ETH zürich

ETH Library

Contact Invariant Model Learning for Legged Robot Locomotion

Journal Article

Author(s):
Grandia, Ruben; Pardo, Diego; Buchli, Jonas

Publication date:
2018-07

Permanent link:
https://doi.org/10.3929/ethz-b-000249409

Rights / license:
In Copyright - Non-Commercial Use Permitted

Originally published in:
IEEE Robotics and Automation Letters 3(3), https://doi.org/10.1109/LRA.2018.2806566

This page was generated automatically upon download from the ETH Zurich Research Collection.
For more information, please consult the Terms of use.

Contact Invariant Model Learning for Legged Robot Locomotion

Ruben Grandia, Diego Pardo, and Jonas Buchli

***Abstract*—In this work we present a new formulation for learning the dynamics of legged robots performing locomotion tasks. Using sensor data we learn error terms at the level of rigid body dynamics and actuation dynamics. The learning framework deals with the hybrid nature of legged systems given by different contact configurations: We use the projection of the rigid body dynamics into a subspace consistent with the contact constraints. The equations of motion in such subspace do not depend on the contact forces, allowing to formulate a learning problem where force sensor data is not required. Additionally, we propose to use the columns of end-effector Jacobians as basis vectors, obtaining a model that generalizes across contact configurations. Both Locally Weighted Projection Regression and Sparse Gaussian Process Regression are used as supervised learning techniques. As application of the learned model, an inverse dynamics control method is extended. Hardware experiments with a quadruped robot show reduced RMS tracking error and a significant reduction in RMS feedback effort during base-only, walking, and trotting motions.**

***Index Terms*—Legged Robots, Dynamics, Model Learning for Control.**

I. Introduction

IMPLEMENTING model-based torque controllers on legged robots exposes the fact that analytical models often do not fully capture the real dynamics. As a consequence, high feedback gains are required, performance suffers, and in the worst case, instabilities occur. Improving the accuracy of the equations of motion increases the effectiveness of model-based controllers, state estimators, and dynamic motion planners.

Parameter identification methods have been extensively explored in robotics research and applied to legged systems [1]. However, due to the fixed basis functions, parametric models struggle to capture general unmodelled phenomena. In an effort to overcome this, it has been shown that general supervised learning techniques can effectively capture unmodelled dynamics for robotics [2]. Yet, to the best of our knowledge, such methods have not been applied to the case of legged robots. The presence of contact forces, velocity discontinuities, and a large state space pose additional challenges for model learning on these type of systems.

Manuscript received: September, 10, 2017; Revised December, 17, 2017; Accepted January, 21, 2018.

This paper was recommended for publication by Editor Nikos Tsagarakis upon evaluation of the Associate Editor and Reviewers' comments. This research has been funded through a Swiss National Science Foundation Professorship award to Jonas Buchli and the NCCR Robotics.

All authors are with the Agile & Dexterous Robotics Lab at the Institute of Robotics and Intelligent Systems, ETH Zurich, Switzerland. email: {rgrandia, depardo, buchlij}@ethz.ch

Digital Object Identifier (DOI): see top of this page.

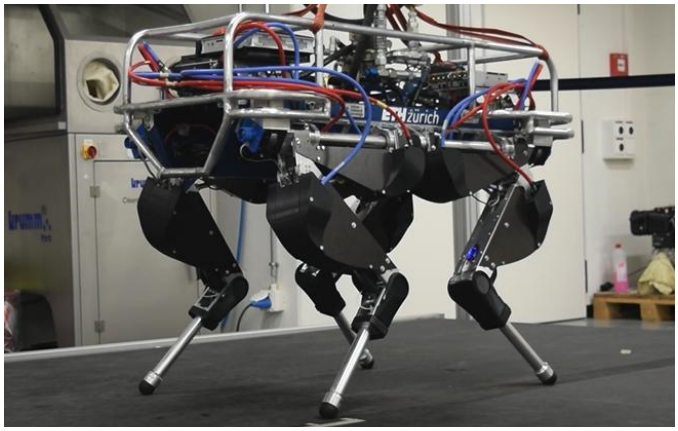


Fig. 1: The Hydraulically-actuated Quadruped (HyQ) robot, used for the experiments presented in this paper: base motions, walking, and trotting.

In that context, we envision a formulation that i) captures the general dynamics of legged robots beyond the Rigid Body Dynamics (RBD) assumption, ii) does not require contact force sensor measurements, and iii) is able to learn and predict the dynamics in a wide variety of motions and contact configurations.

We emphasize that the contribution of this work is not a new learning algorithm but rather a formulation that enables model error learning for legged robots. We follow a direct learning approach by learning a model from command/motion pairs. This is in contrast to indirect approaches (e.g., feedback error learning [3], [4]), where the output of a feedback controller provides the learning signal.

We propose to learn error terms at the level of RBD and actuation dynamics using existing supervised learning algorithms. Inverse dynamic controllers are synthesised based on these learned models. We validate the proposed method by executing locomotion tasks (base-motions, walking and trotting) with the Hydraulically-actuated Quadruped (HyQ) robot [5] shown in Fig. 1.

A. Related work

The field of model learning for robot control has been actively researched for a long time. A complete survey can be found in [2]. Modern and successful approaches recognize that the structure of RBD provides a useful prior on the dynamics and exploit this together with the flexibility of general supervised learning techniques in Semi-parametric methods [6], [7], [8], [9]. With a focus on transfer learning [10] proposes to learn multiple models on individual joint

level instead of a single global model. It was shown that such structure to the learning problem improves generalization performance between tasks [11], [12].

Research on learning of system dynamics for robot control has focused on fixed-base systems. In contrasts, legged robots are floating-base systems permanently influenced by contact forces. Additionally, during locomotion tasks different set of points are in contact with the environment, i.e., the contact configuration changes over time. For tackling model learning in contact rich environments, [13] proposes to model the effect of contact forces on a robot arm from skin measurements. A mixture of experts is used to generalize from a single contact to multi-contact situations. However, this approach requires the contact free dynamics to be known to separate the contribution due to contact forces from the rest of the dynamics.

Another viewpoint treats the contacts as part of the model and learns a global model that implictely incorporates the contact constraints [14]. However, this requires the constraint to be a fixed function of the robot's state. For legged robots this assumption is too restrictive when we want a model that is valid for general terrain geometry, not just the one experienced during training.

Motivated by the challenge to approximate unknown non-linearities in a high-dimensional input space, several non-parametric supervised learning algorithms have been proposed for robotic applications. Locally Weighted Projection Regression (LWPR) [15] provides a fast, incremental learning algorithm. At the same time, Gaussian Process Regression (GPR)[16] provides higher regression accuracy, but has high computational cost. Sparse [17], local [18], [19], and spectral [20] approximations to GPR have been proposed to provide a balance between accuracy and computation.

B. Method overview

In this work we propose a formulation for learning a structured error in the dynamics of a legged robot. Starting from one of the variants explored in [6], the analytical RBD is fixed a priori and used as the nominal model around which an error term is learned from data. As an advantage of this formulation, the learned model falls back to the nominal model in unexplored parts of the state space. We separately learn model errors that occur at the RBD and actuation dynamics level.

As presented in Section II, we use the approach proposed in [21] for modelling constrained multibody systems based on a linear projector operator: The equations of motion are projected onto the null-space of the constraints. In such subspace the dynamics of the system does not depend on the contact forces. Learning in this subspace thus means that contact force measurements are no longer required. Additionally, we use the columns of the end-effector Jacobians as a basis for the models, which allows to obtain models that generalizes across contact configurations.

In Section III the learning of actuator dynamics errors is discussed. By learning actuator specific errors we aim to achieve better generalization of such errors as in [10].

In Section IV, we show how the learned models are used in an inverse dynamics control framework applied on a torque-controlled, hydraulically-actuated quadruped robot. In Section V, we show the results from two different perspectives: (i) the quality of the error learning and (ii) the improvement of the controller based on the learned model.

II. Learning rigid body dynamics errors

In this section we provide the necessary background on the projection of the RBD into the null-space of the constraints. At the core of the contribution of this paper, we derive a new projected RBD error term. In Section II-C, we propose a change of basis that allows for foothold invariant model learning. Finally, in Section II-D we formulate a supervised learning problem for modeling the projected error term using sensor data.

A. Projected rigid body dynamics

A legged robot can be described by $n+6$ degrees of freedom (DoF): $\boldsymbol{q} = \left[\boldsymbol{q}_b^T, \boldsymbol{q}_j^T\right]^T$, where $\boldsymbol{q}_b \in SE(3)$ is the floating base position and orientation in Euler angles, and $\boldsymbol{q}_j \in \mathcal{R}^n$ is the joint configuration.

The RBD of a floating base robot is given by

$$\mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \mathbf{S}^T\boldsymbol{\tau} + \mathbf{J}_c^T(\boldsymbol{q})\boldsymbol{\lambda}_c, \tag{1}$$

where $\mathbf{M} \in \mathcal{R}^{(n+6)\times(n+6)}$ is the inertia matrix, $\boldsymbol{h} \in \mathcal{R}^{(n+6)}$ contains the Coriolis, centrifugal and gravitational forces, $\mathbf{S} \in \mathcal{R}^{n\times(n+6)}$ is the torque selection matrix mapping each torque $\boldsymbol{\tau} \in \mathcal{R}^n$ to the corresponding joint, $\mathbf{J}_c \in \mathcal{R}^{k\times(n+6)}$ is the constraint Jacobian of the k constraints, and $\boldsymbol{\lambda}_c \in \mathcal{R}^k$ corresponds to the vector of constraint forces. The subscript c is used throughout this paper to indicate that certain terms explicitly depend on the set of points in contact, i.e., the contact configuration.

As proposed in [21], we use the linear projection operator $\mathbf{P}_c = \mathbf{I} - \mathbf{J}_c^\dagger\mathbf{J}_c$, to project the dynamics into the null-space of the constraint Jacobian, i.e. $\mathbf{P}_c\mathbf{J}_c^T = \mathbf{0}$, where the remaining free dynamics are consequently given by

$$\mathbf{P}_c\left(\mathbf{M}\ddot{\boldsymbol{q}} + \boldsymbol{h}\right) = \mathbf{P}_c\mathbf{S}^T\boldsymbol{\tau}. \tag{2}$$

As can be seen from (2), contact forces $\boldsymbol{\lambda}_c$ no longer appear in the projected dynamics. The first contribution of this paper consists in learning the RBD errors in such projected subspace.

B. Projected rigid body dynamics errors

As in [6], a model error term $\boldsymbol{\Phi}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$ is introduced as a generalized force in (1),

$$\mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{\Phi}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \mathbf{S}^T\boldsymbol{\tau} + \mathbf{J}_c^T(\boldsymbol{q})\boldsymbol{\lambda}_c. \tag{3}$$

The model error $\boldsymbol{\Phi}$ depends on position, velocity, and acceleration. This allows to capture general unmodelled dynamics. Multiplying this equation with the projection operator allows us to solve for the projected RBD errors,

$$\mathbf{P}_c\boldsymbol{\Phi}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \mathbf{P}_c\left(\mathbf{S}^T\boldsymbol{\tau} - \mathbf{M}\ddot{\boldsymbol{q}} - \boldsymbol{h}\right). \tag{4}$$

With only known terms on the right-hand side, (4) can be used to compute the projected model errors $\mathbf{P}_c\boldsymbol{\Phi}$ from

measurements $\{c, \boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \boldsymbol{\tau}\}$. The contact configuration c is required since the projector depends on the contact Jacobian. For a single contact configuration (4) is sufficient to learn a projected model error.

However, this also implies that every contact configuration has its own projected RBD error function. It would then be necessary to independently learn m projected error models, one per contact configuration. For example, for the case of a point-feet quadruped robot we would have $m = 2^4$ separate models.

The complexity of maintaining different models and switching between them motivates further developing our approach to generalize across contact configurations.

We note that for a fixed contact constraint $f_c(\boldsymbol{q}) = 0$, the contact state c can be determined from $\boldsymbol{q}$. Each of the m models would then live in a different part of the model input space, and a consistent global model can be obtained. However, for our application, we aim for a model that is independent of terrain geometry. Already the arbitrary origin of base position after state estimator initialization changes the contact constraint $f_c(\boldsymbol{q}) = 0$ between experiments.

C. Contact invariant formulation

We propose a change of basis for the error term $\boldsymbol{\Phi}$. With this change of basis, we find a space where the projected errors $\mathbf{P}_c\boldsymbol{\Phi}$ of one contact configuration can be related to those of another configuration, forming a contact invariant formulation.

We assume that a legged robot only interacts with its environment through the end-effectors, with Jacobian $\mathbf{J}_{ee,i}$. In the case of point-contacts, only the rows associated to linear velocity are used and the Jacobian becomes $\mathbf{J}_{ee,i} \in \mathcal{R}^{3\times(n+6)}$.

For our formulation we require that the block associated to the joints, $\partial \boldsymbol{p}_{ee,i}/\partial \boldsymbol{q}_j$, is full row rank. This ensures that the columns of our proposed basis remain linearly independent.

Next, we define the Jacobian between the floating base and the generalized coordinates, trivially given by $\mathbf{J}_0 = \left[\mathbf{I}_{6\times6}, \mathbf{0}_{6\times n}\right]$. We proceed by concatenating the floating base Jacobian and Jacobian of N_{ee} end-effectors to obtain

$$\mathbf{J}^T = \left[\mathbf{J}_0^T, \mathbf{J}_{ee,1}^T, \ldots, \mathbf{J}_{ee,N_{ee}}^T\right]. \tag{5}$$

In case all end-effectors are without redundant joints, $\mathbf{J}^T \in \mathcal{R}^{(n+6)\times(n+6)}$ is a square, full rank matrix. When end-effectors with redundancy are present, $\mathbf{J}^T$ can be augmented to a square, full rank matrix $\mathbf{J'}^T = \left[\mathbf{J}^T, \mathbf{J}_{virt.}^T\right]$, where $\mathbf{J}_{virt.}^T$ is a virtual Jacobian whose columns span the null space of $\mathbf{J}$. The matrix $\mathbf{J}^T$ is used as a basis to represent the generalized force error $\boldsymbol{\Phi}$ with:

$$\boldsymbol{\Phi}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \mathbf{J}^T\tilde{\boldsymbol{\Phi}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}). \tag{6}$$

We refer to this new representation, $\tilde{\boldsymbol{\Phi}} \in \mathcal{R}^{n+6}$, as *the contact invariant model errors*. Since $\mathbf{J}^T$ is invertible, $\tilde{\boldsymbol{\Phi}} \in \mathcal{R}^{n+6}$ is unique for a given input $(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$.

Furthermore, we introduce the notion of constrained and unconstrained directions. Since $\mathbf{J}^T$ contains all end-effector Jacobians, it can form all possible constraint Jacobians $\mathbf{J}_c^T$ with a subset of its columns. The remaining columns then form the unconstrained directions $\mathbf{J}_u^T \in \mathcal{R}^{(n+6)\times(n+6-k)}$ as the complement. Now, for each contact configuration, we can group the entries of the model errors as

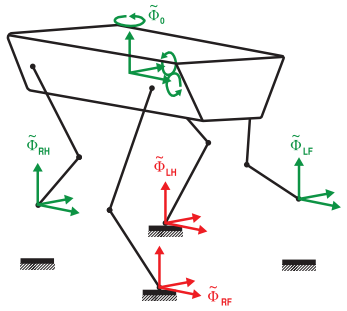


Fig. 2: Contact invariant model error visualization for a point-feet quadruped robot with the left-hind (LH) and right-front (RF) legs in contact. The errors located at the feet in contact form the constrained model error directions $\tilde{\boldsymbol{\Phi}}_c = [\tilde{\boldsymbol{\Phi}}_{RF}^T, \tilde{\boldsymbol{\Phi}}_{LH}^T]^T$. The unconstrained directions are $\tilde{\boldsymbol{\Phi}}_u = [\tilde{\boldsymbol{\Phi}}_0^T, \tilde{\boldsymbol{\Phi}}_{LF}^T, \tilde{\boldsymbol{\Phi}}_{RH}^T]^T$.

$$\mathbf{J}^T\tilde{\boldsymbol{\Phi}} = \mathbf{J}_c^T\tilde{\boldsymbol{\Phi}}_c + \mathbf{J}_u^T\tilde{\boldsymbol{\Phi}}_u, \tag{7}$$

where $\tilde{\boldsymbol{\Phi}}_c \in \mathcal{R}^k$, and $\tilde{\boldsymbol{\Phi}}_u \in \mathcal{R}^{n+6-k}$ denote the entries corresponding to constrained and unconstrained model error directions respectively.

The contact invariant model errors are visualized for a point-feet quadruped robot in Fig. 2 to provide intuition for the proposed method: model errors are represented by virtual forces at the base and end-effectors.

Connecting to the previous discussion on projected errors, we observe the projected model errors under the contact invariant representation,

$$\mathbf{P}_c\boldsymbol{\Phi} = \mathbf{P}_c\mathbf{J}_c^T\tilde{\boldsymbol{\Phi}}_c + \mathbf{P}_c\mathbf{J}_u^T\tilde{\boldsymbol{\Phi}}_u = \mathbf{P}_c\mathbf{J}_u^T\tilde{\boldsymbol{\Phi}}_u. \tag{8}$$

The projection, $\mathbf{P}_c\mathbf{J}_c^T = \mathbf{0}$, removes the constrained directions of the model errors. The new formulation is thus able to represent the vector $\mathbf{P}_c\boldsymbol{\Phi} \in \mathcal{R}^{n+6}$, which lives in a $(n+6-k)$-dimensional subspace due to the projection, by using exactly $(n+6-k)$ elements of the contact invariant model errors $\tilde{\boldsymbol{\Phi}}$.

After substituting (8) into (4), we obtain

$$\mathbf{P}_c\mathbf{J}_u^T\tilde{\boldsymbol{\Phi}}_u(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \mathbf{P}_c\left(\mathbf{S}^T\boldsymbol{\tau} - \mathbf{M}\ddot{\boldsymbol{q}} - \boldsymbol{h}\right), \tag{9}$$

which allows a unique observation of $\tilde{\boldsymbol{\Phi}}_u$. Such solution exists because the projected unconstrained Jacobian spans the constraint consistent subspace:

$$S(\mathbf{P}_c\mathbf{J}_u^T) = S(\mathbf{P}_c\mathbf{J}^T) = S(\mathbf{P}_c), \tag{10}$$

where the second equality holds because $\mathbf{J}^T$ is full rank.

The observation function is given by:

$$\tilde{\boldsymbol{\Phi}}_u(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \left(\mathbf{P}_c\mathbf{J}_u^T\right)^\dagger\mathbf{P}_c\left(\mathbf{S}^T\boldsymbol{\tau} - \mathbf{M}\ddot{\boldsymbol{q}} - \boldsymbol{h}\right). \tag{11}$$

To summarize, we propose a change of basis for the model errors in (6). Afterwards, we distinguish between terms associated with constrained and unconstrained directions in (7). With (11) we show how to uniquely solve for the elements associated with unconstrained directions. Each contact configuration therefore provides a partial view on the contact invariant error representation $\tilde{\boldsymbol{\Phi}}$. This error representation is

shared by all configurations and thus enables generalization among them.

D. Supervised learning

A supervised learning problem is formulated from (11) with input and output spaces given by

$$\boldsymbol{x} = [\underline{\boldsymbol{q}}^T, \dot{\boldsymbol{q}}^T, \ddot{\boldsymbol{q}}^T]^T, \tag{12}$$

$$\boldsymbol{y}_u = \left(\mathbf{P}_c \mathbf{J}_u^T\right)^\dagger \mathbf{P}_c \left(\mathbf{S}^T \boldsymbol{\tau} - \mathbf{M}\ddot{\boldsymbol{q}} - \boldsymbol{h}\right), \tag{13}$$

where $\underline{\boldsymbol{q}}$ is the vector of generalized coordinates without the yaw orientation component as well as the base positions. These states are removed because they have an arbitrary value and do not affect the system dynamics. The target outputs $\boldsymbol{y}_u$ can be obtained from data samples $\{c, \boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \boldsymbol{\tau}\}$. A separate model is learned for each output dimension.

We assume that full body position and velocity estimates are provided by a state estimator, but acceleration measurements are not directly available. In the current work we therefore use a linear phase finite impulse response (FIR) derivative filter to obtain accelerations from velocity measurements. To avoid the amplification of noise in the velocity signal, the differentiation filter is convolved with a low pass filter with a cutoff frequency of 10Hz. All position, velocity and torque trajectories are then smoothed with the same low pass filter so that the same frequency content is present in every signal. A filter width of 31 samples is used for all signals, which are all sampled at 250Hz.

In this work, we use LWPR [15] and Sparse GPR (SGPR) [17] with the Matérn-3/2 kernel to learn the contact invariant model errors $\tilde{\boldsymbol{\Phi}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$. For SGPR we use the GPy implementation [22].

III. Learning Actuation Errors

A second source of errors in the control of legged robots is the difference between torque command $\boldsymbol{u}$ and the measured torque $\boldsymbol{\tau}$. While low level controllers are in place, there are still systematic errors made due to complex non-linear phenomenon like friction inside the actuation subsystem. Here we see an opportunity to learn these errors, include them in the model, and compensate for them in the feedforward command.

The term $\boldsymbol{\Gamma}$ is introduced to model all errors between commanded and measured torque. The learning of actuation errors is done independently for each actuator, which results into n supervised learning problems

$$\boldsymbol{\Gamma}_i(\boldsymbol{q}_{j,i}, \dot{\boldsymbol{q}}_{j,i}, \ddot{\boldsymbol{q}}_{j,i}) = \boldsymbol{\tau}_i - \boldsymbol{u}_i. \tag{14}$$

Additionally, the input space for the learning problem is limited to the position, velocity, and acceleration of the particular joint associated to the actuator.

In order to relate the RBD and actuation level error learning, the measured torque in the RBD error formulation in (3) is substituted for the commanded torque and actuation errors:

$$\begin{aligned}\mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{\Phi}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \\ \mathbf{S}^T \left(\boldsymbol{u} + \boldsymbol{\Gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})\right) + \mathbf{J}_c^T(\boldsymbol{q})\boldsymbol{\lambda}_c.\end{aligned} \tag{15}$$

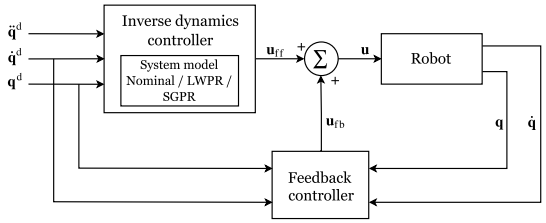


Fig. 3: Control diagram of the feedforward QP controller based on the learned model. The Inverse dynamics controller runs at 250Hz. The Feedback controller contains a feedback linearized acceleration feedback running at 250Hz, and a joint position controller running at 1kHz.

Both sources of error can now be combined into a single model error term $\boldsymbol{\epsilon}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$, resulting in

$$\mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{\epsilon}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \mathbf{S}^T\boldsymbol{u} + \mathbf{J}_c^T(\boldsymbol{q})\boldsymbol{\lambda}_c, \tag{16}$$

$$\boldsymbol{\epsilon}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \boldsymbol{\Phi}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) - \mathbf{S}^T\boldsymbol{\Gamma}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}). \tag{17}$$

IV. Application to Control

As hypothesized in the introduction, the use of a better model should improve the performance of model based control techniques. We extend an existing model based optimal control method [23] for legged robots, incorporating the learned model. We use the extended controller for executing locomotion tasks on a real quadruped robot. A schematic overview of the controller is provided in Fig. 3.

A. Feedforward control

The feedforward controller in [23] is based on inverse dynamics control with optimal distribution of contact forces. This controller defines a quadratic program (QP) and allows cost functions quadratic in both torques and contact forces. This formulation relies on the QR decomposition of the constraint Jacobian,

$$\mathbf{J}_c^T = \mathbf{Q}\begin{bmatrix}\mathbf{R}\\ \mathbf{0}\end{bmatrix} = \begin{bmatrix}\mathbf{Q}_c, \mathbf{Q}_u\end{bmatrix}\begin{bmatrix}\mathbf{R}\\ \mathbf{0}\end{bmatrix}. \tag{18}$$

Multiplying the system dynamics with $\mathbf{Q}^T$ divides the system into constrained and unconstrained subsystems. We apply this decomposition to (16), extending the two subsystems with the learned error models,

$$\mathbf{Q}_u^T\left(\mathbf{M}\ddot{\boldsymbol{q}}^d + \boldsymbol{h}\right) + \mathbf{Q}_u^T(\mathbf{J}^T\tilde{\boldsymbol{\Phi}} - \mathbf{S}^T\boldsymbol{\Gamma}) = \mathbf{Q}_u^T\mathbf{S}^T\boldsymbol{u}, \tag{19}$$

$$\mathbf{Q}_c^T\left(\mathbf{M}\ddot{\boldsymbol{q}}^d + \boldsymbol{h}\right) + \mathbf{Q}_c^T(\mathbf{J}^T\tilde{\boldsymbol{\Phi}} - \mathbf{S}^T\boldsymbol{\Gamma}) = \mathbf{Q}_c^T\mathbf{S}^T\boldsymbol{u} + \mathbf{R}\boldsymbol{\lambda}_c, \tag{20}$$

where all terms are evaluated at the desired trajectory $\left(\boldsymbol{q}^d, \dot{\boldsymbol{q}}^d, \ddot{\boldsymbol{q}}^d\right)$. The QP is posed with the unconstrained dynamics as equality constraints,

$$\begin{aligned}\arg\min_{\boldsymbol{u},\boldsymbol{\lambda}} \quad & \frac{1}{2}\boldsymbol{u}^T\mathbf{W}_{\boldsymbol{u}}\boldsymbol{u} + \boldsymbol{b}_{\boldsymbol{u}}^T\boldsymbol{u} + \frac{1}{2}\boldsymbol{\lambda}^T\mathbf{W}_{\boldsymbol{\lambda}}\boldsymbol{\lambda} + \boldsymbol{b}_{\boldsymbol{\lambda}}^T\boldsymbol{\lambda} \\ \text{s.t. } & \mathbf{Q}_u^T\mathbf{S}^T\boldsymbol{u} = \mathbf{Q}_u^T\left(\mathbf{M}\ddot{\boldsymbol{q}}^d + \boldsymbol{h}\right) + \mathbf{Q}_u^T(\mathbf{J}^T\tilde{\boldsymbol{\Phi}} - \mathbf{S}^T\boldsymbol{\Gamma}) \\ & \mathbf{C}_{\boldsymbol{u}}\boldsymbol{u} \leq \boldsymbol{d}_{\boldsymbol{u}} \\ & \mathbf{C}_{\boldsymbol{\lambda}}\boldsymbol{\lambda} \leq \boldsymbol{d}_{\boldsymbol{\lambda}}.\end{aligned} \tag{21}$$

In [23] it is shown that this problem can be rewritten to a QP with only the torque commands as decision variables by

solving (20) for the contact forces and substituting them into (21).

Here we use a simple cost function to minimize the applied torque by setting $\mathbf{W_u} = \mathbf{I}$, $\mathbf{W_\lambda} = \mathbf{0}$, $\boldsymbol{b_\tau} = \mathbf{0}$, and $\boldsymbol{b_\lambda} = \mathbf{0}$. The inequality constraint for the contact forces is obtained by requiring normal forces to be positive and by an octagon approximation to the friction cone constraint, where the friction coefficient is set to 0.5 for all experiments.

B. Feedback control

The feedback controller is the same for all controllers, and it is given by a sum of two stabilizing terms,

$$\boldsymbol{u}_{fb} = \boldsymbol{u}_{fb,pd} + \boldsymbol{u}_{fb,lin.}. \tag{22}$$

The first term is a default joint position PD controller,

$$\boldsymbol{u}_{fb,pd} = -\mathbf{K}_{d,pd}\dot{\boldsymbol{e}} - \mathbf{K}_{p,pd}\boldsymbol{e}, \tag{23}$$

where $\dot{\boldsymbol{e}}$ and $\boldsymbol{e}$ are the tracking errors in velocity and position.

The second feedback component is provided through a feedback linearization of the nominal model,

$$\boldsymbol{u}_{fb,lin.} = \left(\mathbf{P}_c\mathbf{S}^T\right)^\dagger \mathbf{P}_c\mathbf{M}\left(-\mathbf{K}_{d,lin.}\dot{\boldsymbol{e}} - \mathbf{K}_{p,lin.}\boldsymbol{e}\right). \tag{24}$$

V. Results

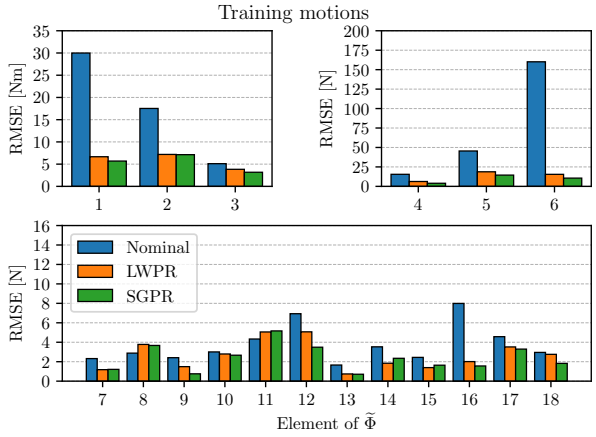
We apply these methods to the HyQ robot (see Fig. 1). This robot weighs 80kg, and has 3 actuated joints per leg, i.e. $n = 12$. Sensor data for the state of the base are provided by a state estimator [24].

We use standing, walking, and trotting motions for both data collection and validation. For the standing motion trajectories, a reference is generated with a cubic spline interpolation of waypoints. For walking and trotting, full body reference trajectories are generated with a trajectory optimization framework [25].

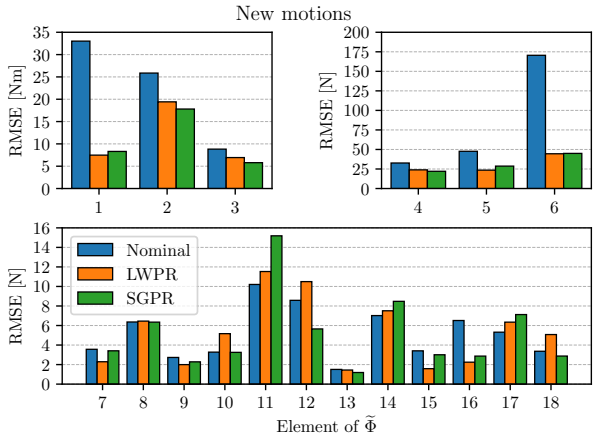
We collected data at 250Hz during repeated experiments and obtained a training dataset of 42,481 data points for standing, 8,615 for walking, and 2,707 for trotting. This dataset is used to learn error models using LWPR, and SGPR, on an Intel Core i7/2.5 GHz Quadcore laptop. Training times for LWPR, with 5 training epochs, amount to 29min for the RBD and 2min for the actuation model. The models per dimension converge to between 100 and 250 receptive fields for the RBD errors and around 15 for the actuation errors. For SGPR with 100 inducing inputs per dimension for actuation errors and 250 for the RBD errors, 58min and 26min are required.

We use *training motions*, new executions of the training tasks, to validate the learned models. Generalization between tasks is evaluated on *new motions*; a set of different walking and trotting motions. The hardware experiments are shown in a video available at https://youtu.be/Kfy3IQ9wujk.

All experiments consist of only one gait cycle, because stabilizing multiple gait cycles without replanning introduces a larger performance variance per controller. We argue that such variance distracts from the comparison between controllers and that a cycle gait cycle is enough to expose performance differences.



(a)



(b)

Fig. 4: Root Mean Squared Error (RMSE) of model error predictions for the RBD error term $\tilde{\boldsymbol{\Phi}}$ plotted per dimension for training motions (a) and new motions (b). The predictions are grouped into base torques {1-3} and forces {4-6}; and the feet forces: LF {7-9}, RF {10-12}, LH {13-15}, and RH {16-18}.

A. Model learning performance

In this section, prediction accuracy of the learned models is evaluated. The validation set consists of 7,413 data points for standing, 1,419 for walking, and 897 for trotting. For the case of new tasks, the validation set consists of 4,839 data points for walking, and 1,629 for trotting. Note that for validating the performance of the model learning, the controller used for executing the experiments is irrelevant: The performance is measured by comparing predicted and measured values of the model errors $\tilde{\boldsymbol{\Phi}}$ and $\boldsymbol{\Gamma}$. We report the Root Mean Squared Error (RMSE) per dimension of the error term. The results for the nominal model provide the baseline of predicting zero for all errors, i.e. $\tilde{\boldsymbol{\Phi}}_i = 0$. The model error subscript i in this section is used to indicate the analysed dimension of the error.

The RMSE results for the RBD level are shown separately for training motions in Fig 4a, and new motions in Fig 4b. The accuracy of the learned model is significantly better than the nominal case for the error at the floating base, $\tilde{\boldsymbol{\Phi}}_{1-6}$, for both the training and new motions.

For the model errors associated to the end-effectors, i.e. $\tilde{\boldsymbol{\Phi}}_{7-18}$, the performance of the nominal model is already

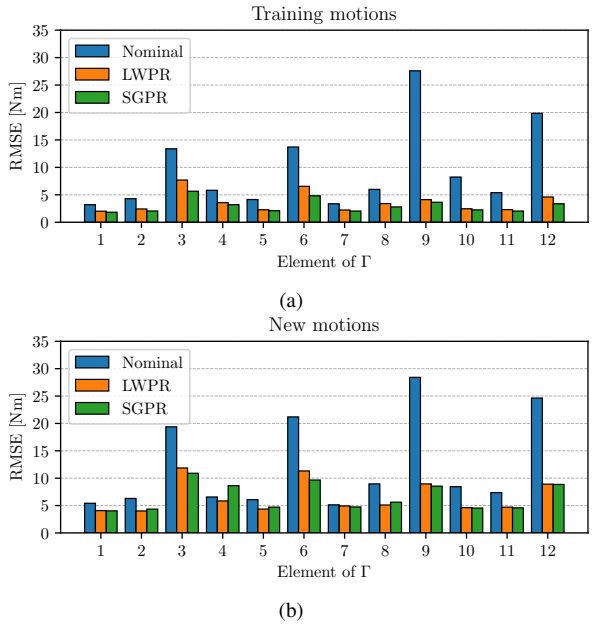


Fig. 5: Root Mean Squared Error (RMSE) of model error predictions for the actuator error term $\boldsymbol{\Gamma}$ plotted per joint for training motions (a) and new motions (b).

of high quality. As seen in Figs. 4a and 4b, most of the dimensions have a RMSE in the order of 4N. The learned models are able to slightly improve on that within the training tasks, but this improvement does not generalize to the new motions.

RMSE of the actuation dynamics model error is also shown and reported per joint for training motions in Fig 5a, and new motions in Fig 5b. The learned models show improved prediction accuracy compared to the nominal model baseline across joints and motions.

A closer look is provided at the model error predictions during a 45cm trot experiment, which is part of the new motions. The estimated gait pattern is shown in Fig. 6. Impacts cause a discontinuity in velocity when changing foothold. The acceleration estimate is affected while this contact change is inside its filter window, which introduces noise in both model error estimates and function learning input space. We therefore discarded half a window size of training samples before and after a change in contact configuration.

The RBD error predictions for the trotting trajectory are shown in Fig. 7. The difference between the measured and predicted model errors are indeed significantly worse in the indicated areas around a foothold change. Outside of these regions the model errors are consistent with the confidence interval of the SGPR.

The actuation error predictions shown in Fig. 8 are much less affected by the aforementioned issues. The SGPR is able to accurately predict the error in both the stance phase and the swing phase between 0.80s and 1.05s. We observe that actuation errors are especially high during swing phases. The next section shows how the ability to predict these model errors with high accuracy is beneficial for control.

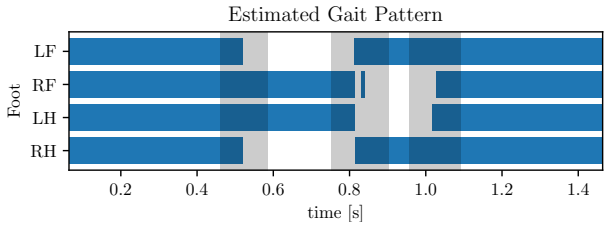


Fig. 6: Estimated gait pattern for a trotting trajectory. The gray band indicate the band in which the acceleration filter is still affected by the change in contact.

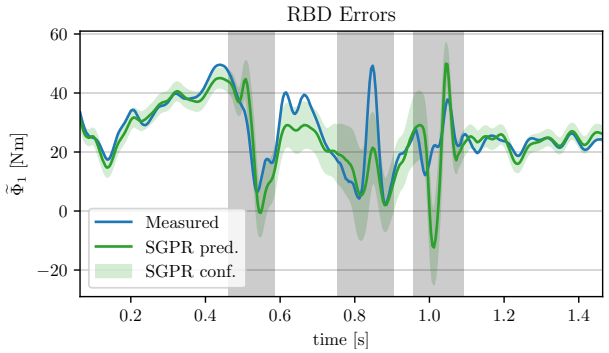


Fig. 7: Measured and predicted RBD errors in the roll direction of the base, $\tilde{\boldsymbol{\Phi}}_1$. The SGPR prediction are plotted together with the 95% confidence interval. The grey bands around contact changes in Fig. 6 are repeated.

B. *Controller performance*

We analyse the effect of using a learned model in an inverse dynamics based controller. To compare the controllers performance we define the following metric to capture Root Mean Square (RMS) feedback effort:

$$\text{RMS}(||\boldsymbol{u}_{fb}||) = \sqrt{\frac{1}{N}\sum_{k=0}^{N-1} \boldsymbol{u}_{fb,k}^T \boldsymbol{u}_{fb,k}}, \tag{25}$$

where $\boldsymbol{u}_{fb}$ is the applied feedback signal. Similarly, RMS tracking errors are defined for the attitude $(\boldsymbol{e}_{att})$, position, $(\boldsymbol{e}_{pos})$, and joint $(\boldsymbol{e}_{joint})$ errors. These metrics allow comparison of controllers within a certain task. Comparison between tasks is not possible, as some tasks are inherently more difficult to stabilize.

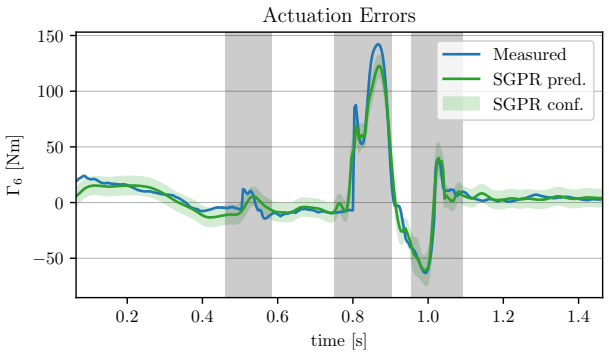


Fig. 8: Measured and predicted actuation errors in the right front knee joint, $\boldsymbol{\Gamma}_6$. The SGPR prediction are plotted together with the 95% confidence interval. The grey bands around contact changes in Fig. 6 are repeated.

TABLE I: Controller performance for new executions of the training tasks.

		RMS$(\cdot)$			
		$\|\|u_{fb}\|\|$	$\|\|e_{att}\|\|$	$\|\|e_{pos}\|\|$	$\|\|e_j\|\|$
Motion	Model	[Nm]	[deg]	[mm]	[deg]
Base-only	Nominal	40.6	1.0	7.3	6.5
	LWPR	17.4	0.8	4.3	3.1
	SGPR	**16.2**	**0.7**	**3.8**	**2.9**
Walk 35cm	Nominal	56.8	1.3	21.5	11.1
	LWPR	28.8	**0.9**	17.1	4.8
	SGPR	**25.8**	0.9	**13.9**	**4.6**
Trot 35cm	Nominal	58.4	1.4	32.2	11.9
	LWPR	27.2	0.9	**11.1**	5.1
	SGPR	**25.4**	**0.9**	12.0	**4.9**

TABLE II: Controller performance for new tasks.

		RMS$(\cdot)$			
		$\|\|u_{fb}\|\|$	$\|\|e_{att}\|\|$	$\|\|e_{pos}\|\|$	$\|\|e_j\|\|$
Motion	Model	[Nm]	[deg]	[mm]	[deg]
Walk 20cm	Nominal	61.2	1.6	24.9	10.9
	LWPR	**30.4**	1.2	20.5	6.2
	SGPR	36.7	**0.9**	**17.5**	**5.8**
Walk 30cm	Nominal	58.1	1.3	25.7	10.9
	LWPR	29.3	0.8	11.3	**4.6**
	SGPR	**27.1**	**0.6**	**10.9**	4.9
Rear Walk 30cm	Nominal	54.6	1.0	21.9	10.4
	LWPR	**32.8**	**0.8**	**20.4**	**4.8**
	SGPR	37.9	1.2	32.5	5.7
Side walk 25cm	Nominal	55.8	1.3	21.3	8.8
	LWPR	**26.8**	**0.8**	17.6	**4.0**
	SGPR	40.8	1.1	**17.4**	7.9
Trot 20cm	Nominal	54.0	1.3	17.3	10.4
	LWPR	24.4	0.8	**7.8**	**4.8**
	SGPR	**23.9**	**0.7**	8.1	5.3
Trot 45cm	Nominal	69.4	1.7	39.9	13.1
	LWPR	**25.5**	0.9	**16.8**	**5.1**
	SGPR	27.5	**0.5**	22.6	6.8

The results for all motion and controller combinations are shown in Table I for new executions of the training tasks and Table II for new tasks. The controller based on the SGPR model uses the least amount of feedback effort and achieves lower tracking errors on the training motions. This shows that the inverse dynamics controller is able to exploit the more accurate model predictions of the SGPR shown in Figs. 4a and 5a. For the new motions in Table II, the performance difference between LWPR and SGPR is less pronounced.

The torque commands for the right front (RF) leg during trotting are plotted in Fig. 9 to show the difference in control signals when using different models. The controllers based on the learned models have clearly different feedforward signals across all joints. As a result, overall lower feedback signals as seen in the bottom figure for each joint.

We did not encounter visual difference between experiments with and without a learned model. As seen in Table I and II, the tracking errors are not so large that visual differences are expected. Stiff feedback gains were used to ensure tracking under the nominal model. While the same feedback gains were used with the learned models to allow comparison, these gains could in the future be lowered due to improved feedforward signals. This would in return reduce oscillations that we occasionally experience due to high gains. Such compliant behaviour while maintaining tracking performance will be especially beneficial for rough terrain locomotion.

VI. Discussion

In this work we verified generalization between repetitions of the same task and to similar tasks. Each error element showed a different degree of generalization. Actuation errors generalize very well to the new motions and are naturally at an advantage with respect to the RBD error model due to the lower input dimensionality. RBD errors associated to the base do, but those for the end-effector dimensions do not generalize to new motions. We hypothesize that the latter are more locally dependent on joint states, and thus generalize less to the new motions, which differ more in joint motions than base motion. It is therefore the combination of predicted base level RBD errors and actuation errors that result in the improved control performance on the new tasks.

To discuss generalization to a wider range of tasks, we point back to the chosen error formulation in (3). We use learning methods that have local support around training data and adjust to the local landscape. Moreover, the user has control over how much the learned models will extrapolate training data. Stronger regularization will force the error prediction back to zero when moving away from training examples and returns the method to nominal performance. This limits the generalization to tasks in unseen parts of the input spaces. However, we prefer such conservative behaviour over excessive extrapolation of the training data and believe that it is important for a robust execution on hardware.

The reduced prediction quality around contact changes limits performance on highly dynamic motions. We encountered the current filtering strategy to be a limiting factor. An improved approach should address velocity discontinuities around contact switches. Furthermore, fusing IMU measurements with the filter based estimates could improve acceleration estimates. Finally, increasing the filter's cut-off frequency results in noisier data for the learning problem, but can in principle be compensated with more training data. Experimenting with this trade off could benefit the overall performance, especially for highly dynamic motions.

The assumption of non-singular end-effector Jacobians allowed us to solve for the model errors in (11). A singular Jacobian prevents a unique attribution of the model error to the individual elements of the error vector. As often done in robotics, this assumption can be relaxed by adding a regularization term. For our case of three joints per end-effector a small term of $\gamma \mathbf{I}_{3\times 3}$ can be added to the block $\partial \boldsymbol{p}_{ee,i}/\partial \boldsymbol{q}_{j,ee}$, where $\boldsymbol{q}_{j,ee}$ are the joints associated to that end-effector. However, the effect of such a regularization on the learning should be studied more carefully.

Throughout this work we have separated formulation from learning methods and used established learning libraries to focus on the formulation. The final control performance could now be further improved with the state-of-the-art methods in supervised learning.

VII. Conclusions

We presented a novel model learning formulation for legged robots. The formulation enables learning of model errors that

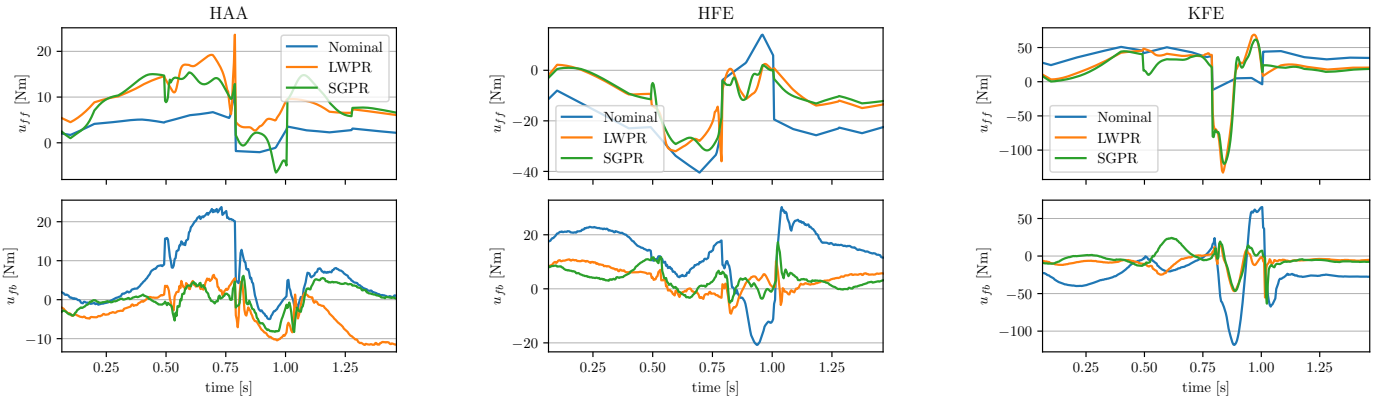


Fig. 9: Feedforward and feedback torques in the hip abduction/adduction (HAA), hip flexion/extension (HFE), and knee flexion/extension (KFE) of the right front (RF) leg during the 45cm trot. The control signals from experiments with inverse dynamics controllers based on the Nominal, LWPR, and SGPR models are compared. The swing phase of this leg occurs between 0.80s and 1.05s.

are contact invariant, without the need for contact force measurements. The results show that using supervised learning, a single model can improve model predictions across a variety of motions and contact configurations. Even though the training data does not cover the entire input space of the robot, we showed generalization to new, similar tasks.

Additionally, we showed how inverse dynamics control using a learned model significantly improves tracking performance on hardware. The controller is able to exploit accurate predictions by the learned model to achieve reduction in RMS feedback effort and tracking errors.

REFERENCES

[1] M. Mistry, S. Schaal, and K. Yamane, "Inertial parameter estimation of floating base humanoid systems using partial force sensing," in *9th IEEE-RAS International Conference on Humanoid Robots, HUMANOIDS09*, 2009, pp. 492–497.

[2] D. Nguyen-Tuong and J. Peters, "Model learning for robot control: A survey," *Cognitive Processing*, vol. 12, no. 4, pp. 319–340, 2011.

[3] M. Kawato, "Feedback-error-learning neural network for supervised motor learning," *Advanced neural computers*, vol. 6, no. 3, pp. 365–372, 1990.

[4] J. Nakanishi and S. Schaal, "Feedback error learning and nonlinear adaptive control," *Neural Networks*, vol. 17, no. 10, pp. 1453–1465, 2004.

[5] C. Semini, N. G. Tsagarakis, E. Guglielmino, M. Focchi, F. Cannella, and D. G. Caldwell, "Design of hyq a hydraulically and electrically actuated quadruped robot," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 225, no. 6, pp. 831–849, 2011.

[6] D. Nguyen-tuong and J. Peters, "Using Model Knowledge for Learning Inverse Dynamics," *IEEE International Conference on Robotics and Automation*, pp. 2677–2682, 2010.

[7] J. de la Cruz, D. Kulić, and W. Owen, "Online incremental learning of inverse dynamics incorporating prior knowledge," *Autonomous and Intelligent Systems*, pp. 167–176, 2011.

[8] R. Camoriano, S. Traversaro, L. Rosasco, G. Metta, and F. Nori, "Incremental semiparametric inverse dynamics learning," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 544–550.

[9] C.-A. Cheng, H.-P. Huang, H.-K. Hsu, W.-Z. Lai, and C.-C. Cheng, "Learning the inverse dynamics of robotic manipulators in structured reproducing kernel Hilbert space," *IEEE transactions on cybernetics*, vol. 46, no. 7, pp. 1691–1703, 2016.

[10] T. T. Um, M. S. Park, and J.-m. Park, "Independent Joint Learning : A Novel Task - to - Task Transfer Learning Scheme for Robot Models," *Proc. of IEEE International Conference on Robotics and Automation*, pp. 5679–5684, 2014.

[11] K. Caluwaerts and J. J. Steil, "Independent joint learning in practice: Local error estimates to improve inverse dynamics control," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, Nov 2015, pp. 643–650.

[12] Z. Shareef, P. Mohammadi, and J. Steil, "Improving the Inverse Dynamics Model of the KUKA LWR IV+ Using Independent Joint Learning," *IFAC-PapersOnLine*, vol. 49, no. 21, pp. 507–512, 2016.

[13] R. Calandra, S. Ivaldi, M. P. Deisenroth, E. Rueckert, and J. Peters, "Learning inverse dynamics models with contacts," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 3186–3191.

[14] C. A. Cheng and H. P. Huang, "Learn the lagrangian: A vector-valued rkhs approach to identifying lagrangian systems," *IEEE Transactions on Cybernetics*, vol. 46, no. 12, pp. 3247–3258, Dec 2016.

[15] S. Vijayakumar, A. D 'souza, and S. Schaal, "Incremental Online Learning in High Dimensions," *Neural Computation*, vol. 17, pp. 2602–2634, 2005.

[16] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. MIT Press MIT Press, 2004, vol. 14, no. 2.

[17] J. Quiñonero-Candela and C. E. Rasmussen, "A unifying view of sparse approximate Gaussian process regression," *Journal of Machine Learning Research*, vol. 6, no. Dec, pp. 1939–1959, 2005.

[18] D. Nguyen-Tuong, M. Seeger, and J. Peters, "Model Learning with Local Gaussian Process Regression," *Advanced Robotics*, vol. 23, no. 15, pp. 2015–2034, 2009.

[19] F. Meier, P. Hennig, and S. Schaal, "Efficient Bayesian local model learning for control," *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2244–2249, 2014.

[20] A. Gijsberts and G. Metta, "Real-time model learning using incremental sparse spectrum gaussian process regression," *Neural Networks*, vol. 41, no. Supplement C, pp. 59 – 69, 2013, special Issue on Autonomous Learning.

[21] F. Aghili, "A unified approach for inverse and direct dynamics of constrained multibody systems based on linear projection operator: Applications to control and simulation," *IEEE Transactions on Robotics*, vol. 21, no. 5, pp. 834–849, 2005.

[22] GPy, "{GPy}: A Gaussian process framework in python," http://github.com/SheffieldML/GPy, since 2012.

[23] L. Righetti, J. Buchli, M. Mistry, M. Kalakrishnan, and S. Schaal, "Optimal distribution of contact forces with inverse-dynamics control," *The International Journal of Robotics Research*, vol. 32, no. 3, pp. 280–298, 2013.

[24] M. Bloesch, M. Hutter, M. A. Hoepflinger, S. Leutenegger, C. Gehring, C. D. Remy, and R. Siegwart, "State estimation for legged robots-consistent fusion of leg kinematics and IMU," *Robotics*, vol. 17, pp. 17–24, 2013.

[25] D. Pardo, M. Neunert, A. Winkler, R. Grandia, and J. Buchli, "Hybrid direct collocation and control in the constraint-consistent subspace for dynamic legged robot locomotion," in *Proceedings of Robotics: Science and Systems*, Cambridge, Massachusetts, July 2017.