Learn Fast, Forget Slow: Safe Predictive Learning Control for Systems With Unknown and Changing Dynamics Performing Repetitive Tasks

Christopher D. McKinnon and Angela P. Schoellig

***Abstract*—We present a control method for improved repetitive path following for a ground vehicle that is geared toward long-term operation, where the operating conditions can change over time and are initially unknown. We use weighted Bayesian linear regression (wBLR) to model the unknown dynamics, and show how this simple model is more accurate in both its estimate of the mean behavior and model uncertainty than Gaussian process regression and generalizes to novel operating conditions with little or no tuning. In addition, wBLR allows us to use *fast adaptation* and *long-term learning* in one unified framework to adapt quickly to new operating conditions and learn repetitive model errors over time. This comes with the added benefit of lower computational cost, longer look-ahead, and easier optimization when the model is used in a stochastic model-predictive controller (MPC). In order to fully capitalize on the long prediction horizons that are possible with this new approach, we use Tube MPC to reduce the growth of predicted uncertainty. We demonstrate the effectiveness of our approach in the experiment on a 900-kg ground robot showing results over 3.0 km of driving with both physical and artificial changes to the robot's dynamics. All of our experiments are conducted using a stereo camera for localization.**

***Index Terms*—Learning and adaptive systems, model learning for control, robot safety, field robots.**

I. Introduction

THIS letter presents a new probabilistic method for modelling robot dynamics geared towards stochastic Model Predictive Control (MPC) and repetitive path following tasks. The goal of our approach is to enable a robot to operate in challenging and changing environments with minimal expert input and prior knowledge of the operating conditions. Our study is motivated by our previous work with Gaussian Processes (GPs) on this topic [1] and an interest in deploying robots in a wide range of operating conditions. Our method requires the unknown part of the dynamics to be linear in a set of model parameters.

Manuscript received September 10, 2018; accepted February 4, 2019. Date of publication February 25, 2019; date of current version March 7, 2019. This letter was recommended for publication by Associate Editor S. Calinon and Editor D. Lee upon evaluation of the reviewers' comments. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC), in part by the Ontario Research Fund (ORF) through an Early Researcher Award, and in part by a Sloan Research Fellowship. *(Corresponding author: Christopher D. McKinnon.)*

The authors are with the Dynamic Systems Lab, University of Toronto Institute for Aerospace Studies, North York, ON M3H 5T6, Canada (e-mail: chris.mckinnon@mail.utoronto.ca; schoellig@utias.utoronto.ca).

Digital Object Identifier 10.1109/LRA.2019.2901638

Safe control methods have emerged as a way to guarantee that safety constraints (e.g. a bound on maximum path tracking error) are kept in the face of model errors. Having an accurate estimate of model error is of critical importance to the validity of these safety guarantees. In order to derive models for complex systems or systems operating in challenging operating conditions, researchers increasingly rely on tools from machine learning. In particular, probabilistic models are used since they provide a measure of model uncertainty which can naturally be used to derive an upper bound on model error. Two common methods for doing this are GP regression [1]–[3] and various forms of local linear regression [4]–[6].

In our previous work [1], we used GPs to learn the robot dynamics in a number of different operating conditions by leveraging experience gathered over multiple traverses of a path. However, we found that they have a number of limitations that make them difficult to apply in a wide range of operating conditions. First, they are computationally expensive, which limits the number of training points that can be used in the model for control [1]. This limits the region of the input space over which the GP is accurate. Second, using maximum likelihood optimization to identify hyperparameters offline did not always result in good closed loop performance. For this reason, we used a fixed set of hyperparameters which limited the range of operating conditions where the learning was effective. Third, given fixed hyperparameters, the GP assumes that the unknown dynamics are globally homoscedastic even though we only fit the model locally along the path. This further limits the effectiveness of a GP-based approach.

In this letter, we propose a new approach to address these limitations: we use weighted Bayesian Linear Regression (wBLR) to model part of the robot dynamics locally along the path (see Fig. 1). A wBLR model is computationally inexpensive to fit and evaluate. This enables us to use more previous experience to learn repetitive model errors and current experience to adapt quickly to novel operating conditions. We leverage the fact that we are doing a repetitive path following task and a predictive control strategy to efficiently partition past data for fitting our local model. Our approach does not otherwise depend on hyperparameters which, in addition to its relatively simple parametric form, makes it very data efficient and thus able to adapt quickly and reliably to new operating conditions. Finally, in a special case, wBLR can be designed so that it preserves convexity of the optimization problem solved as part of the MPC-based control

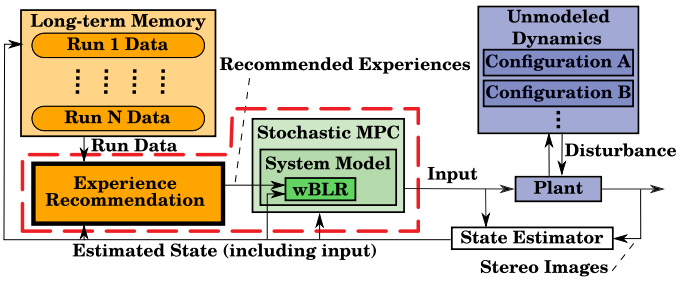

Fig. 1. Block diagram showing the proposed model learning method in closed-loop with a safe controller (red dashed box). The system dynamics can change from one run to another and over the course of a run. We use weighted Bayesian Linear Regression (wBLR) to learn the actuator dynamics of the plant. This approach, which enables fast adaptation and long-term learning, is shown to be highly effective in experiment. We encourage the reader to watch our video showing the experiments and datasets used in this letter: http://tiny.cc/fast-slow-learn.

strategy. As a result, improvements in the model translate well into improvements in control. In this letter, we also show how the model can be combined with Tube MPC to double the look-ahead horizon of our previous approach to three seconds.

II. Related Work

This work considers the problem of model learning for repetitive path following and a stochastic MPC. Recent work on this subject can be broadly grouped into three categories depending on how they group data to construct a model for robot dynamics.

First, single mode learning control. This class of methods learns a single model for the robot dynamics. This means that all data gathered by the robot can be grouped into one model and used to train any model parameters and validate them to avoid overfitting. This class of methods has shown impressive results control of ground robots [2], [7], quadrotors [6], manipulators [5] and humanoid robots [8]. This style of approach can learn new dynamics quickly, but if the robot dynamics can change due to a factor that is not included in the model (e.g. snow or wet ground changing the dynamics of ground robot) this class of methods only has the capacity to learn the robot dynamics in one such operating condition. It must either 'forget' all previous experience and adapt to the new operating condition from scratch or risk unsafe and sub-optimal behaviour due to model inaccuracy.

To address this, multi-modal learning methods learn a set of models to account for the dynamics in all operating conditions. The number of models in this set may be fixed or grow as new conditions are encountered during robot operation. This class of methods can still leverage all data accumulated in each operating condition to fit and validate complex models. This class of methods has shown impressive results in motion planning to avoid dynamic obstacles [9], repetitive path following [1], and legged robot locomotion [10] among others [11]–[13]. The main drawback of these methods is that they either assume the number of operating conditions is fixed, which presents similar limitations to the single mode methods, or, in the case of [1] which was performing repetitive path following, take one full traverse of the path to adapt to new operating conditions rather than adapting to new operating conditions over the course of a run. Adapting quickly to new operating conditions as they arise remains a challenge.

To bridge this gap, recent methods such as [14], [15] include both a complex model trained on lots of data with a simple online adaptation term to that can be updated quickly to adapt to new, previously unseen tasks. The simplicity of this online learning term enables fast adaptation to new conditions without worrying about overfitting or gathering sufficient data to do a complex model identification and validation. The long-term learning components, however, remain fixed and it is not clear how to update the long-term learning models efficiently. For example, [15] used a neural network trained on several hours of data and then fixed as the long-term learning component and linear regression updated recursively based on recent measurements to construct a 'fast adaptation' term that also captured the uncertainty in the robot dynamics. In this work, we propose a solution that couples a relatively simple model structure that can be adapted quickly to novel operating conditions with the ability to leverage lots of data gathered over many traverses of the path in various operating conditions. We use local models to achieve high performance with this relatively simple model form, and data weighting to incorporate the most relevant past data to improve from repeated traverses in similar operating conditions. This combines the long-term and fast adaptation components in one, unified, probabilistic framework.

In light of the current approaches and their limitations, the contributions of the letter are (i) to present a model learning framework that supports *fast adaptation*, *long-term learning*, and is tailored to predictive control; (ii) to incorporate that model (and its model uncertainty estimate) in a stochastic predictive control scheme; and (iii) to demonstrate the advantage of *fast adaptation* and *long-term learning* in path tracking experiments over challenging terrain.

III. Problem Statement

The goal of this work is to learn a probabilistic model for the dynamics of a ground robot performing a repetitive task, and show how it can be integrated with a state-of-the-art path following controller for high performance control while maintaining a quantitative measure of safety. The robot may be subjected to changes in its dynamics due to factors such as payload, terrain, or tyre pressure. We assume that these factors cannot be measured directly and all possible disturbances are not known ahead of time. A good algorithm should scale to long-term operation, take advantage of repeated runs in the same operating conditions, and adapt quickly to new operating conditions. The model must include a reasonable estimate of model uncertainty that acts as an upper bound on model error at all times.

We consider systems with dynamics of the form:

$$\mathbf{s}_{k+1} = \mathbf{s}_k + dt \overbrace{\mathbf{f}(\mathbf{s}_k, \boldsymbol{\xi}_k)}^{known}, \tag{1}$$

$$\boldsymbol{\xi}_{k+1} = \underbrace{\mathbf{g}^0(\boldsymbol{\xi}_k, \mathbf{u}_k)}_{known} + dt \underbrace{\mathbf{g}_k(\mathbf{x}_k)}_{unknown}, \tag{2}$$

where the state of the system $\mathbf{s}$ evolves according to known dynamics $\mathbf{f}(\cdot)$ that depend on $\mathbf{s}$ and the state of the actuators $\boldsymbol{\xi}$. We assume that our control input $\mathbf{u}$ affects the actuator dynamics which consist of a known part $\mathbf{g}^0(\cdot)$ and an unknown and potentially changing part $\mathbf{g}_k(\cdot)$ that we wish to learn. The unknown dynamics depend on a feature vector $\mathbf{x}$ that may be, for example, composed of $\boldsymbol{\xi}$ and $\mathbf{u}$ or nonlinear functions of these depending on prior knowledge about the system. The subscript refers to the timestep and dt is the duration of a timestep.

The system is constrained by state and input constraints. Let $\mathbf{z}_k = [\mathbf{s}_k^T, \boldsymbol{\xi}_k^T]^T$. Then:

$$\mathbf{z}_k \in \mathcal{S}, \mathbf{u}_k \in \mathcal{U}. \tag{3}$$

We assume a Gaussian belief over the state at each time step and enforce constraints probabilistically using a chance-constrained formulation so that the probability of violating state and input constraints is kept below an acceptable threshold. Since enforcing these constraints jointly can lead to undesirable, conservative behaviour, we enforce them individually, see [2] for a detailed explanation.

IV. Methodology

In this section, we present our approach for long-term, safe learning control with fast adaptation. Our approach makes extensive use of wBLR to model the system dynamics. We assume a known nonlinear model for the plant with unknown actuator dynamics that are linear in a set of model parameters. We use wBLR to determine the model parameters and a measure of run similarity to determine the data weights. This allows us to compute the posterior for the model parameters in closed form, avoiding iterative approaches such as [5], which also optimizes the data weights. We then formulate the control problem as a Tube MPC problem following work in [2], [16] but using a modified ancillary controller.

A. Weighted Bayesian Linear Regression

In this section, we give a brief overview of wBLR, which is used to learn the actuator dynamics, $\mathbf{g}_k(\cdot)$. It is an extension of Bayesian linear regression (BLR), as presented in [17], and a modification of [5], where we assume a data weighting is obtained in a separate step.

We consider each dimension of $\mathbf{g}_k(\cdot)$ separately. For this section, we will refer to a single dimension of $\mathbf{g}_k(\cdot)$ as $g(\cdot)$. For a given $\mathbf{x}_k$ the corresponding sample for $g(\mathbf{x}_k)$, denoted as g_k, may be calculated as $g_k = (\xi_{k+1} - g^0(\boldsymbol{\xi}_k, \mathbf{u}_k))/dt$, where ξ_{k+1} and $g^0(\cdot)$ are the relevant dimensions of $\boldsymbol{\xi}_{k+1}$ and $\mathbf{g}^0(\cdot)$, respectively.

Suppose we are given a *weighted* dataset $\mathcal{D}^l = \{\mathbf{x}_i, g_i, l_i\}_{i=1}^n$ with scalar weights $l_i \in [0, 1]$ that determine the importance of each data point. If $l_i = 0$, the point has no influence on the regression, and if $l_i = 1$, the point is fully included. In a simple scenario, all weights can be set to 1, in which case we recover regular BLR. We assume that the dynamics of interest depend on a vector of model parameters $\mathbf{w}$ and are of the form

$$g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + \eta, \tag{4}$$

where $\eta \sim \mathcal{N}(0, \sigma^2)$. The goal of wBLR is to determine the distribution for $\mathbf{w}$ and σ^2 given $\mathcal{D}^l$.

We start by assuming that each data point is independent and weight the contribution of each point as follows:

$$p(\mathbf{g} \mid \mathbf{X}, \mathbf{w}, \sigma^2) = \prod_{i=1}^{n} \mathcal{N}(g_i \mid \mathbf{w}^T \mathbf{x}_i, \sigma^2)^{l_i}, \tag{5}$$

where $\mathbf{g}$ is a vector of stacked g_i, and $\mathbf{X}$ is a matrix with rows $\mathbf{x}_i^T$. The intuition is one point raised to $l_i = 2$ would have the same contribution as two identical points and two identical points with $l_i = 0.5$ would have the same contribution as one data point. To avoid over-confident estimates, we restrict $l_i \in [0, 1]$. With this likelihood, the conjugate prior is a Normal Inverse Gamma (NIG) distribution [17] which gives us the following priors for $\mathbf{w}$ and σ^2:

$$p(\mathbf{w}|\sigma^2) \sim \mathcal{N}(\mathbf{w} \mid \mathbf{w}_0, \sigma^2 \mathbf{V}_0), \tag{6}$$

$$p(\sigma^2) \sim IG(\sigma^2 \mid a_0, b_0), \tag{7}$$

where $\mathbf{w}_0$ is the prior mean for the weights, $\mathbf{V}_0$ is a prior inverse sum of squares of $\mathbf{x}$, and a_0 and b_0 are the parameters of the Inverse Gamma distribution, which are proportional to the effective number of data points in the prior and a_0 times the prior output variance.

The likelihood, (5), can be manipulated into a NIG distribution over $\mathbf{w}, \sigma^2$ so that (6) and (7) form a conjugate prior and the posterior joint distribution over $\mathbf{w}$ and σ^2 is:

$$p(\mathbf{w}, \sigma^2|\mathcal{D}^l) = NIG(\mathbf{w}, \sigma^2|\mathbf{w}_N, \mathbf{V}_N, a_N, b_N) \tag{8}$$

$$\triangleq \mathcal{N}(\mathbf{w} \mid \mathbf{w}_N, \sigma^2 \mathbf{V}_N) IG(\sigma^2 \mid a_N, b_N), \tag{9}$$

where,

$$\mathbf{w}_N = \mathbf{V}_N(\mathbf{V}_0^{-1}\mathbf{w}_0 + \mathbf{X}^T \mathbf{L}\mathbf{g}), \tag{10}$$

$$\mathbf{V}_N = (\mathbf{V}_0^{-1} + \mathbf{X}^T \mathbf{L}\mathbf{X})^{-1}, \tag{11}$$

$$a_N = a_0 + tr(\mathbf{L})/2, \tag{12}$$

$$b_N = b_0 + \frac{1}{2}(\mathbf{w}_0^T \mathbf{V}_0^{-1} \mathbf{w}_0 + \mathbf{g}^T \mathbf{L}\mathbf{g} - \mathbf{w}_N^T \mathbf{V}_N^{-1} \mathbf{w}_N), \tag{13}$$

where $tr(\cdot)$ is the trace operator and $\mathbf{L}$ is a diagonal matrix of the data weights l_i. The posterior marginals are then:

$$p(\sigma^2|\mathcal{D}^l) = IG(\sigma^2 \mid a_N, b_N), \tag{14}$$

$$p(\mathbf{w}|\mathcal{D}^l) = \mathcal{T}(\mathbf{w} \mid \mathbf{w}_N, \frac{b_N}{a_N} \mathbf{V}_N, 2a_N) \tag{15}$$

where $\mathcal{T}$ is a Student t distribution. This gives us all of the components we need to make predictions of the state at future timesteps. It is important to note that while the uncertainty in σ^2 decreases as more data is added, the mean value for σ^2 can increase or decrease to reflect the data. The model uncertainty is then passed to the controller. This is in contrast to a GP (with fixed hyperparameters) where the uncertainty only decreases to a value determined by the hyperparameters as data is added. While it is possible to update the hyperparameters for a GP online, this is a computationally expensive operation that scales poorly with the size of the dataset and validating hyperparameters on a sufficiently large dataset is important to avoid overfitting.

1) Recursive Updates: When dealing with streaming data such as the data generated by a robot driving, it can be useful to continually update the model with recent data in order to adapt quickly to new scenarios. To do this while ensuring the model stays flexible enough to adapt to sudden changes, we recursively update the prior parameters while keeping the strength of the prior fixed at a pre-determined value n_0. The value of n_0 determines how many effective data points we attribute to the prior. A large value for n_0 results in smoother estimates for the $\mathbf{w}$ and σ^2 while a smaller value for n_0 allows them to vary more quickly. If we start with fewer than n_0 points in the prior, e.g. $a_0 < n_0/2$, we update the prior using (10)–(13) with the weight for the new point set to one, and set the posterior parameters to the prior for the next timestep. Once a_0 reaches $n_0/2$, we use (10)–(13) with the weight for the new point set to one and then use the following re-weighting to keep n_0 constant:

$$\mathbf{V}_{0^*} = \frac{n_0+1}{n_0}\mathbf{V}_N, \qquad \mathbf{w}_{0^*} = \mathbf{w}_N, \tag{16}$$

$$a_{0^*} = \frac{n_0}{n_0+1}a_N, \qquad b_{0^*} = \frac{n_0}{n_0+1}b_N. \tag{17}$$

The parameters $(\cdot)_{0^*}$ are the re-weighted parameters which become the new prior. This is equivalent to assigning the prior *and* the new point a weight of $n_0/(n_0+1)$ and carrying out a weighted update using (10)-(13). Compared to GPs, this gives us more control on how fast the model adapts. For a GP, a new point must either displace an existing one if the model has fixed size or increase the model size, which increases the computational cost of the model and will make it less flexible over time as more points are added. For wBLR, the influence of old data decreases after each re-weighting. The rate at which this happens depends on n_0, which is a parameter of our choosing and does not affect the computational cost of the model.

2) Preserving Convexity for MPC: MPC usually uses a gradient-based solver to compute the optimal control sequence efficiently. It is therefore desirable to maintain properties such as convexity in the optimization problem. Suppose that the MPC optimization problem is convex to begin with (e.g. the objective and inequality constraints are convex and $\mathbf{f}(\cdot)$ and $\mathbf{g}^0(\cdot)$ are affine). Then, if $\mathbf{g}_k(\cdot)$ is affine in $\mathbf{x}_k$, the new optimization problem will be convex for any choice of $\mathbf{w}$. See [18, Sec. 4.2].

B. Data Management

The purpose of our method is to construct the best possible model of the system dynamics for MPC. MPC uses the dynamics over the upcoming section of the path to compute the control input. Referring to Fig. 2, we use data from the *recent section path* to determine the weights that indicate which runs are most similar to the current run. Given these weights, we use data over the *upcoming* section of path (determined by the MPC look-ahead horizon) to construct a predictive model for the robot dynamics using wBLR. We use two mechanisms to adapt quickly to new scenarios and take advantage of repeated traverses in similar conditions.

1) Fast Adaptation: In order to adapt quickly to new scenarios, we use the most recent data pair $\{g_i, \mathbf{x}_i\}$ generated by the robot to update the model at every timestep. We use the *recursive update* explained in the previous section. These parameters are used as the prior at each timestep. In our previous work [1], the model reverted to a conservative form when the current dynamics did not match the dynamics in any previous run. While this preserved safety, it took one traverse of the path before the robot could adapt to new conditions. The approach presented in this letter enables the robot to adapt to new conditions as they arise, which is demonstrated in Section VI-C.

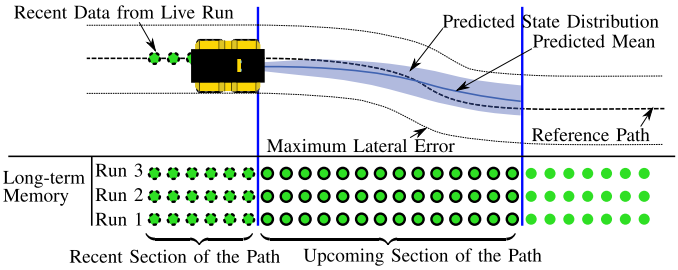

Fig. 2. The predicted trajectory (shaded blue) is shown superimposed over the reference path in parallel with the storage structure for data from previous runs (green circles) that is indexed by run and location along the path. Data along the recent section of the path (circles with dotted outlines) is used to estimate the similarity between the current run and each previous run. This similarity is used to weight data from the upcoming section of the path (circles with solid outlines) and construct the predictive model used in MPC. We also use recent data from the current run to recursively update the model and adapt quickly to novel operating conditions and non-repetitive changes. The size of the regions of the path considered *upcoming* and *recent* may be considered hyperparameters that are linked to the MPC problem.

2) Long-Term Learning: To improve controller performance in the face of repetitive changes, we leverage data from previous runs in similar operating conditions. We consider data from all previous runs because the model update is efficient and the cost to evaluate the model does not depend on the number of points used to construct it. Let $\mathcal{D}_j^-$ be data from previous run j over the *recent section of the path* (see Fig. 2) and $\hat{m}_j^-$ be a model constructed from $\mathcal{D}_j^-$. Let $(\cdot)_{i,j}$ refer to point i in run j and let n be the current run.

a) Outlier rejection: First, we check whether using data from each previous run is likely to result in model errors that violate the assumptions of the safe controller. Namely that a given percentile of model uncertainty is a reasonable upper bound for model error. For each previous run, we use $\hat{m}_j^-$ to generate predictions for the mean and variance corresponding to each $\mathbf{x}_{i,n}$ in recent data from the current run. We then compute the Z-score for each prediction given the associated measurement $g_{i,n}$ and compare this to the Z-score associated with the percentile of model error used as an upper bound in MPC (e.g. a Z-score of 2 for the 95th percentile). If the proportion of points outside of this threshold is higher than would be expected by chance (using the binomial test), we reject the run from further consideration. See [1] for details.

b) Weighted model update: Now that we have identified runs that will produce a model with valid confidence intervals (for safety), we weight data from each run according to its similarity to the current run (for performance). We compute the posterior probability of model $\hat{m}_j^-$ using:

$$p(\hat{m}_j^-|\mathcal{D}_n^-) \propto p(\mathcal{D}_n^-|\hat{m}_j^-)p(\hat{m}_j^-). \tag{18}$$

The first term on the right is the likelihood of recent data given model $\hat{m}_j^-$. The second term on the right is the prior, which we assume to be equal for all runs; however, it could be informed by other sources such as computer vision, a weather report, or user input. Similar to our previous work [1], we reject any run that has lower probability than the prior of generating $\mathcal{D}_n^-$. This is to ensure that experience added is likely to improve the performance beyond what could be achieved with no additional experience.

To update the parameters of the predictive model, we collect data from each previous run over the *upcoming section of the path* and weight each point in run j by $l_{i,j} = p(\mathcal{D}_n^-|\hat{m}_j^-)/p(\mathcal{D}_n^-|\hat{m}_{j^*}^-)$, $i = 1..n_j^+$ where n_j^+ is the number of points in run j over the *upcoming section of the path* and j^* is the run with maximum posterior probability. This satisfies $l_{i,j} \in [0, 1]$ and means that the effective number of points can increase with each additional run.

With these weights, we use (10)–(13) to compute the posterior parameters of the predictive model. This update (based on data from previous runs) is considered to be location specific and therefore discarded after computing the control; that is, the recursively updated prior becomes the prior for the next timestep.

C. Path Following MPC Controller Design

This section outlines our MPC formulation including the path parametrization, cost function, ancillary control design, and uncertainty propagation. We use a Model Predictive Contouring approach, based on [16], which expresses position error as lag error (parallel to the path) and contouring error (perpendicular to the path) and uses a virtual input to drive reference states along the path.

1) Uncertainty Propagation: We assume a Gaussian belief over the state at each time step and nonlinear dynamics for the plant. This allows us to use the Extended Kalman Filter (EKF) prediction equations to propagate our belief of the state into the future given a series of inputs [2]. We include uncertainty in the full state $\mathbf{z} = [\mathbf{s}^T, \boldsymbol{\xi}^T]^T$, the actuator model parameters $\mathbf{w}$, and the actuator model offset $\boldsymbol{\eta}$. Let $\mathbf{h}(\cdot)$ be the combined dynamics model (1) and (2) and $\mathbf{A}$ be the Jacobian of $\mathbf{h}(\cdot)$ with respect to the stacked full state and parameters, $\mathbf{A} = [\mathbf{A}_\mathbf{z}, \mathbf{A}_\mathbf{w}]$. The mean $\bar{\mathbf{z}}_k$ and covariance $\boldsymbol{\Sigma}_k^{\mathbf{zz}}$ can be updated using:

$$\bar{\mathbf{z}}_{k+1} = \mathbf{h}(\bar{\mathbf{z}}_k, \mathbf{u}_k), \tag{19}$$

$$\boldsymbol{\Sigma}_{k+1}^{\mathbf{zz}} = \mathbf{A}\mathbf{P}_k\mathbf{A}^T + \mathbf{Q}_k, \tag{20}$$

$$\mathbf{P}_k = \begin{bmatrix} \boldsymbol{\Sigma}_k^{\mathbf{zz}} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_k^{\mathbf{ww}} \end{bmatrix}, \tag{21}$$

where $\boldsymbol{\Sigma}_k^{\mathbf{ww}}$ is a block-diagonal matrix containing the model weight covariance matrix from (15) for each dimension of $\mathbf{g}_k(\cdot)$, $\mathbf{Q}_k$ is the process noise covariance, and $\mathbf{u}_k$ comes from MPC. The only non-zero components in $\mathbf{Q}_k$ are the diagonal elements corresponding to uncertainty in the output of the actuators for which we use the posterior mean of the variance from (14). In this framework, we can include uncertainty in the evolution of the model parameters $\mathbf{w}$ by modelling their dynamics as a random walk. In this work, we consider them to be fixed at the posterior estimate over the lookahead horizon.

The predicted uncertainty can be used to compute a confidence set around the mean prediction that the true system is guaranteed to lie within with high probability.

2) Ancillary State Feedback Controller: The method for uncertainty propagation in Section IV-C1 but does not take into account the fact that the controller can take corrective actions to reduce the predicted uncertainty [2]. The result is that the predicted uncertainty can grow quickly and without bound resulting in conservative control actions [2]. A common approach to account for feedback when predicting uncertainty is to use Tube MPC [19] and use an ancillary controller in the predictive model that drives the state towards the predicted mean [2].

In contrast to other approaches for tube MPC for non-linear systems, we make use of the fact that our actuator dynamics are linear to design linear ancillary controllers for these states. This keeps the uncertainty in these states bounded, which limits the uncertainty *growth* in other states over the prediction horizon. Section V-B shows how we apply this to a unicycle-type robot.

3) Constraint Tightening: Since our predictive model has uncertainty, we must tighten the constraints on the state and input to make sure the true system respects the true constraints (with high probability), and that the ancillary control policy remains feasible for our choice of the inputs. Our treatment of the constraint tightening follows [2]. For contouring error e^c, our chance constraints are:

$$p(e_k^c \leq e^{c,max}) \geq 1 - \epsilon_c \tag{22}$$

$$\Leftrightarrow e_k^c + r^c\sqrt{(\mathbf{t}_k^\perp)^T\boldsymbol{\Sigma}_k^{\mathbf{zz}}\mathbf{t}_k^\perp} \leq e^{c,max}, \tag{23}$$

where r^c is the quantile of the Gaussian CDF corresponding to the small probability of violating the contouring constraint ϵ_c (e.g. 2.0 for $\epsilon_c = 0.05$) [2], and $\mathbf{t}_k^\perp$ is a unit vector perpendicular to the path at time k. Other constraints on the state may be treated analogously.

Analogous treatment of the input constraints yields:

$$p(u_k^{[i]} < u^{[i],max}) \leq 1 - \epsilon_{u^{[i]}} \tag{24}$$

$$\Leftrightarrow u_k^{[i]} + r^{u^{[i]}}K_{u^{[i]}}\sqrt{(\sigma_k^e)^2} \leq u^{[i],max}, \tag{25}$$

where $u^{[i]}$ is the ith of $\mathbf{u}$, $K_{u^{[i]}}$ is an associated ancillary gain which acts on an error of our choosing, e, and σ^e is the standard deviation associated with that error. Here, we can see that while the ancillary controller reduces the prediction uncertainty it will also reduce the control input available for controlling the nominal state.

The feedback gain can be chosen as an infinite horizon LQR controller with the same cost function as MPC [2], [7] or included in the optimization problem [20], but we found that a wide range of gains worked for our system so left the gain as a tuning parameter.

D. Optimal Control Problem

At each timestep, we wish to solve for the optimal states and inputs subject to a set of safety constraints derived from the model uncertainty, path tracking error and actuator

Fig. 3. Clearpath Grizzly in the *loaded* configuration traversing a gravel mound at a target speed of 2.0 m/s with the proposed algorithm.

constraints. The decision variable is $\boldsymbol{\nu}_H = [\mathbf{u}_0, \mathbf{z}_1, ...\mathbf{u}_{N-1}, \mathbf{z}_N]^T$. This leads to the following optimization problem:

$$\underset{\bar{\boldsymbol{\nu}}_H}{\text{minimize}} \quad J(\bar{\boldsymbol{\nu}}_H) \tag{26}$$

$$\text{subject to} \quad \bar{\mathbf{z}}_{k+i+1} = \mathbf{h}(\bar{\mathbf{z}}_{k+i}, \mathbf{u}_{k+i}, \mathbf{x}_{k+i}),\ i = 0..N-1, \tag{27}$$

$$p(\mathbf{z}_{k+i+1} \in \mathcal{S}) \geq 1 - \boldsymbol{\epsilon}^{\mathbf{z}},\ i = 0..N-1, \tag{28}$$

$$p(\mathbf{u}_i \in \mathcal{U}) \geq 1 - \boldsymbol{\epsilon}^{\mathbf{u}},\ i = 0..N-1, \tag{29}$$

where $\boldsymbol{\epsilon}^{(\cdot)}$ is a vector of small, acceptable probabilities of violating each state and input constraint, which must be solved at every timestep and $J(\cdot)$ is a quadratic cost that penalizes position, heading, and velocity error, and includes a smoothing term to avoid high frequency inputs. We use the mean of each random variable $\bar{(\cdot)}$ to approximate the expected cost and enforce the dynamics constraints.

V. Application to a Ground Robot

This section outlines how to apply our method to the unicycle ground robot pictured in Fig. 3.

A. Robot Model

Let $\mathbf{s} = [x, y, \theta]^T$, the 2D position and heading of the robot, $\boldsymbol{\xi} = [v, \omega]^T$, the speed and turn rate of the robot, and $\mathbf{u} = [v^{cmd}, \omega^{cmd}]^T$, the commanded speed and turn rate of the robot. We assume that the dynamics of $\mathbf{s}$ are well approximated by a unicycle

$$\underbrace{\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \end{bmatrix}}_{\mathbf{s}_{k+1}} = \underbrace{\begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix}}_{\mathbf{s}_k} + dt \underbrace{\begin{bmatrix} v_k \cos\theta_k \\ v_k \sin\theta_k \\ \omega_k \end{bmatrix}}_{\mathbf{f}(\cdot)}, \tag{30}$$

which is of the form (1). For wBLR, we will model the dynamics of $\boldsymbol{\xi}$ as

$$\underbrace{\begin{bmatrix} v_{k+1} \\ \omega_{k+1} \end{bmatrix}}_{\boldsymbol{\xi}_{k+1}} = \underbrace{\begin{bmatrix} v_k \\ \omega_k \end{bmatrix}}_{\mathbf{g}^0(\cdot)} + dt \underbrace{\begin{bmatrix} [v_k^{cmd}, v_k]\mathbf{w}_k^v + \eta_k^v \\ [\omega_k^{cmd}, \omega_k]\mathbf{w}_k^\omega + \eta_k^\omega \end{bmatrix}}_{\mathbf{g}_k(\cdot)}, \tag{31}$$

which is of the form (2).

B. Ancillary Control Design for the Unicycle With First Order Actuator Dynamics

The ancillary controller is meant to reduce uncertainty growth over the prediction horizon. For the unicycle, lateral uncertainty growth (which is constrained) depends on heading uncertainty and speed. Keeping uncertainty in these states low therefore keeps the lateral uncertainty low reducing the amount that the constraints are tightened (see (23)). With a linear feedback controller on the heading and speed error, the speed and turn rate dynamics become:

$$\begin{bmatrix} v_{k+1} \\ \omega_{k+1} \end{bmatrix} = \begin{bmatrix} v_k \\ \omega_k \end{bmatrix} + dt \begin{bmatrix} [v_k^{cmd} + K_v e_k^v, v_k]^T \mathbf{w}^v \\ [\omega_k^{cmd} + K_\theta e_k^\theta, \omega_k]^T \mathbf{w}^\omega \end{bmatrix} \tag{32}$$

where $e_k^{(\cdot)} = (\cdot)_k - \bar{(\cdot)}_k$ is the difference between the state $(\cdot)_k$ and the predicted mean at time step k. These controllers keep the system close to the predicted speed and heading.

VI. Experiments

Experiments were conducted on a 900 kg Clearpath Grizzly skid-steer ground robot shown in Fig. 3. First, we compare the predictive performance of a GP to our proposed method on a dataset with varied payload and terrain type. Second, we demonstrate the effectiveness of each component of our algorithm in closed loop. Finally, we demonstrate the path tracking performance of our algorithm at high speed on a 175 m off-road course.

A. Implementation

Our algorithm was implemented in C++ on an Intel i7 2.70 GHz 8 core processor with 16 GB of RAM. Our controller relies on a vision-based system, Visual Teach and Repeat [21], for localization, which runs on the same laptop. The controller runs at 10 Hz with a three second look-ahead discretized by 30 points. The optimization problem (26)-(29) is solved as a sequential quadratic program and re-linearized three times, taking an average of 70 ms to compute the control. The model updates (Sections IV-B1 and IV-B2) are executed at every time step.

We consider the last three seconds of data (30 samples) from the live run for $\mathcal{D}_n^-$. The penalties on lag, contouring, heading, speed, and turn rate error are 50, 200, 200, 2, and 2 respectively. The penalties on commanded speed, turn rate, and reference speed from their references are 1, 1, and 50 respectively. The penalties on rate of change of commands in the same order are 10, 15, and 5. The maximum lateral error is 2 m, r^c is 1, and the ancillary controller gains are both -5. The prior strength, n_0, was set to 100. For the high speed experiment, we increased the penalty on commanded turning acceleration from 15 to 20 to achieve smoother performance on the rough terrain.

B. Model Predictive Performance Comparison

In order to evaluate the suitability of the proposed method for predictive control, we evaluate the predictive performance of the proposed method (Sections IV-B1 and IV-B2) to a context-aware GP (c.f. [1], except we learn the actuator dynamics and

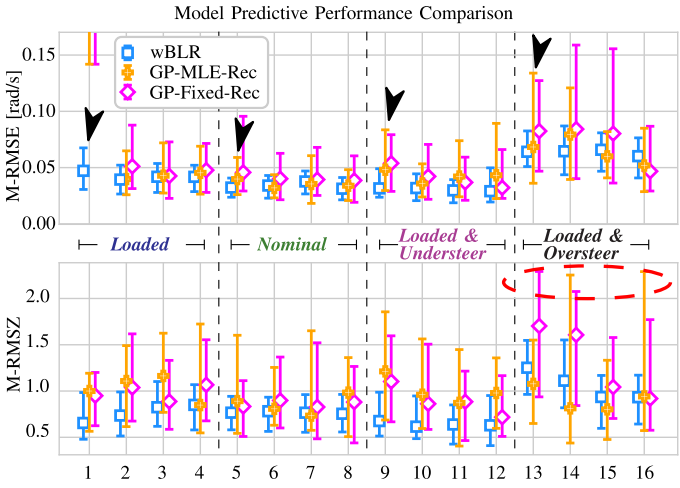

Fig. 4. A comparison of M-RMSE and M-RMSZ for the rotational dynamics with the vehicle in four different configurations for two context-aware GP-based methods and the proposed method. The error bars indicate the 25th and 75th percentiles and the marker indicates the median. The 65 m path traversed sand, gravel, and concrete. Runs 1-4 are in the *Loaded* configuration, with 6 gravel bags in the rear of the Grizzly (see Fig. 3), runs 6–8 are with the vehicle in the *Nominal* configuration (no modification), runs 9–12 are with the vehicle in the *Loaded & Understeer* configuration, where it is loaded and the turn rate commands are multiplied by 0.7, runs 13–16 are in the **Loaded & Oversteer** configuration, where the vehicle is loaded and the turn rate commands are multiplied by 1.2. The black arrows indicate the first time the vehicle is driven in a new operating condition. The red circle indicates where the GP-based methods were over-confident, frequently producing M-RMSZ values above 2.0.

not an additive model error) with fixed hyperparameters (GP-Fixed-Rec) and with hyperparmeters optimized using MLE and a sliding window of the last 100 datapoints (GP-MLE-Rec). We consider the rotational dynamics because they differ the most between configurations.

We compare the model predictions given the inputs that were actually applied to the vehicle over the MPC prediction horizon to the actual state of the vehicle recorded at the corresponding times. To measure the accuracy of the prediction of the mean, we use the Multi-Step RMS Error (M-RMSE) over this horizon. To measure the accuracy of the model uncertainty estimate, we use the Multi-Step RMS Z-score (M-RMSZ) over this horizon:

$$M-RMSZ_k = \sqrt{\frac{1}{H}\sum_{q=0}^{H-1}\left(\frac{\omega_{k+q+1}-\bar{\omega}(\mathbf{x}_{k+q})}{\sigma^{\omega}(\mathbf{x}_{k+q})}\right)^2}, \quad (33)$$

where $\bar{\omega}(\mathbf{x}_k)$ is the predicted mean value of ω_k given the predicted $\mathbf{x}_k$ and H is the number of timesteps in the prediction horizon. To generate the predictions, we use the controls inputs that were actually applied to the vehicle. An accurate model uncertainty estimate is important to ensure that the probability of violating the chance constraints formulated in Section IV-C3 is kept at an acceptable level, specified by $\boldsymbol{\epsilon}^{\mathbf{z}}$ and $\boldsymbol{\epsilon}^{\mathbf{u}}$. We consider an M-RMSZ between -0.5 and 1.5 to be acceptable. If this value exceeds 2.0, the model uncertainty estimate is overconfident which could lead to violation of the chance constraints.

Fig. 4 compares the proposed method to GP-Fixed-Rec and GP-MLE-Rec. The proposed method consistently achieves lower M-RMSE, especially during run 1 before the GP-based methods have data, and the first time the system encounters a

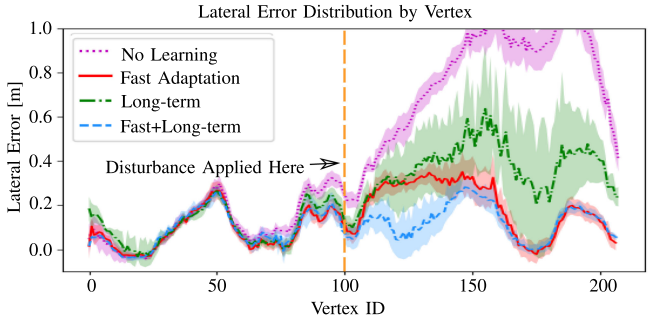

Fig. 5. This figure shows the closed-loop performance of the controller when we introduce a large, repetitive disturbance at vertex 100 by multiplying the turn rate commands by 0.5 after this point. This introduces a large, repeatable disturbance such as one might expect if the vehicle was traversing a patch of ice. The solid line indicates the median lateral tracking error over eight runs and the shaded region indicates the 50th and 75th percentiles. The proposed method with both long-term and fast adaptation learning achieves the lowest error and fastest convergence. No learning is when the controller uses a fixed wBLR model to compute the controls.

new configuration as indicated by the black arrows. This is the proposed method is able to incorporate relevant data from the current run using *fast adaptation*. While online hyperparameter optimization generally improves the M-RMSE, it causes the GP overfit in the most challenging scenario, ***Loaded & Oversteer***, which can be inferred by the M-RMSZ value exceeding 2.0 during runs 14 and 16. In contrast, the proposed method is much more consistent and the M-RMSZ stays between 0.5 and 1.5, indicating the model has a reasonable estimate of model uncertainty.

C. Closed Loop Tracking Performance Comparison

To demonstrate the impact of each component of our method in closed-loop and show that it can adapt to repetitive model errors, we drive the vehicle around two laps of a circular course and apply an artificial disturbance by multiplying the turn rate commands by 0.5 at the start of the second lap (vertex 100 in Fig. 5). Physically, this may be similar to the vehicle getting a flat tyre or losing power in one motor. We compare the tracking performance of each component of our algorithm over eight repeats of the path. For this experiment, the desired speed was 2 m/s.

Fig. 5 shows that all methods achieve similar performance before the disturbance is applied because the model for all methods was a good representation of the vehicle dynamics over this portion of the path. After this point, the non-learning controller incurs a large lateral error because the model is no longer accurate. Long-term learning (Section IV-B2) similarly incurs a large path tracking error on the first run (see Fig. 6) since there are no previous runs with experience. However, after the first run, it improves greatly but then converges slowly because it is constantly working against a static prior (the same model used for the non-learning comparison), that is incorrect after the disturbance is applied. When fast adaptation (Section IV-B1) is enabled, the controller incurs a large tracking error at the moment the disturbance is applied but adapts quickly to the new robot dynamics to achieve low error as expected. When

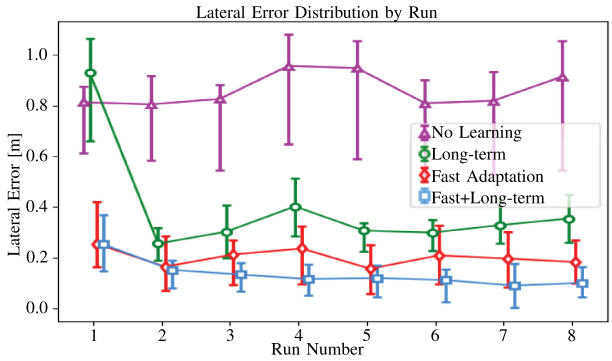

Fig. 6. Figure showing the 25th, 50th, and 75th percentiles of lateral error after the large, repetitive disturbance described in Section VI-C was applied. This figure shows that the proposed algorithm was able to quickly adapt to the disturbance and that the combination of fast adaptation (Section IV-B1) and long-term learning (Section IV-B2) achieves the best performance. No learning is when the controller uses a fixed, prior model to compute the controls. The horizontal position of each point is offset slightly for clarity.

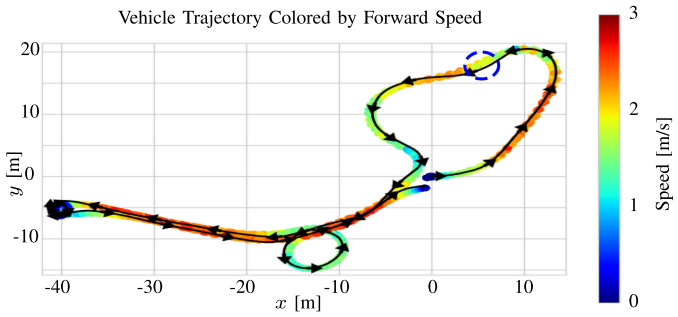

Fig. 7. This figure shows the path taken by the vehicle on five traverses of a 175 m course. The direction of travel is indicated by the black arrows. The maximum path tracking error is 0.7 m when the controller cuts a corner (dashed blue circle). The vehicle was in the $Nominal$ configuration.

both fast adaptation and long-term learning are enabled, the fast adaptation keeps the prior close to the true dynamics such that the long-term learning is able to reduce the transient error by leveraging data from the upcoming section of the path. This combination achieves the lowest path tracking error and the fastest convergence (see Fig. 6).

D. High Speed Tracking Performance

Finally, we evaluated the performance of our controller on a 175 m off-road course with tight turns and fast straights. The desired speed was 3 m/s and the controller achieved an average speed of 1.6 m/s with a top speed of 2.7 m/s and a RMS lateral error of 0.25 m. This is a 60% improvement over our previous work, where the controller achieved an average speed around 1.0 m/s on pavement [1].

VII. Conclusions

In this letter, we have proposed a new method for long-term, safe learning control based on local, weighted BLR. This method is computationally inexpensive which enables fast model updates and allows us to leverage large amounts of data gathered over previous traverses of a path. This enables both fast adaptation to new scenarios and high-accuracy tracking in the presence of repetitive model errors. The model parameters can be determined reliably online which enables our method to be applied in a wide range of operating conditions with little to no tuning. We have demonstrated the effectiveness of the proposed approach in a range of challenging, off-road experiments. We encourage the reader to watch our video at http://tiny.cc/fast-slow-learn showing the experiments and datasets used in this letter.

References

[1] C. McKinnon and A. P. Schoellig, "Experience-based model selection to enable long-term, safe control for repetitive tasks under changing conditions," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 2977–2984.

[2] L. Hewing and M. N. Zeilinger, "Cautious model predictive control using Gaussian process regression," 2017, *arXiv:1705.10702*.

[3] A. Akametalu, J. Fisac, J. Gillula, S. Kaynama, M. Zeilinger, and C. Tomlin, "Reachability-based safe learning with Gaussian processes," in *Proc. 53rd IEEE Conf. Decis. Control*, 2014, pp. 1424–1431.

[4] L. Jamone, B. Damas, and J. Santos-Victor, "Incremental learning of context-dependent dynamic internal models for robot control," in *Proc. IEEE Int. Symp. Intell. Control*, 2014, pp. 1336–1341.

[5] J. Ting, A. D'Souza, S. Vijayakumar, and S. Schaal, "A Bayesian approach to empirical local linearization for robotics," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2008, pp. 2860–2865.

[6] V. Desaraju, A. Spitzer, and N. Michael, "Experience-driven predictive control with robust constraint satisfaction under time-varying state uncertainty," in *Proc. Robot. Sci. Syst. Conf.*, 2017.

[7] Y. Gao, A. Gray, H. Tseng, and F. Borrelli, "A tube-based robust nonlinear predictive control approach to semiautonomous ground vehicles," *Veh. Syst. Dyn.*, vol. 52, no. 6, pp. 802–823, 2014.

[8] G. Sicard, C. Salan, S. Ivaldi, V. Padois, and O. Sigaud, "Learning the velocity kinematics of ICUB for model-based control: XCSF versus LWPR," in *Proc. 11th IEEE-RAS Int. Conf. Humanoid Robots*, 2011, pp. 570–575.

[9] G. Aoude, B. Luders, J. Joseph, N. Roy, and J. How, "Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns," *Auton. Robots*, vol. 35, no. 1, pp. 51–76, 2013.

[10] R. Pautrat, K. Chatzilygeroudis, and J. Mouret, "Bayesian optimization with automatic prior selection for data-efficient direct policy search," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2018, pp. 7571–7578.

[11] R. Calandra, S. Ivaldi, M. Deisenroth, E. Rueckert, and J. Peters, "Learning inverse dynamics models with contacts," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 3186–3191.

[12] B. Luders, I. Sugel, and J. How, "Robust trajectory planning for autonomous parafoils under wind uncertainty," in *Proc. AIAA Conf. Guid., Navigat. Control*, 2013, Paper AIAA 2013-4584.

[13] K. Jo, K. Chu, and M. Sunwoo, "Interacting multiple model filter-based sensor fusion of GPS with in-vehicle sensors for real-time vehicle positioning," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 1, pp. 329–343, Mar. 2012.

[14] K. Pereida, D. Kooijman, R. Duivenvoorden, and A. P. Schoellig, "Transfer learning for high-precision trajectory tracking through L1 adaptive feedback and iterative learning," *Int. J. Adaptive Control Signal Process.*, vol. 52, no. 6, pp. 802–823, 2018.

[15] J. Fu, S. Levine, and P. Abbeel, "One-shot learning of manipulation skills with online dynamics adaptation and neural network priors," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2016, pp. 4019–4026.

[16] D. Lam, C. Manzie, and M. Good, "Model predictive contouring control," in *Proc. IEEE Conf. Decis. Control*, 2010, pp. 6137–6142.

[17] K. Murphy, *Machine Learning: A Probabilistic Perspective*. Cambridge, MA, USA: MIT Press, 2012.

[18] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[19] A. Aswani, H. Gonzalez, S. Sastry, and C. Tomlin, "Provably safe and robust learning-based model predictive control," *Automatica*, vol. 49, no. 5, pp. 1216–1226, 2013.

[20] M. Vitus and C. Tomlin, "On feedback design and risk allocation in chance constrained control," in *Proc. 50th IEEE Conf. Decis. Control/Eur. Control Conf.*, 2011, pp. 734–739.

[21] M. Paton, F. Pomerleau, K. MacTavish, C. Ostafew, and T. Barfoot, "Expanding the limits of vision-based localization for long-term route-following autonomy," *J. Field Robot.*, vol. 34, no. 1, pp. 98–122, 2017.