmussen. Gaussian processes for machine learning. 2006.
27. C.E. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.
28. R.M. Sanner and J.-J.E. Slotine. Gaussian networks for direct adaptive control. *Neural Networks, IEEE Transactions on*, 3(6):837863, nov 1992.
29. Gang Tao. *Adaptive Control Design and Analysis*. Wiley, New York, 2003.
30. Liuping Wang. *Model predictive control system design and implementation using MATLAB®*. springer, 2009.