Learning Task-Specific Dynamics to Improve Whole-Body Control

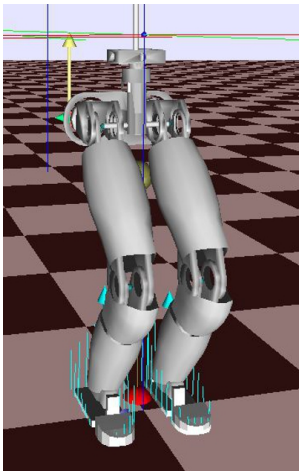
Andrej Gams[1], Sean A. Mason[2], Aleš Ude[1], Stefan Schaal[2] and Ludovic Righetti[3,4]

***Abstract*— In task-based inverse dynamics control, reference accelerations used to follow a desired plan can be broken down into feedforward and feedback trajectories. The feedback term accounts for tracking errors that are caused from inaccurate dynamic models or external disturbances. On underactuated, free-floating robots, such as humanoids, good tracking accuracy often necessitates high feedback gains, which leads to undesirable stiff behaviors. The magnitude of these gains is anyways often strongly limited by the control bandwidth. In this paper, we show how to reduce the required contribution of the feedback controller by incorporating learned task-space reference accelerations. Thus, we i) improve the execution of the given specific task, and ii) offer the means to reduce feedback gains, providing for greater compliance of the system. In contrast to learning task-specific joint-torques, which might produce a similar effect but can lead to poor generalization, our approach directly learns the task-space dynamics of the center of mass of a humanoid robot. Simulated and real-world results on the lower part of the Sarcos Hermes humanoid robot demonstrate the applicability of the approach.**

I. INTRODUCTION

Unmodeled dynamics (i.e. friction, link flexibilities, unmodeled actuator dynamics or approximate model parameters) can have severe effects on tracking performance of legged robots, and can be problematic not only for balancing but also to properly achieve other tasks. Models, however, are often difficult to obtain and/or incorrect, and while parameter identification can improve their quality [1], [2], it does not take into account unmodeled dynamic effects. The lack of accurate models has led to the use of combined feedforward and feedback control, which preserves stability and robustness to disturbances. The greater the error of the model, the greater the feedback gains necessary to ensure robust task achievement. This comes at the cost of a significant increase in stiffness and damping requirements, which can be a problem due to limited control bandwidth and which is often undesirable to prevent high impedance behaviors.

Different methods of acquiring dynamic models [3] and exploiting them in control have been proposed [4]. Optimization based approaches, such as hierarchical inverse dynamics [5]–[7], have gained in popularity in the recent years for the control of legged robots. However, these approaches rely on dynamics models and often necessitate high task-space feedback gains to ensure good tracking performance on real robots which do not have accurate dynamic models.



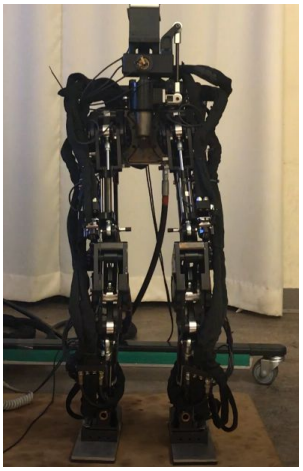


Fig. 1. Simulated and real-world lower part of the Sarcos Hermes humanoid robot used in the experiments.

Acquiring dynamic models of robots and tasks can be partially offset by iterative learning of the control signals, which relies on one of the main characteristics of robots: repeatability of the control actions from the same input signals. Iterative learning control (ILC) [8] was extensively applied in robotics, including for learning task-specific joint control torques [9].

A. Problem Statement

In this paper we investigate task-specific dynamics learning to improve task execution while increasing compliance in the scope of optimization-based inverse dynamics control. Therefore, we pursue the following objectives:

- reduce the required contribution of the feedback term in the control,
- consequently improve task-space tracking while increasing compliance, and
- act directly in the task space of interest, typically the center of mass (CoM) dynamics.

Acting directly in the task space as opposed to joint space will potentially enable applications beyond the scope of the learned dynamics, i. e., through generalization.

The intended application of the proposed algorithm is to improve control of dynamic tasks on humanoid robots. We performed our experiments on the lower part of the Sarcos Hermes humanoid robot, depicted in Fig. 1.

*This work was supported by the Slovenian Research Agency project BI-US/16-17-063, New York University, the Max-Planck Society and the European Unions Horizon 2020 research and innovation programme (grant agreement No 780684 and European Research Councils grant No 637935).

[1]Humanoid and Cognitive Robotics Lab, Dept. of Automatics, Biocybernetics and Robotics, Jožef Stefan Institute, Ljubljana, Slovenia. name.surname@ijs.si

[2]Computational Learning and Motor Control Lab, University of Southern California, Los Angeles, California, USA. name.surname@usc.edu

[3]Tandon School of Engineering, New York University, New York, USA. ludovic.righetti@nyu.edu

[4]Max Planck Institute for Intelligent Systems, Tuebingen, Germany.

This paper is organized as follows:. Section II provides a short literature overview. Section III gives an overview of the QP inverse dynamics controller used in the paper. Section IV presents our new algorithm that provides learned torques for task-space inverse dynamics control. Simulation and real robot experiments are present in Section V. Finally, we discuss our results and conclude in Sections VI and VII.

II. RELATED WORK

Two bodies of work are relevant to this research: 1) learning and exploiting control torques for improved task execution, and 2) task-space inverse dynamics control of floating-base systems.

Learning control signals to iteratively improve task execution using the aforementioned ILC [8], has been extensively used in robotics [10]. Feedback-error-learning, where the feedback motor command is used as an error signal to train a neural network which then generates a feedforward motor command, was proposed by Kawato [11]. The idea was extended for learning of contact interaction. In their work, Pastor et al. [12] combined learning of forces with dynamical systems to improve force tracking in interaction tasks. Similarly, in [13], dynamic movement primitives (DMP) [14] were used in combination with learning of interaction forces.

Learning to improve contact interaction was also applied directly to actuation torques. Joint torques along kinematic trajectories were learned and encoded as DMPs in [15] and used to increase the accuracy in subsequent executions of in-contact tasks. This approach was also applied to full-sized humanoid robots, for example in [16], where a particular trajectory was run, and the joint torques from that trial were used as the feedforward term on the next trial. These methods can go beyond mere repetition of the same task. In [9], the authors show how the learned joint-torque signals, encoded in parametric form as compliant movement primitives (CMPs), can be generalized for new task parameters (weight, speed, etc.), and in [17] how to effectively learn them. However, because the approach is rooted in joint-torques, generalization is somewhat limited to variations of the same full body motions, which is a strong limitation for highly redundant robots such as humanoids that can perform several concurrent tasks.

Optimization-based inverse dynamics control has become a very popular method to control legged robots [5], [6], [18] as it allows to directly specify control objectives for multiple tasks, while ensuring priorities between tasks and constraint satisfaction (actuation limits, friction cones, etc). The redundancy of a complex robot such as a humanoid can therefore be optimally exploited. It is also possible to compute optimal feedback gains in a receding horizon manner directly in task space by leveraging the task reduced dynamics [5], [7], [19]. Such methods have remarkable capabilities, but are often limited by modeling errors which necessitate to significantly increase feedback gains, which is either very limited by the effective control bandwidth or leads to very stiff behaviors.

Some approaches, as far back as [20], have proposed to learn task-specific dynamic models that can then be used to synthesize control laws. Iterative methods to compute locally optimal control policies have, for example, recently been used with such learned models [21]. Iterative repetitions of the process are then used to collect additional data and re-learn a new policy [21], [22]. In these approaches, the learned models and resulting control policies operate on the whole robot. It is therefore not clear how other tasks and additional constraints can be incorporated without impairing the resulting behaviors.

In this paper, we learn task-specific feedforward models which take into account the error in dynamic models during task execution and combine them with a Quadratic Programming (QP) based inverse dynamics controller. This allows to significantly improve task tracking while creating more compliant behaviors.

III. CONTROL

In this Section we briefly introduce the task-space inverse dynamics controller we use in the paper and that was originally proposed in [5]. We model the floating-base dynamics of a legged humanoid robot as

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

with a vector of position and orientation of the robot in space and its joint configurations $\boldsymbol{q} \in \mathbb{SE}(3) \times \mathbb{R}^n$, the mass-inertia matrix $\mathbf{M} \in \mathbb{R}^{(n+6)\times(n+6)}$, the generalized Coriolis, centrifugal and gravity forces collected in $\mathbf{h} \in \mathbb{R}^{n+6}$, the actuation matrix $\mathbf{S} \in [\mathbf{O}_{n\times 6}\ \mathbf{I}_{n\times n}]$ and the end-effector contact Jacobian $\mathbf{J}_c \in \mathbb{R}^{6m\times n}$, where n is the number of robot's degrees of freedom, $\boldsymbol{\tau}$ are actuation torques and $\boldsymbol{\lambda}$ are the contact forces.

As discussed in [5], the dynamics can be decomposed into actuated and unactuated (floating base) parts, respectively

$$\mathbf{M}_a(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}_a(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau} + \mathbf{J}_{c,a}^T\boldsymbol{\lambda}, \tag{2}$$

$$\mathbf{M}_u(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathbf{h}_u(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \mathbf{J}_{c,u}^T\boldsymbol{\lambda}. \tag{3}$$

and it is only necessary to enforce Eq. (3) to ensure dynamic consistency as $\boldsymbol{\tau}$ is a redundant variable which can be eliminated and replaced by a combination of $\ddot{\boldsymbol{q}}$ and $\boldsymbol{\lambda}$ according to Eq. (2) where necessary. Eq. (3) is the first constraint to be satisfied by the controller. Kinematic contact constraints ensuring that the part of the robot in contact with the environment does not move,

$$\mathbf{J}_c\ddot{\boldsymbol{q}} + \dot{\mathbf{J}}_c\dot{\boldsymbol{q}} = 0, \tag{4}$$

are additional equality constraints for the optimization, where $\mathbf{J}_c$ is the Jacobian for m_c constrained endeffectors. Additionally, we limit foot center of pressure (CoP), friction forces, resultant normal torques, joint torques and joint accelerations with linear inequality constraints. The cost to be minimized is

$$\begin{aligned} \min_{\ddot{\boldsymbol{q}},\lambda} \sum_t ||\ddot{\boldsymbol{x}}_t - \ddot{\boldsymbol{x}}_{t,\mathrm{des}}||^2_{\mathbf{W}_t} + \\ ||P_{\mathrm{null}}(\ddot{\boldsymbol{q}} - \ddot{\boldsymbol{q}}_{\mathrm{des}})||^2_{\mathbf{W}_q} + ||\lambda - \lambda_{\mathrm{des}}||^2_{\mathbf{W}_\lambda} \end{aligned} \tag{5}$$

where $\boldsymbol{x}_t$ are either Cartesian end-effector poses ($\boldsymbol{x}_t \in SE(3)$) or the center of mass position ($\boldsymbol{x}_t \in \mathbb{R}^3$). The $\mathbf{W}_t$ are weighting positive definite matrices, P_{null} projects into the null space of all the Cartesian tasks, Joint and Cartesian tasks are related through

$$\ddot{\boldsymbol{x}}_t = \mathbf{J}_t \ddot{\boldsymbol{q}} + \dot{\mathbf{J}}_t \dot{\boldsymbol{q}}, \tag{6}$$

where $\mathbf{J}_t$ is the Jacobian of the unconstrained end-effector or CoM. Desired end-effector and CoM accelerations are computed through

$$\ddot{\boldsymbol{x}}_{t,\text{des}} = \ddot{\boldsymbol{x}}_{t,\text{ref}} + \mathbf{P}_t \left(\boldsymbol{x}_{t,\text{ref}} - \boldsymbol{x}_i\right) + \mathbf{D}_t \left(\dot{\boldsymbol{x}}_{t,\text{ref}} - \dot{\boldsymbol{x}}_t\right). \tag{7}$$

Positive definitve matrices $\mathbf{P}_x$ and $\mathbf{D}_x$ represent stiffness and damping gains of the PD controller, respectively. Desired joint accelerations are specified by

$$\ddot{\boldsymbol{q}}_{\text{des}} = \mathbf{P}_q(\boldsymbol{q}_{\text{ref}} - \boldsymbol{q}) - \mathbf{D}_q \dot{\boldsymbol{q}}.$$

For more details on the solver, see [5].

IV. Task Specific Dynamics

The controller presented in the previous section allows to track any type of Cartesian task. To improve tracking in task space, we introduce an additional feedforward term. Indeed, tracking performance depends on the accuracy of the model. This can be seen in (7), where

$$\ddot{\boldsymbol{x}}_{\text{des}} = \underbrace{\ddot{\boldsymbol{x}}_{\text{ref}}}_{\text{feedforward}} + \underbrace{\mathbf{P}_x \left(\boldsymbol{x}_{\text{ref}} - \boldsymbol{x}\right) + \mathbf{D}_x \left(\dot{\boldsymbol{x}}_{\text{ref}} - \dot{\boldsymbol{x}}\right)}_{\text{feedback}}.$$

If the model were perfect, the contribution of the feedback part would amount to 0. However, in the real world it is not, and the feedback part accounts for the discrepancy. We propose recording the feedback contribution part and adding it in the next repetition of the exact same task (desired motion). Thus, we get

$$\ddot{\boldsymbol{x}}_{\text{des},i} = \underbrace{\ddot{\boldsymbol{x}}_{\text{ref}} + \ddot{\boldsymbol{x}}_{\text{fb},\ i-1}}_{\text{updated feedforward}} + \mathbf{P}_x \left(\boldsymbol{x}_{\text{ref}} - \boldsymbol{x}\right) + \mathbf{D}_x \left(\dot{\boldsymbol{x}}_{\text{ref}} - \dot{\boldsymbol{x}}\right), \tag{8}$$

where

$$\ddot{\boldsymbol{x}}_{\text{fb},\ i-1} = \ddot{\boldsymbol{x}}_{\text{fb},i-2} + \mathbf{P}_x \left(\boldsymbol{x}_{\text{ref}} - \boldsymbol{x}_{i-1}\right) + \mathbf{D}_x \left(\dot{\boldsymbol{x}}_{\text{ref}} - \dot{\boldsymbol{x}}_{i-1}\right). \tag{9}$$

It means that at each iteration, we add to the new feedforward term the previous contributions of the feedback terms, therefore learning the error dynamics. The number of learning iterations is arbitrary, depending on the desired accuracy. However, stability of the learning process needs to be ensured, see [8] for details on ILC. In our experiments, we only used the feedback from 1 previous iteration (i = 2) as it was sufficient to already significantly improve performance.

Unlike [16] or [9], the feedforward part is added in the task-space of the robot, and not in its joint space. It is then combined with the QP-based inverse dynamics controller.

Using the recorded (learned) feedback signal in the next iteration of the same task provides us with an improved feedforward signal, which is task-specific. However, we can build up a database of such signals for different task variations. Thus, the proposed algorithm can significantly correct the discrepancy between the model and the real system. Furthermore, we can use the database to generate an appropriate signal for previously untested tasks and task variations using statistical generalization as in [9].

A. Encoding the Feedforward Signal

The recorded (learned) signal can be encoded in any form. For example, for end-to-end (discrete) tasks, discrete DMPs can be used [14]. The use-case example in this paper is periodic squatting. Because our task is periodic, we chose to encode the signal as a linear combination of radial basis functions (RBF) appropriate for periodic tasks. Using RBFs has the advantage that the signal encoding is compact and the signal itself is inherently filtered. As discussed in [9], this representation allows for computationally light[1] generalization using Gaussian Process Regression (GPR) [23]. A linear combination of RBF as a function of the phase ϕ is given by

$$\ddot{\boldsymbol{x}}_{\text{fb},\ i-1}(\phi) = \frac{\sum_{j=1}^{L} w_j \Gamma_j(\phi)}{\sum_{j=1}^{L} \Gamma_j(\phi)}, \tag{10}$$

where Γ_j denotes the basis function, given by

$$\Gamma_j(\phi) = \exp(h_j(\cos(\phi - c_j) - 1)), \tag{11}$$

w_j is the weight of the j-th basis function, L is the number of basis functions, c_i are centers of the basis functions and $h_i > 0$ their widths. The periodic phase ϕ is determined by the phase oscillator

$$\dot{\phi} = \Omega, \tag{12}$$

where Ω is the frequency of oscillations. While in our use-case the phase is linear because of a constant task frequency, it can change over time and even adapt to external signals [24].

B. Generalization

In the manner of generalizing joint-space feedforward torques [9], we can also generalize the learned task-space CoM accelerations. Having chosen RBF encoding, we can generalize between the weights, for example using GPR. The goal of generalization is to provide us with a function

$$\boldsymbol{F}_{\boldsymbol{D}_b} : \kappa \longmapsto [\boldsymbol{w}] \tag{13}$$

that provides the output in the form of a vector of RBF weights w, given the database of trained feedforward terms $\boldsymbol{D}_b$ and the input, i. e., the query κ. We refer the reader to [9], [23] for details on GPR. This kind of generalization is analog to the one in [9], but in task space.

[1]The calculation of the hyperparameters for GPR is computationally expensive, but it is performed offline. Simple matrix multiplication is performed online.

V. EXPERIMENTAL RESULTS

Experiments were performed on the lower part of the hydraulically actuated, torque controlled, Sarcos humanoid robot. It has in total 17 degrees of freedom (DoFs), with 7 in each leg and three in the torso. The system is depicted in Fig. 1. We used the SL simulation and control environment [25] for evaluation in simulation.

To demonstrate the applicability of the approach we use a periodic squatting task. We compare position error of the robot's CoM without and with the added feedforward term, given by (9). Note that any kind of task, be it periodic or an end-to-end task, can be implemented in the same manner. However, for a different kind of task, some other trajectory encoding might be beneficial.

A. Squatting Simulation

In the first experiment, we performed periodic squatting at a frequency of 0.25 Hz. For the use-case example, squatting was defined as a vertical sinusoidal motion of 3cm amplitude; this range is close to the maximal motion the robot can perform without hitting joint limits. Any other squatting trajectory, obtained for example through motion capture, could also be used.

In the simulation experiments, we use a different dynamic model for the inverse dynamic computation than for the simulation, in order to test the performance of our approach with modeling errors. We changed the model so that there was a 10% error in the mass and inertia matrices of each link. A 20% difference would not achieve meaningful squatting with the given parameter set; we discuss this in Section VI.

Results in Fig. 2 show the difference in tracking error when using the learned feedforward term, and when not using it. Reduction of tracking error is clearly visible when the learned feedforward term is used. Furthermore, when using the learned feedforward term, we can significantly reduce the $\mathbf{P}$ gain (in this case 5X) without visibly increasing the tracking error. This demonstrates that the contribution of the feedback term is low and that increased compliance can be achieved. Before the beginning of squatting (depicted by the black dashed line), the learned feedforward term was a constant value learned for the start of the squat from steady-state squatting. This does not completely match the initial posture of the robot in simulation, which has a small starting randomness built-in. The higher error in the starting posture when using low gains indicates higher compliance of the robot (i.e. the errors of the model induce an error in steady state positions).

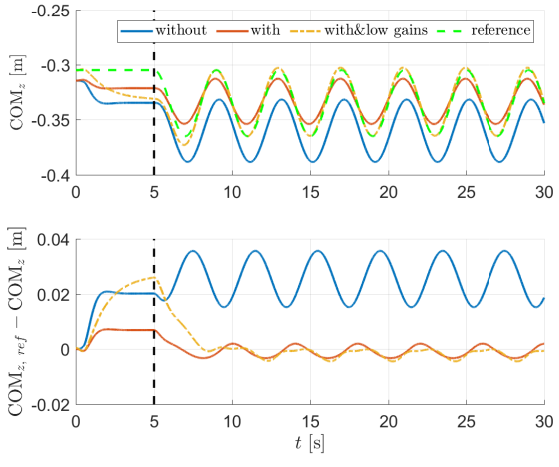


Fig. 2. Top: Center of mass position in the vertical z axis during a squatting experiment without the learned feedforward term (blue), with the learned feedforward term (red), and with the feedforward term but with 5 times reduced $\mathbf{P}$ feedback gains (ocher). Desired CoM motion in dashed-green. Black dashed line depicts the start of squatting. Bottom: Error of CoM_z tracking for all three cases.

The plot in Fig. 3 shows the amplitude of the feedback part of the controller for the same three cases as for Fig. 2. We can again see the difference in the necessary feedback correction. Feedback correction is by an order of magnitude larger if the learned feedforward term is not used. The plot also shows that the encoded feedforward signal (purple) very closely matches the original feedback signal. The matching could be increased, for example, with a higher number of basis functions. In the experiments we used $L = 25$ basis functions; the number was chosen empirically.

B. Generalization to different squatting amplitudes

We performed a generalization experiment over a variation of the task, to show that the approach can be used with generated feedforward signals. We used GPR to generalize the feedforward term for a squatting amplitude of $\kappa = 5$ cm. The database consisted of feedforward terms for different squatting amplitudes[2] $\kappa = 2, 4, 6, 8$ cm. Fig. 4 shows that the generalized feedforward term allowed for very similar, low CoM_z tracking errors as the recorded feedforward term for $\kappa = 5$ cm. While this kind of generalization is similar to the one represented in [9], in our case the generalization was in task space, i. e., generalization was between the weights for 1 DoF. To achieve the same in joint space would require generalization for all 17 DoF of the robot. Our approach therefore leads to a simpler generalization function thanks to the combination of the task-space feedforward term and the inverse dynamics controller.

[2]The database is too small. Typically it would consist of tens of entries, but in this toy example they would be too close together.

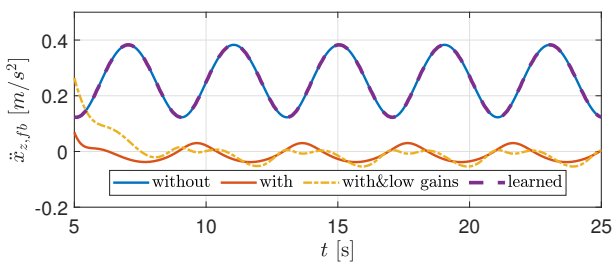


Fig. 3. The value of the feedback term when squatting as defined in the experiment. Without the additional feedforward term in blue, with the additional term in red, with the term but with reduced gains in ocher, and the encoded feedforward signal in dashed purple.

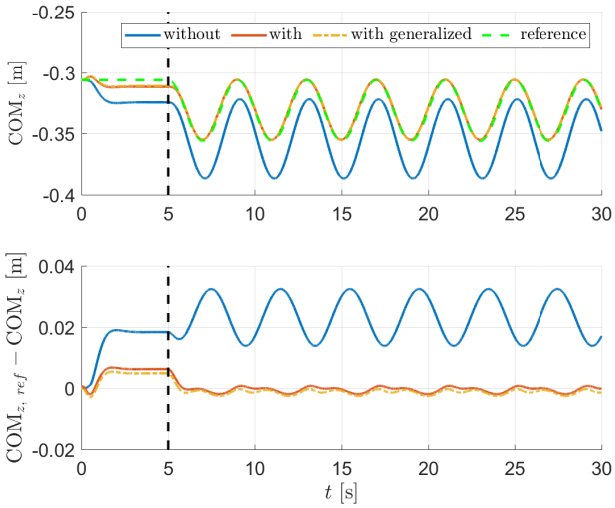


Fig. 4. Top: Center of mass position in the vertical z axis during a squatting experiment without the learned feedforward term (blue), with the learned feedforward term (red), and with the feedforward term generalized from a database (ocher). Desired CoM motion in dashed-green. Black dashed line depicts the start of squatting. Bottom: Error of CoM_z tracking for all three cases.

C. Real Robot Experiments

We performed the same experiment on the real robot. To vary the dynamics of the task, we tested our approach for two different squatting frequencies: 0.25 Hz and 0.5 Hz. The error of CoM_z tracking for both cases is depicted in Fig. 5. We can see in the plots that the error is again significantly reduced for both cases. Small oscillations in the behavior are the consequence of acting on a real system with an imperfect model and feedback signals. CoM position on the real system was estimated using the joint encoders and the kinematics, with assumed flat feet on the ground. Fig. 6 shows a series of still photos depicting the real system performing one squat.

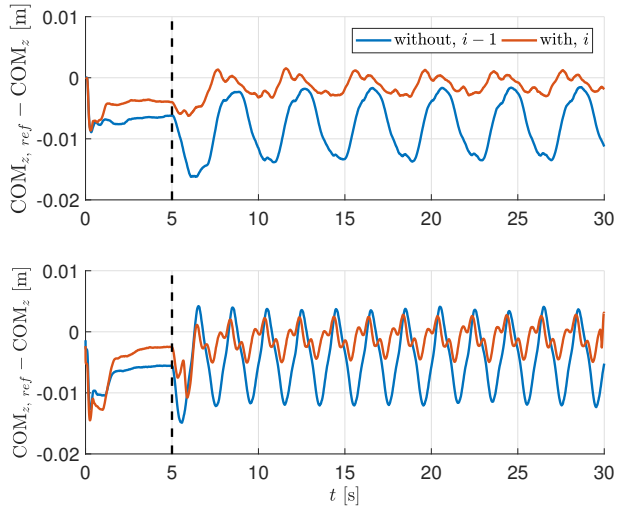


Fig. 5. Real-world error of CoM_z tracking with and without the added term for the squatting experiment at two different squatting frequencies, 0.25 Hz in the top and 0.5 Hz in the bottom. In both plots the results without the added term are in blue, and with the added feedforward term in red.

VI. DISCUSSION

We have proposed a method to learn task-space dynamics which allows to improve task performance while reducing feedback gains. Both the simulation and real robot results show a reduced contribution of the feedback term; a clear improvement of the tracking performance and the possibility to reduce the feedback gains to thus increase the compliance. In the following, we briefly discuss these points.

We first discuss the improvement of the tracking performance and the reduction of the feedback term contribution. The question whether the system behavior is the same with the additional task-specific feedforward term and low feedback gains or without the additional task-specific feedforward term and high gains has previously been studied [26], [27]. An equivalent feedback can always be constructed from the ILC parameters with no additional plant knowledge for proper causal LTI systems, whether or not the ILC includes current-cycle feedback. Our system (8) is not proper causal, therefore complete equivalence cannot be claimed, but the results show a clearly reduced contribution of the feedback term. Herein lies the main advantage of using the proposed approach – lower feedback gains can be used, resulting in increased compliance. Compliance of the system has been recognized as one of the key elements for real-world deployment of robots in unstructured environments, as it provides robustness for unplanned disturbances [18].

Our approach reduces the contribution of the feedback term in a manner similar to an improved dynamic model. As shown in the literature (e. g. in [1]), a dynamic model never completely describes the behavior of a complex system. On a real system, such as the lower part of the Hermes humanoid robot used in this work, unmodeled hydraulic hoses, actuator dynamics, friction and flexibilities can have a significant effect. The originality of our approach is that we improve a task-specific dynamic model. The learned feedback torques for squatting are by default only applicable to squatting. As already outlined in Section III, building up a database is a rather straightforward solution. This has not only been applied to the *model* (for lack of a better word), but also to control policies as a result of optimization. In [28], a database of such control policies is used to warm-start the optimization. A more advanced solution than just building up a database is to use the database to generate feedforward terms for previously untrained situations. Different methods can be applied, for example statistical learning, such as GPR, which was used in a similar manner for joint torques in [9].

As shown in Section V, one of the advantages of the proposed approach of generalization in the task space is the reduced dimensionality of the task. Another advantage of learning and applying the feedforward terms in task space is the similarity it has over different tasks. The proposed approach requires first a working solution, so that the feedforward term can be learned. However, this working solution might be difficult to achieve for complex tasks if the model is not sufficiently accurate. Even our squatting use-case will not work if the model is 20% off. Achieving a complex

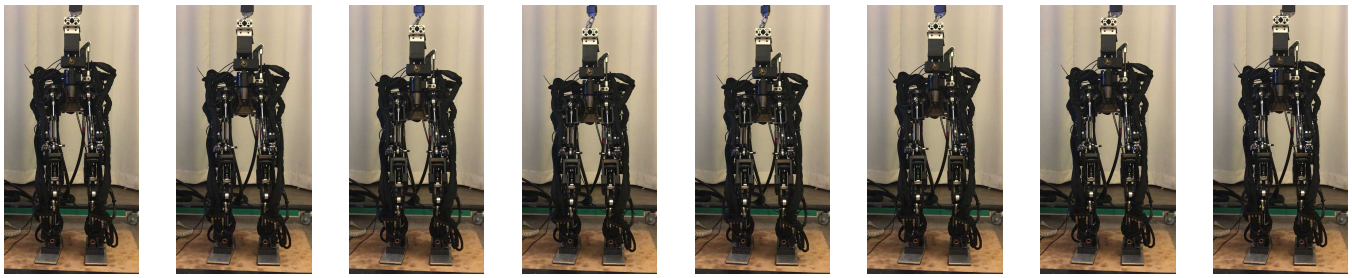


Fig. 6. Still images of the real-world lower part of the Sarcos Hermes robot performing one squat. See also the accompanying video.

task can be challenging but we posit that the feedforward term of a less-complex task could be used to bootstrap the execution of the more complex task. During walking, the CoM position is moving from one side to the other, which is (from the CoM point of view) the same as simply shifting the weight from one side to the other without lifting the feet. With feedforward terms for shifting of the weight from one side to the other, which makes the execution of this task more accurate, it is only one (algorithmic) step to implement stepping in place. This cross-task generalization, however, remains an open research question. Furthermore, such generalization can only be applied over similar tasks, and steps that ensure that the feedforward term does not worsen the solution need to be taken.

VII. CONCLUSION & FUTURE WORK

In this paper we showed that basic iterative learning can be applied to task-space accelerations in order to improve the task execution of a complex, free-floating robot system, controlled with task-space inverse dynamic controllers and that generalization to variations of the task is also possible. Furthermore, it has the potential to improve the behavior of model-based control methods with the application of generalized signals for different task parameters, and possibly even across different task. The latter, however, remains for future work.

REFERENCES

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

[2] P. M. Wensing, S. Kim, and J.-J. E. Slotine, "Linear Matrix Inequalities for Physically Consistent Inertial Parameter Identification: A Statistical Perspective on the Mass Distribution," *IEEE Robotics and Automation Letters*, vol. 3, pp. 60–67, Aug. 2017.

[3] D. Nguyen-Tuong and J. Peters, "Model learning for robot control: a survey," *Cognitive Processing*, vol. 12, pp. 319–340, Nov 2011.

[4] D. W. Franklin and D. M. Wolpert, "Computational mechanisms of sensorimotor control," *Neuron*, vol. 72, no. 3, pp. 425 – 442, 2011.

[5] A. Herzog, N. Rotella, S. Mason, F. Grimminger, S. Schaal, and L. Righetti, "Momentum control with hierarchical inverse dynamics on a torque-controlled humanoid," *Autonomous Robots*, vol. 40, no. 3, pp. 473–491, 2016.

[6] A. Escande, N. Mansard, and P.-B. Wieber, "Hierarchical quadratic programming: Fast online humanoid-robot motion generation," *Int. J. of Robotics Research*, vol. 33, no. 7, pp. pp. 1006–1028, 2014.

[7] S. Kuindersma, F. Permenter, and R. Tedrake, "An efficiently solvable quadratic program for stabilizing dynamic locomotion," in *IEEE Int. Conf. on Rob. and Aut. (ICRA)*, pp. 2589–2594, 2014.

[8] D. A. Bristow, M. Tharayil, and A. G. Alleyne, "A survey of iterative learning control," *IEEE Cont. Sys.*, vol. 26, no. 3, pp. 96–114, 2006.

[9] M. Deniša, A. Gams, A. Ude, and T. Petrič, "Learning compliant movement primitives through demonstration and statistical generalization," *IEEE/ASME Transactions on Mechatronics*, vol. 21, pp. 2581–2594, Oct 2016.

[10] M. Norrlöf, *Iterative Learning Control – Analysis, Design, and Experiments*. PhD thesis, Linköpings Universitet, 2000.

[11] M. Kawato, "Feedback-error-learning neural network for supervised motor learning," in *Advanced Neural Computers* (R. ECKMILLER, ed.), pp. 365 – 372, Amsterdam: North-Holland, 1990.

[12] P. Pastor, M. Kalakrishnan, L. Righetti, and S. Schaal, "Towards associative skill memories," in *12th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pp. 309–315, 2012.

[13] A. Gams, B. Nemec, A. J. Ijspeert, and A. Ude, "Coupling movement primitives: Interaction with the environment and bimanual tasks," *IEEE Transactions on Robotics*, vol. 30, pp. 816–830, Aug 2014.

[14] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical Movement Primitives: Learning Attractor Models for Motor Behaviors.," *Neural computation*, vol. 25, pp. 328–73, 2013.

[15] F. Steinmetz, A. Montebelli, and V. Kyrki, "Simultaneous kinesthetic teaching of positional and force requirements for sequential in-contact tasks," in *IEEE-RAS Int. Conf on Hum. Robots*, pp. 202–209, 2015.

[16] N. S. Pollard, J. K. Hodgins, M. J. Riley, and C. G. Atkeson, "Adapting human motion for the control of a humanoid robot," in *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 2, pp. 1390–1397 vol.2, 2002.

[17] T. Petrič, L. Colasanto, A. Gams, A. Ude, and A. J. Ijspeert, "Bio-inspired learning and database expansion of compliant movement primitives," in *IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pp. 346–351, Nov 2015.

[18] B. J. Stephens and C. G. Atkeson, "Dynamic balance force control for compliant humanoid robots," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 1248–1255, 2010.

[19] A. Herzog, N. Rotella, S. Schaal, and L. Righetti, "Trajectory generation for multi-contact momentum control," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pp. 874–880, 2015.

[20] S. Schaal and C. Atkeson, "Robot juggling: implementation of memory-based learning," *IEEE Cont. Sys.*, vol. 14, pp. 57 – 71, 1994.

[21] S. Levine and P. Abbeel, "Learning neural network policies with guided policy search under unknown dynamics," in *Advances in Neural Information Processing Systems 27* (Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, eds.), pp. 1071–1079, Curran Associates, Inc., 2014.

[22] M. P. Deisenroth and C. E. Rasmussen, "Pilco: A model-based and data-efficient approach to policy search," in *Int. Conf. on Machine Learning*, ICML'11, (USA), pp. 465–472, Omnipress, 2011.

[23] C. Rasmussen and C. Williams, *Gaussian processes for machine learning*. Cambridge, MA, USA: MIT Press, 2006.

[24] A. Gams, A. J. Ijspeert, S. Schaal, and J. Lenarčič, "On-line learning and modulation of periodic movements withnonlinear dynamical systems," *Autonomous Robots*, vol. 27, pp. 3–23, Jul 2009.

[25] S. Schaal, "The sl simulation and real-time control software package," tech. rep., Los Angeles, CA, 2009.

[26] P. B. Goldsmith, "On the equivalence of causal lti iterative learning control and feedback control," *Automatica*, vol. 38, no. 4, pp. 703 – 708, 2002.

[27] M. H. A. Verwoerd, G. Meinsma, and T. J. A. de Vries, "On the use of noncausal lti operators in iterative learning control," in *IEEE Conf. on Decision and Control*, vol. 3, pp. 3362–3366 vol.3, 2002.

[28] N. Mansard, A. Del Prete, M. Geisert, S. Tonneau, and O. Stasse, "Using a Memory of Motion to Efficiently Warm-Start a Nonlinear Predictive Controller," in *IEEE Int. Conf. Rob.Aut.*, 2018.