Bayesian Nonparametric Model Reference Adaptive Control Using Gaussian Process Regression

Juan Augusto Paredes Salazar
jparedes@umich.edu

***Abstract*— Model Reference Adaptive Control (MRAC) methods require a reference model with an adaptive term that mitigates errors caused by model uncertainty. Most implementations rely on parametric adaptive elements, such a Radial Basis Functions (RBFs), usually requiring fixing a number of parameters and tuning through expert intuition. However, in these cases, tracking model uncertainty becomes less accurate as the state moves away from the assumed domain. In this paper, a Gaussian Process (GP) will be proposed as a nonparametric alternative for estimating the model uncertainty that requires minimal tuning and doesn't require previous knowledge regarding the domain of the state or the model uncertainty. For this purpose, Gaussian Process Regression will be explained, as well as methods for its online implementation. Finally, both a RBFN and a GP based MRAC controller will be tested on a stochastic system whose state strays from the assumed domain to compare their reference and uncertainty tracking performance.**

I. INTRODUCTION

In many real-world applications, obtaining an exact model of the underlying dynamics that affect a system is either implausible, time consuming or, in the case a high-fidelity model is available, would require a substantial amount of computational power to account for all the represented dynamics. Adaptive control algorithms offer the promise of being capable of controlling complex systems while having minimal knowledge about the plant to the controlled. One such algorithm is the Model Reference Adaptive Control (MRAC), which offers good performance guarantees in the presence of uncertainty. A regular manner of estimating the model uncertainty inherent to these sort of controllers is composed by Radial Basis Function Networks (RBFNs) [1] [2]. However, this approach presents a locality issue, meaning that they are not guaranteed to capture an accurate uncertainty model if the state strays away from the presumed domain. As an alternative, both [3] and [4] propose employing Gaussian Processes (GPs) as Bayesian nonparametric adaptive elements to obtain an uncertainty model approximation which doesn't require previous state domain information and only needs hyperparameter tuning. Through Gaussian Process Regression (GPR), the model uncertainty will be approximated. The next sections will be focused on explaining the basics of MRAC, the implementation of RBFNs for uncertainty estimation, the online implementation of GPR in the MRAc algorithm and comparing both approaches via stochastic model simulations.

II. APPROXIMATE MODEL INVERSION BASED MODEL REFERENCE ADAPTIVE CONTROL (AMI-MRAC)

The AMI-MRAC algorithm overall scheme is displayed in figure 1. Consider state $x(t) = [x_1^T(t), x_2^T(t)]^T \in \mathbb{R}^n$, with $x_1(t) \in \mathbb{R}^{n_s}$ and $x_2(t) \in \mathbb{R}^{n_s}$, such that $2n_s = n$. Let $\delta \in \mathbb{R}^l$ $(l \leq n_s)$ be the input for the following multiple input controllable control-affine nonlinear dynamical system:

$$\dot{x}_1(t) = x_2(t)$$
$$\dot{x}_2(t) = f(x(t)) + b(x(t))\delta(t), \tag{1}$$

such that functions $f(t)$ $(f(0) = 0)$ and b are partially unknown and there exists a solution to (1) over a given domain.

Under these settings, the AMI-MRAC approach attempts to find a pseudo-control input $\nu \in \mathbb{R}^{n_s}$ that allows the system to achieve a desired acceleration (represented by $\dot{x}_2(t)$ in (1)). Should the plant dynamics in (1) be known and invertible, finding ν becomes trivial. However, since this is not usually the case, an approximate inversion model is employed, such as $\hat{f}(x) + \hat{b}(x)\delta$, in which $\hat{b}$ is chosen to be nonsingular for any feasible state $x(t)$. Then, given a desired pseudo-control input ν, a control input δ can be obtained through the following inversion:

$$\delta = \hat{b}^{-1}(x)(\nu - \hat{f}(x)). \tag{2}$$

The use of the previously mentioned approximate model leads to a modelling error Δ for the system, such that, considering $z = [x^T, \delta^T]^T \in \mathbb{R}^{n+l}$:

$$\dot{x}_2 = \nu(z) + \Delta(z), \tag{3}$$

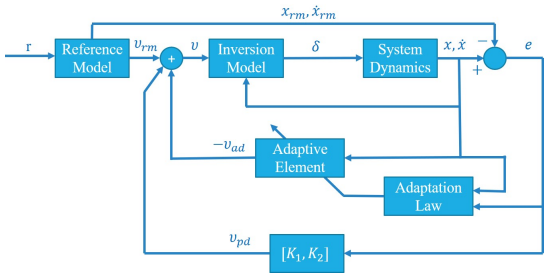

Fig. 1. Approximate Model Inversion Based Model Reference Adaptive Control (AMI-MRAC) Scheme.

with,

$$\Delta(z) = f(x) - \hat{f}(x) + (b(x) - \hat{b}(x))\delta. \tag{4}$$

If b is known and invertible with respect to δ, then an inversion model that does not depend on δ exists. A reference model is used to characterize the desired system response, such as:

$$\begin{aligned} \dot{x}_{1_{rm}} &= x_{2_{rm}}, \\ \dot{x}_{2_{rm}} &= f_{rm}(x_{rm}, r), \end{aligned} \tag{5}$$

in which $f_{rm}(x_{rm}, r)$ represents the reference model dynamics, x_{rm} is the reference state (to be tracked) and $r(t)$ represents the command of the reference model (not to be confused with the desired input). It is assumed that $f_{rm}(x_{rm}, r)$ allows x_{rm} to be bounded under a bounded reference command.

The tracking error is defined as $e(t) = x_{rm}(t) - x(t)$ and pseudo-control input ν is represented by the following equation:

$$\nu = \nu_{rm} + \nu_{pd} - \nu_{ad}, \tag{6}$$

in which $\nu_{rm} = \dot{x}_{2_{rm}}$, $\nu_{pd} = [K_1, K_2]e$, such that $K_1 \in \mathbb{R}^{n_s \times n_s}$ and $K_2 \in \mathbb{R}^{n_s \times n_s}$, and ν_{ad} represents the adaptive term that accounts for the modelling uncertainty.

While it is assumed that there exists unique fixed-point solution for $\nu_{ad} = \Delta(x, \nu_{ad})$ for all x in a controllable subspace, this is required for ν_{ad} to successfully cancel the model uncertainty. Under said assumption and using (3), the tracking error dynamics can be written as:

$$\dot{e} = \dot{x}_{rm} - \dot{x} = \dot{x}_{rm} - \begin{bmatrix} x_2 \\ \nu + \Delta \end{bmatrix} \tag{7}$$

Letting $A = \begin{bmatrix} 0 & I \\ -K_1 & -K_2 \end{bmatrix}$, $B = \begin{bmatrix} 0 // I \end{bmatrix}$, in which $0 \in \mathbb{R}^{n_s \times n_s}$ and $I \in \mathbb{R}^{n_s \times n_s}$ correspond to the zero and identity matrices correspondingly, and replacing ν using (6), the equation turns into:

$$\begin{aligned} \dot{e} &= \dot{x}_{rm} - \begin{bmatrix} x_2 \\ \dot{x}_{2_{rm}} + [K_1, K_2]e - \nu_{ad} + \Delta(z) \end{bmatrix} \\ \dot{e} &= \begin{bmatrix} x_{2_{rm}} - x_2 \\ -[K_1, K_2]e \end{bmatrix} + \begin{bmatrix} 0 \\ \nu_{ad} - \Delta(z) \end{bmatrix} \\ \dot{e} &= Ae + B[nu_{ad}(z) - \Delta(z)] \end{aligned} \tag{8}$$

Hence, the state controller feedback ν_{pd} must be chosen to make A Hurwitz to obtain a asymptotically stable response (given the previously mentioned assumption), such that, for any positive definite matrix $Q \in \mathbb{R}^{n \times n}$, a positive definite matrix $P \in \mathbb{R}^{n \times n}$ exists as a solution for the Lyapunov equation:

$$0 = A^T P + PA + Q \tag{9}$$

A. Gaussian RBFN for model uncertainty estimation

When implementing Gaussian RBFN for estimating Δ, the adaptive term of the pseudo-control input ν is represented by a linear combination of Gaussian RBFs, such that:

$$\nu_{ad} = W^T \Phi(z), \tag{9}$$

composed by a vector of weights $W \in \mathbb{R}^{n_2 \times q}$ and a q dimensional vector of RBFs $\Phi(z) = [1, \phi_2(z), \phi_3(z), ..., \phi_q(z)]^T$. For each RBF, such that $\phi_i = exp\left(\frac{-||x - c_i||^2}{2\mu_i^2}\right)$, given an index $i = 2, 3, ..., q$, a centroid c_i and a width μ_i need to be set. Furthermore, given adaptation law:

$$\dot{W} = Proj(-\Gamma_W e^T P B \Phi(z)), \tag{10}$$

in which the projection operator is used to bound weights (as proposed in [5] and [2]), Γ_W represents a positive definite learning rate matrix and the rest of the parameters have been previously defined (P can be obtained from any feasible Q, even when the latter one is the identity matrix), given enough excitation from the measured states $x(t)$ and a large enough q, the modelling uncertainty can be accurately reproduced by the Gaussian RBFN given that the system operates within the domain dictated by the centroids c_i. Therefore, there is no guarantee that the accuracy of the estimation will remain accurate if the state strays away from the domain defined by the preallocated centers. Under these circumstances, a new adaptation term is proposed to assess the domain limitation.

III. ADAPTIVE CONTROL USING GAUSSIAN PROCESS REGRESSION (GPR)

Assuming that model uncertainty can be described by a time varying mean and covariance function, it is proposed that a Gaussian Process (GP) can be used to accurately approximate said uncertainty. A GP is defined as a collection of random variables in which every finite subset is jointly Gaussian. A GP is a distribution over function, meaning that a function always results from a GP. Furthermore, the joint Gaussian condition implies that each GP can be completely characterized by their second order statistics (mean and covariance). Assuming that Δ follows a GP, then the following can be assessed:

$$\Delta(\cdot) \sim \mathcal{GP}(m(\cdot), k(\cdot, \cdot)), \tag{11}$$

where $m(\cdot)$ is the mean function and $k(\cdot, \cdot))$ is a positive definite covariance kernel function.

A. Gaussian process regression (GPR)

Given a set of state measurements $Z_\tau = \{z_1, z_2, \cdots, z_\tau\}$ and considering an associated measured output $y(z_i)$ for each z_i, such that $y(z_i) = m(z_i) + \epsilon_i$, such that $\epsilon_i \sim \mathcal{N}(0, \omega^2)$, training set $\mathcal{BV}$ is composed by Z_τ and the stacked corresponding outputs $Y_\tau = [y_1, y_2, \cdots, y_\tau]^T$. Set Z_τ defines a covariance matrix $K_{ij} := k(z_i, z_j)$, which relies on the chosen covariance kernel. The most common kernel

(and the one to be implemented) is the Gaussian RBF kernel of the form:

$$k(z, z') = exp\left(-\frac{||z - z'||^2}{2\mu^2}\right). \tag{12}$$

Given a new measured input $z_{\tau+1}$, Gaussian Process Regression can be used to estimate the measured output $y_{\tau+1}$ that would correspond to the previously mentioned state. Defining $k_{z_{\tau+1}} = K(z_{\tau+1}, Z_\tau)$, $k^*_{\tau+1} = k(z_{\tau+1}, z_{\tau+1})$, $C_\tau = (K(Z_\tau, Z_\tau) + \omega^2 I)^{-1}$, $\beta_{\tau+1} = C_\tau Y_\tau$; then, the posterior distribution obtained by conditioning the joint Gaussian distribution over observation $z_{\tau+1}$ is defined by:

$$p(y_{\tau+1}|Z_\tau, Y_\tau, z_{\tau+1}) \sim \mathcal{N}(\hat{m}_{\tau+1}, \hat{\Sigma}_{\tau+1}), \tag{13}$$

in which:

$$\hat{m}_{\tau+1} = \beta^T_{\tau+1} k_{z_{\tau+1}}, \tag{14}$$

$$\hat{\Sigma}_{\tau+1} = k^*_{\tau+1} - k^T_{z_{\tau+1}} C_\tau k_{z_{\tau+1}}, \tag{15}$$

correspond to the updated mean and covariance estimates respectively, that is, the values of the mean and covariance functions of the GP evaluated at $z_{\tau+1}$. This scheme will be used to obtain an accurate estimate of the model uncertainty.

B. MRAC using Gaussian Processes (GP-MRAC) and online GPR

Assuming that model uncertainty can be modeled such that:

$$\Delta(z) \sim \mathcal{GP}(m(z), k(z, z')), \tag{16}$$

then the proposal for the adaptive element ν_{ad} to counter said uncertainty is the following:

$$\nu_{ad}(z) \sim \mathcal{GP}(\hat{m}(z), k(z, z')), \tag{17},$$

in which $\hat{m}(z)$ is and estimate of $m(z)$ and will be set in the GP-MRAC algorithm as ν_{ad}. Since GPs are completely characterized by their first two moments (as mentioned before), then $||\hat{m}(z) - m(z)|| < \epsilon_1$ implies $||\nu_{ad} - \Delta|| < \epsilon_2$ for some $\epsilon_1, \epsilon_2 > 0$, meaning that, as the size of the training set $\mathcal{BV}$ increases, the estimated posterior will converge to the true posterior. However, this becomes untenable as $\mathcal{BV}$ grows in size in an online implementation since calculations become more complex and computationally expensive (especially the inverse operations). Therefore, aside from setting a maximum number of data points p_{max}, a manner of determining when to add new data points to $\mathcal{BV}$ and to eliminate entries when p_{max} is reached need to be defined.

Given a new measured output $y_{\tau+1}$ at measured state $z_{\tau+1}$, this set should only be added to $\mathcal{BV}$ if:

$$\gamma_{\tau+1} = k^*_{\tau+1} - k^T_{z_{\tau+1}} K^{-1}_{Z_\tau} k_{z_{\tau+1}} > \epsilon_{tol}, \tag{18}$$

in which $\gamma_{\tau+1}$ is a measure of how distinct measured state $z_{\tau+1}$ is from the state measurements in Z_τ in $\mathcal{BV}$. Intuitively, this procedure ensures that the training set accepts only entries that enrich the trained model. Depending on how discerning one wants the training set to be, one can make the tolerance parameter ϵ_{tol} larger (training set doesn't easily add more data points) or smaller (admits entries more easily), although a good value is usually in the order of 10^{-4}.

If a data point is to be added to $\mathcal{BV}$ and the maximum number of data points p_{max} has been reached, then, the deletion procedure can be performed in two manners:

- OL method, in which the oldest data point is deleted, which grants set $\mathcal{BV}$ temporal locality. In this manner, $\mathcal{BV}$ retains a set of data points more suited to the states closest to the current one, but might lose data points corresponding to states further from the current one, which may damage the potential global scale of the GP. This is the fastest method of the both.
- KL method, in which the data point whose deletion results in the least Kullback-Leibler (KL) divergence between the complete training set and the set that results from deleting said data point is removed. This allows one to retain the set of data points with more information gain, thus diversifying the set $\mathcal{BV}$ the most and allowing the model to hold on to global information. In order to do this, the data point to remove is the one corresponding to the index i that obtains the smallest score measure given by:

$$\epsilon_i = \frac{|K^{-1}_{Z_\tau} k_{z_{\tau+1}}|_i}{diag(K^{-1}_{Z_\tau})_i}. \tag{19}$$

Since the modeling uncertainty cannot be directly measured, measured output $y_{\tau+1}$ will be estimated via equation (3) by obtaining an estimate of the state acceleration $\dot{\hat{x}}_{2_{\tau+1}}$ using a Kalman filter and the previous ν, such that:

$$y_{\tau+1} = \dot{\hat{x}}_{2_{\tau+1}} - \nu_\tau. \tag{20}$$

Algorithm 1 summarizes the general GP-MRAC procedure. This scheme will be applied to an stochastic differential equation in the following section, using both deletion methods. While the implementation stated in this paper leads to accurate estimates, the update process is computationally expensive and might become unfeasible if kept as is for real-time processors. In order to address this, the matrix updates can be performed in a faster manner if they are sparse, in which case, the methods explained in [6] greatly reduce the complexity of the required inversions. Furthermore, the inversion required for the Z_τ matrix can be performed via a rank-1 inversion if only one entry is changed.

IV. WING ROCK DYNAMICS AND SIMULATION SETUP

Wing rock is the name given to damped oscillations in roll that occur during landing operations of highly swept-back or delta wing aircraft. The dynamics of this phenomenom are highly non-linear and can be represented by the following dynamic equations:

$$\dot{\theta} = p,$$

Algorithm 1: Gaussian Process - Model Reference Adaptive Control

while *new measurements z and y are available* **do**
  Given $z_{\tau+1}$, compute $\gamma_{\tau+1}$ using (18);
  Estimate $\dot{\hat{x}}_{2_{\tau+1}}$ using a Kalman Filter;
  Compute $y_{\tau+1} = \dot{\hat{x}}_{2_{\tau+1}} - \nu_\tau$ as an estimate of model uncertainty;
  if $y_{\tau+1} > \epsilon_{tol}$ **then**
    if *size($\mathcal{BV}$) == p_{max}* **then**
      Remove an element from $\mathcal{BV}$ using either the OL or the KL method, described in Section III-B;
    end
    Add $y_{\tau+1}$ and $z_{\tau+1}$ to set $\mathcal{BV}$;
  end
  Calculate $\hat{m}_{\tau+1}$ and $\hat{\Sigma}_{\tau+1}$;
  Set $\nu_{ad} = \hat{m}_{\tau+1}$;
  Calculate pseudo-control input $\nu_{\tau+1}$ using (6);
  Calculate control input δ using (2);
end

$$\dot{p} = L_{\delta_a}\delta_a + \Delta(x). \tag{21}$$

in which the state x is represented by the angle θ and angular velocity $p = \dot{\theta}$, $L_{\delta_a} = 3$, the aileron control input is denoted as δ_a and $\Delta(x)$ denotes the stochastic modelling error brought about by the wing rock dynamics, such that $\Delta(x) \sim \mathcal{N}(\overline{\Delta}(x), \omega_n^2)$, in which $\overline{\Delta}(x)$ represents a mean function such that:

$$\overline{\Delta}(x) = W_0^* + W_1^*\theta + W_2^* p + W_3^*|\theta|p + W_4^*|p|p + W_5^*\theta^3, \tag{22}$$

whose parameters are the same as in [3] $W_0^* = 0.8$, $W_1^* = 0.2314$, $W_2^* = 0.6918$, $W_3^* = -0.6245$, $W_4^* = 0.0095$ and $W_5^* = 0.0216$. It should be noticed that a bias term representing trim error W_0^* is included in this model.

A. Simulation details

Since the proposed dynamics are described by a stochastic differential equation (SDE), the Euler Maruyama scheme will be used for the numerical integration of the model. Using $Itô\ Calculus$ [7], the formula must be given the following shape:

$$dY_t = b(t, Y_t)dt + h(t, Y_t)dW_t, \tag{23}$$

in which dW_t represents a Wiener process. Given the dynamics described by (21) and (22), the resulting SDE is the following:

$$dx = \begin{bmatrix} x_2 \\ L_{\delta_a}\delta_a + \overline{\Delta}(x) \end{bmatrix} dt + \begin{bmatrix} 0 \\ \omega_n \end{bmatrix} dW_t. \tag{24}$$

Through Euler-Maruyama, at each time step of Δt, the model state will be updated in the following manner:

$$x_{n+1} = x_n + \begin{bmatrix} \{x_2\}_n \\ L_{\delta_a}\delta_{a_n} + \overline{\Delta}(x_n) \end{bmatrix} \Delta t + \begin{bmatrix} 0 \\ \omega_n \end{bmatrix} \Delta W_n, \tag{25}$$

in which $\Delta W_n \sim \mathcal{N}(0, \Delta t)$.

B. Implementation parameters

The inversion model to be used will be $\nu = L_{\delta_a}\delta_a$, which will allow ν_{ad} to directly oppose measurement uncertainties. Furthermore, a second order reference model with natural frequency $w_{rm} = 1$ and damping ratio $\xi_{rm} = 0.5$ will be used, such that

$$\dot{x}_{rm} = \begin{bmatrix} 0 & 1 \\ -w_{rm}^2 & -2\xi_{rm}w_{rm} \end{bmatrix} x_{rm} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} r, \tag{26}$$

in which r will be used to create disturbances that make the model stray away from the equilibrium point in which $\theta = \dot{\theta} = 0$. A timestep of $\Delta t = 0.05$ seconds will be enforced on the simulation as well. Also, the linear feedback gains for all control schemes will be $K_1 = K_2 = 10$ (which will allow acceptable desired state tracking) and the variance of the uncertainty model will be set as $\omega_n = 0.01$. For the RBFN implementation, 121 centers will be evenly distributed in a domain of $[-2, 2] \times [-2, 2]$, the width of all basis functions will be set to $\mu = 1$ and a learning rate constant of $\Gamma_W = 20$ will be used to enforce aggressive learning on the RBFN. For the GP schemes, a maximum number of data points $p_{max} = 100$ will be allowed and a tolerance of $\epsilon_{tol} = 10^{-9}$ will be enforced to accept a greater amount of new points and better appreciate the difference between the proposed deletion schemes. Finally, the width on the used Gaussian kernels will also be $\mu = 1$.

V. SIMULATION RESULTS

Figures 2, 3 and 4 show the results from the simulations explained in the previous section (as well as the mean squared errors (MSEs) obtained while tracking the desired states and the uncertainty), both the GP-MRAC schemes using a Kalman Filter for getting an approximation on model uncertainty. Due to the disturbances introduced in the reference model (red dotted line), the state strays away from the RBFN domain, as denoted near the uncertainty peaks, which the RBFN does not follow at all, probably due to how far off they are from the domain, thus resulting in poor uncertainty modelling and harming the MRAC tracking performance in both states. On the other hand, both GP-MRAC schemes were able to adjust to the introduced uncertainty reasonably well, resulting in improved state tracking. It should be noted that neither were able to completely adjust to perfectly at the peaks of $\Delta(x)$, presumably since not enough data is obtained from these peaks during the small time in which the states remain there.

From the MSEs from both GP-MRAC schemes with different deletion methods, one might be tempted into thinking that the OL deletion scheme performs better than the KL one (even though both offer tracking and estimation errors of the same order). However, this result might explained by

the Kalman Filter estimates. In order to clarify this, instead of estimating the model uncertainty, the actual value of $\Delta(x)$ was used as feedback for the GPR output measurements $y_{\tau+1}$. The plots obtained from tracking the reference model were similar to the ones features in figures 3 and 4, so they won't be analyzed. A more interesting analysis can be performed on the resulting uncertainty estimation plots,

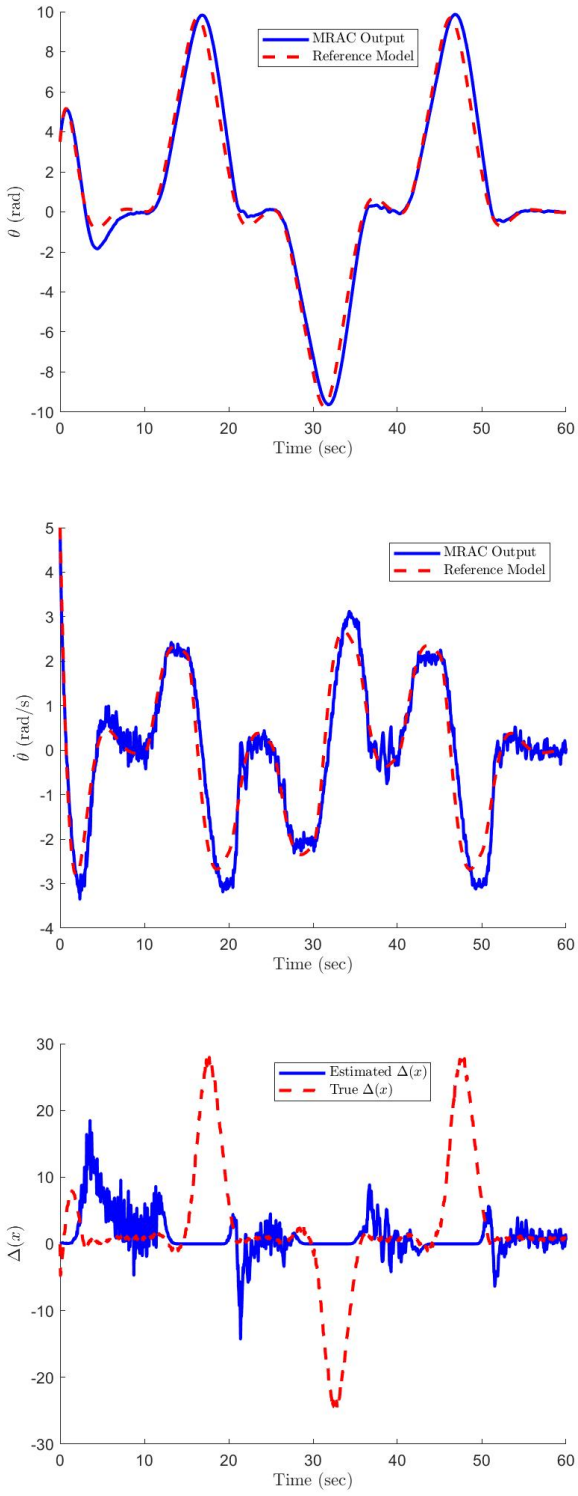

Fig. 2. Results for RBFN-MRAC. **[UP]:** Angle tracking θ (MSE = $4.5751 \cdot 10^{-1}$) **[CENTER]:** Angular velocity tracking $\dot{\theta}$ (MSE = $2.0111 \cdot 10^{-1}$) **[DOWN]:** Estimated uncertainty model $\Delta(x)$ (MSE = $9.226305 \cdot 10^{1}$)

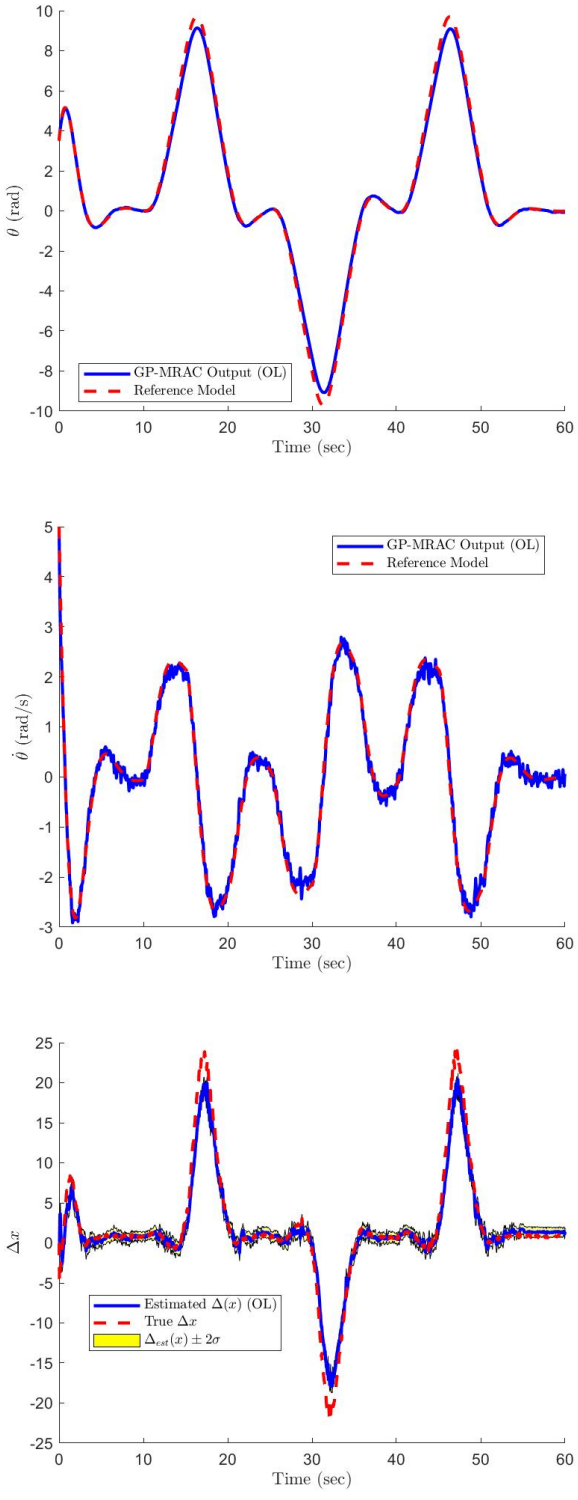

Fig. 3. Results for GP-MRAC using OL (oldest point) deletion scheme. **[UP]:** Angle tracking (MSE = $8.5543 \cdot 10^{-2}$) θ **[CENTER]:** Angular velocity tracking $\dot{\theta}$ (MSE = $1.8979 \cdot 10^{-2}$) **[DOWN]:** Estimated uncertainty model $\Delta(x)$ (MSE = 2.4657)

which are displayed in figure 5. Since both algorithms allow the determination of both a mean and a variance at each timestep, the yellow area around the estimated $\Delta(x)$ in both plots represents the region of confidence around which the actual $\Delta(x)$ can be found. While this area is fuzzy around the 0.8 bias in the OL case, the area remains more consistent around this value in the case of the KL method. This can be explained by the manner in which both algorithm delete data points. While the KL method aims to keep data point with unique features, and this keeps a consistent estimate around previously visited states, the OL method discards the oldest points without regard for how they affect the estimate, thus focusing on the current states rather than in obtaining a more global model. In this manner, while both offer similar results error-wise, the KL method presents itself as a more sophisticated approach to building the set $\mathcal{BV}$. Nonetheless, both methods offer advantages and neither should be discarded right away.

In the end, the algorithms that employed GP-MRAC were more capable of adapting to new states and didn't require previous knowledge of either the domain or the distribution of the model uncertainty to successfully diminish its effect on tracking performance.

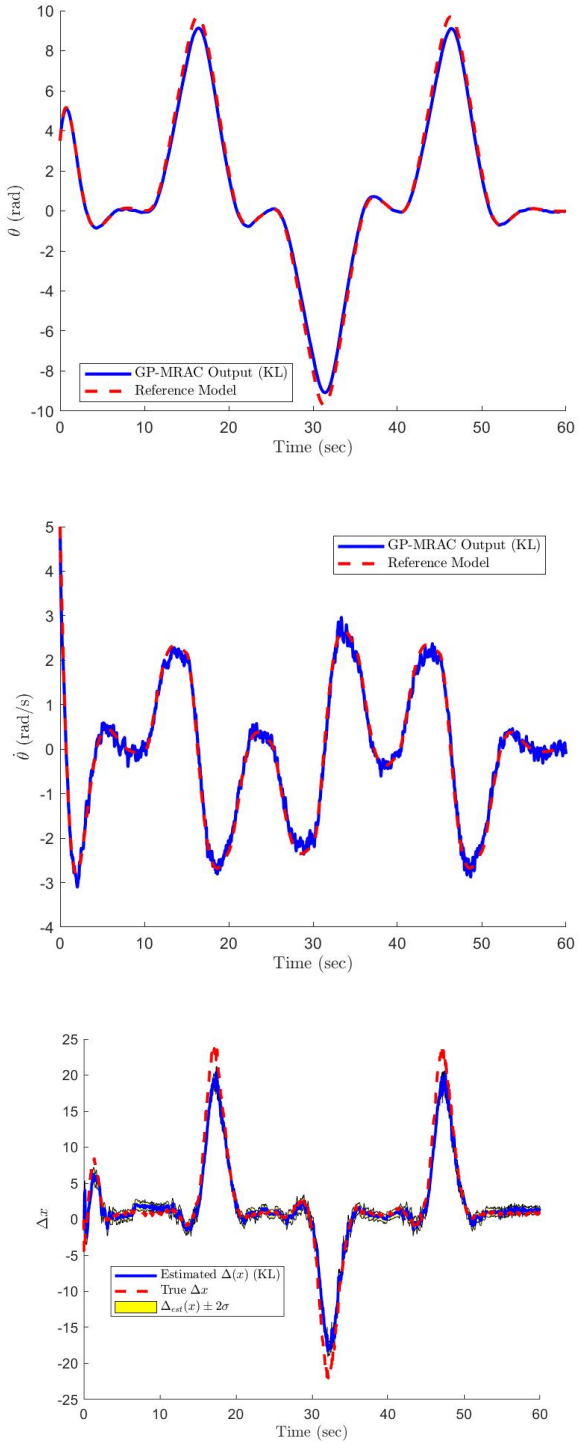

Fig. 4. Results for GP-MRAC using KL (KL divergence) deletion scheme. **[UP]:** Angle tracking θ (MSE = $9.235 \cdot 10^{-2}$) **[CENTER]:** Angular velocity tracking $\dot{\theta}$ (MSE = $2.0582 \cdot 10^{-2}$) **[DOWN]:** Estimated uncertainty model $\Delta(x)$ (MSE = 2.4987)

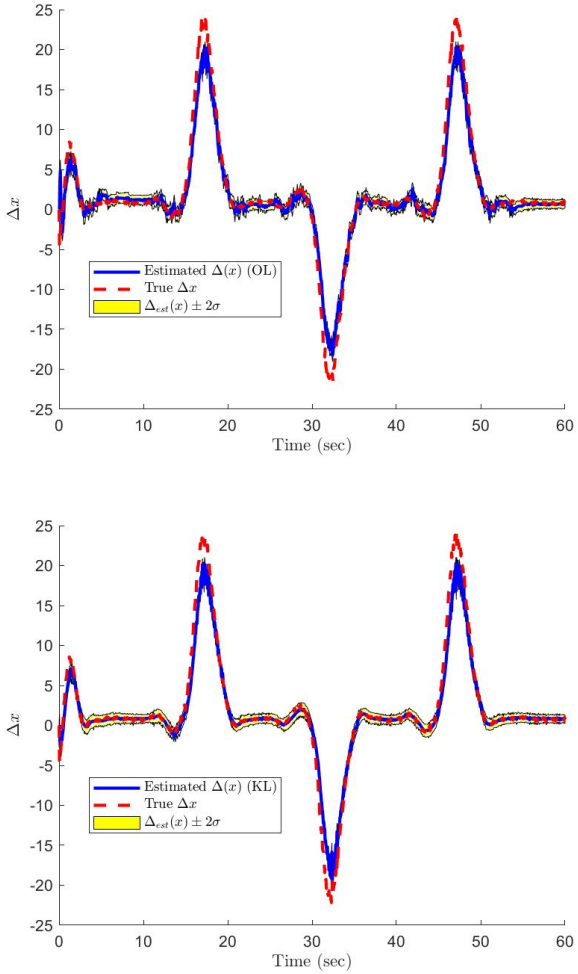

Fig. 5. Uncertainty modelling using true uncertainty $\Delta(x)$ feedback. **[UP]:** GP-MRAC using OL (oldest point) deletion scheme. (MSE = 2.5044) **[DOWN]:** GP-MRAC using KL (KL divergence) deletion scheme. (MSE = 2.3979)

VI. CONCLUSIONS

Gaussian process regression (GPR) is suitable for estimating model uncertainty within the AMI-MRAC scheme without requiring previous knowledge of the domain of operation or the sought after uncertainty, as shown by the results. However, for inline implementation, GPR requires procedures to determine when to add new data points to the training set and how to delete them when the maximum amount of points has been reached. While deleting the oldest data point (OL method) results in an uncertainty model that has temporal locality and results in a better performance if the data remains near a given set point, deletion based on KL divergence (KL method) attempts to retain data with unique features in order to retain global information about the model.

ACKNOWLEDGMENT

The author thanks Ph.D. Alex Gorodetsky for his assistance in setting up the simulation environment and understanding the inner workings of Gaussian Process Regression.

REFERENCES

[1] G. Chowdhary and E. Johnson, "Theory and flight-test validation of a concurrent-learning adaptive controller," *Journal of Guidance, Control, and Dynamics*, vol. 34, pp. 592–607, 3 2011.

[2] H. A. Kingravi, G. Chowdhary, P. A. Vela, and E. N. Johnson, "Reproducing kernel hilbert space approach for the online update of radial bases in neuro-adaptive control," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, pp. 1130–1141, July 2012.

[3] G. Chowdhary, H. A. Kingravi, J. P. How, and P. A. Vela, "Bayesian nonparametric adaptive control using gaussian processes," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, pp. 537–550, March 2015.

[4] M. Liu, G. Chowdhary, B. C. da Silva, S. Liu, and J. P. How, "Gaussian processes for learning and control: A tutorial with examples," *IEEE Control Systems Magazine*, vol. 38, pp. 53–86, Oct 2018.

[5] J. . Pomet and L. Praly, "Adaptive nonlinear regulation: estimation from the lyapunov equation," *IEEE Transactions on Automatic Control*, vol. 37, pp. 729–740, June 1992.

[6] L. Csató and M. Opper, "Sparse on-line gaussian processes," *Neural Comput.*, vol. 14, pp. 641–668, Mar. 2002.

[7] P. Kloeden and E. Platen, "Numerical solution of stochastic differential equations / peter e. kloeden, eckhard platen," *SERBIULA (sistema Librum 2.0)*, 12 2018.