D5.4 Data-driven system identification method

Grant Agreement No	871767
Project Acronym	ReHyb
Project title	Rehabilitation based on Hybrid neuroprosthesis
Start date of the project	01/01/2020
Duration of the project	48 months
Date of submission	04/10/2022
Version	1.0

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement n° 871767

Disclaimer

The ReHyb Consortium consists of the following partners:

No.	Participant organisation name (Short name)	Country
1	Technical University of Munich (TUM)	DE
2	IUVO S.R.L. (IUVO)	IT
3	Scuola Superiore di Studi Universitari e di Perfezionamento Sant'Anna (SSSA)	IT
4	Össur hf (OSS)	IS
5	TECNALIA Research & Innovation (TECN)	ES
6	Imperial College London (ICL)	UK
7	Institute for Bioengineering of Catalonia (IBEC)	ES
8	Technical University of Denmark (DTU)	DK
9	Stelar Security Technology Law Research UG (STELAR)	DE
10	Schön Klinik Bad Aibling SE & CO. KG (SK)	DE
11	Congregazione Suore Infermiere dell’Addolorata (VALDUCE)	IT

Document History

Version	Date	Status	Authors, Reviewers	Description
0.1	15/06/2022	Draft	Samuel Tesfazgi	Initial structure.
0.2	21/06/2022	Draft	Samuel Tesfazgi	Bullet points for defined structure
0.3	28/06/2022	Draft	Samuel Tesfazgi	Draft of content
0.4	13/07/2022	Draft	Samuel Tesfazgi	Include references
0.5	18/07/2022	Draft	Samuel Tesfazgi	Work in comments
0.6	29/09/2022	Draft	Samuel Tesfazgi	Work in comments of internal reviewer
0.7	30/09/2022	Draft	Samuel Tesfazgi	Work in comments of internal reviewer

Executive Summary

Providing assistance and rehabilitation strategies tailored to each patient is one of the key goals of the EU project *Rehabilitation based on Hybrid Neuroprothesis* (ReHyb). In order to achieve this, highly precise models of both the human as well as the robotic system are needed, which can also be adapted safely. While modeling strategies based on first principles are detailed in the other tasks of work package five, this deliverable specifically focuses on data-driven system identification techniques to augment or generate models.

First, the requirements for system identification techniques are listed from the project's perspective and its desired use cases. Subsequently, the findings are used to derive fundamental technical requirements for the potentially deployed machine learning methods. These requirements include among others safety, real-time capabilities, precision, and physical consistency.

Having identified these requirements, a brief literature review is presented to determine possible learning techniques for system identification. Based on these results, non-parametric models are used as a starting point for the development of data-driven system identification methods with a specific focus on Gaussian processes.

In four case studies the state of the art in Gaussian process-based system identification is furthered. In each study, new methodologies are developed to extend the capabilities of Gaussian processes with a specific focus on the previously identified technical requirements for the ReHyb project. Thereby, we contribute to the state of the art on data-driven system identification techniques for the use in rehabilitation robotics and develop new modeling techniques for the use in ReHyb in this deliverable.

Table of Contents

1 Introduction

Robotic systems are becoming more prominent in different areas of life and are being deployed more often in close proximity to humans. In the ReHyb project, the considered use case deals with health care robotics. In particular, we want to provide a high degree of personalization and adaptation to the specific needs of each individual patient to improve the utility of robot-driven neurorehabilitation. In order to achieve this, computational approaches that provide precise estimates of the patient's neuromechanical state are required to inform the rehabilitation process and control strategy. Therefore, the need arises to model the dynamics and inherent uncertainties underlying the human-robot interaction. Since the interaction dynamics in the case of wearable robotics (exoskeletons) are quite complex, novel data-driven system identification methods are needed to provide reliable estimates in a timely and flexible manner.

1.1 Scope of the deliverable

The goal of the EU project ReHyb is to develop rehabilitation concepts and assistance for activities-of-daily-living (ADL) based on exoskeletons and functional electrical stimulation (FES). To this end, work package 5 (WP5) deals with the development of models for the human-exoskeleton system involving both the physical human-exoskeleton-interaction dynamics and the neuromechanics of the human. The resulting insights can then be used to inform both rehabilitation strategies and the control of the ReHyb system.

When developing such models, specifically in safety-critical health-care scenarios, it is prudent to base them on first principles, i.e., using physical and biological concepts that are well understood and allow the generation of analytical models. These types of models account for most of the descriptions of the ReHyb system and are the subject matter of the other tasks of WP5. However, because of the complexity of the hybrid exoskeleton system, which is further exacerbated due to the involvement of the human, it is not always possible, nor computationally feasible, to model all effects using first principles.

Therefore, the focus of this deliverable is the development of machine learning techniques, which can be used to augment and/or substitute the analytical models where necessary. Here, the proposed learning methods are conceived with the application scenario of the ReHyb project and the consequent requirements in mind. Thus, while the precision, flexibility, and overall performance of the system are important, special focus is given to properties which are paramount for the deployment in health care, such as safety, interpretability, and robustness.

We will start by analyzing the use case and functional requirement of the project in order to derive the technical requirements posed to learning methods. Building on these results, we explore the state of the art of data-driven modeling techniques and analyze their advantages and disadvantages to find the most applicable machine learning techniques. Finally, missing properties in the current state of the art are identified and novel data-driven system identification methods are proposed.

1.2 Relation to other project tasks

To identify the requirements for data-driven system identification methods for the ReHyb project, we confer to the results obtained in WP2. Specifically, deliverable 2.1 (D2.1) was concerned with the analysis of the use cases and the definition of clinical demand, and provides us with the overall rehabilitation needs and interaction scenarios. Additionally, D2.2 identified the functional and system requirements of the ReHyb system, which defines the system capabilities we intend to provide and, thus, the desired performance we have to develop the models for.

Alongside the use case definition given in WP2, WP4 is responsible for the design of the hybrid exoskeletons. The hardware design in WP4 determines the available sensory devices, which also connects to the kind of machine learning methods considered in this deliverable.

Further relations to WP5 can be found in WP3 and WP6. In WP3 tasks T3.4 and T3.5 are concerned with the development of an accurate digital twin of the user (the patient), including user-specific properties. Here, individualized models are required to generate an accurate digital twin, thus requiring the results of WP5 as an input. Similarly, the T3.3 within WP3 is concerned with the estimation of human intention, which can also benefit from precise models of the human dynamics.

In a similar fashion, WP6 requires user-specific models as an input. Here, the control of the hybrid exoskeleton is developed, which in many instances is designed as model-based controllers. Therefore, in order to provide tailored assistance for each individual patient, it is necessary to adapt the models on which the controls are based. This can be done using the data-driven system identification methods described in this deliverable.

1.3 Structure of the deliverable

The deliverable is structured as follows. In Section 2, we will analyze the relevance of data-driven system identification methods from the perspective of the use case requirements defined for the ReHyb project. Given this analysis, Section 3 will derive fundamental technical requirements for the potentially deployed machine learning techniques. In Section 4, we provide a general overview of machine learning methods. The reviewed approaches are analyzed with respect to the previously provided technical requirements. Lastly, the novel, data-driven system identification methods developed during the task are presented in Section 5, and conclusions are drawn in Section 6.

2 Requirements for system identification in ReHyb

The ReHyb project aims to develop a novel rehabilitation system for stroke patients based on an individualized digital user model obtained through multi-modal sensing, and actuation techniques from exoskeleton and FES. From an algorithmic perspective the main advances of this system are the intrinsically safe low-level control scheme and advanced cognitive abilities that autonomously analyze the user in a goal-oriented robot-assisted task. Additionally, adaptive control algorithms are deployed in order to achieve a patient-specific, assist-as-needed device, which maximizes orthotic and training efficiency during clinical and home-based rehabilitation and offers a pleasant user experience during ADLs. A detailed description of the objectives and respective KPIs in the ReHyb project are listed in Table 1. which also shows the objectives in direct relation to the deliverable.

In order to achieve an individualized, patient-adaptive, and safe physical interaction, the ReHyb system needs to be able to store and analyze the user data, facilitate an application of tailor-made treatment plans, and support the clinical management of the patient by communicating knowledge with the caregivers. All of this is implemented via the general control architecture shown in Figure 2.1, which provides a unifying framework for rehabilitation and ADL scenarios with the hybrid exoskeleton.

Here, the high-level architecture extracts end-user characteristics and constructs a digital twin via physical interaction with the patient. This model is then used for monitoring rehabilitation progress as well as managing the interaction strategies executed by the system by generating desired task goals. To this end, the high-level system components require quantitative feedback on the mental and physical state of the patient, i.e., fatigue or spasticity, which requires the use of computational models describing complex neuromechanical properties of the human.

Table 1 - List of objectives and KPIs of the project relevant to deliverable

Objectives	Goals	KPIs	Relevance to D5.4
1) Interfacing digital user twin with robotics technology	• To develop a generic user reference model, and personalize it to generate a digital user twin through sensing and actuation technologies. • To develop estimation techniques for internal states such as neuromechanics and spasticity of the muscles. • To effectively handle the dynamics and uncertainty arising from the time-varying, nonlinear mapping of the FES and user performance.	○ Neuromechanical state and muscle spasticity estimated during pHRI with accuracy higher than 70%. ○ The relative modelling error for the combined FES multi-joint hand muscle system is less than 35%.	› By developing novel learning-based system identification methods, the description of the neuormechanical model and state estimation becomes more precise. Therefore, this can be used as an input to precise, analytical spasticity estimation methods developed in T5.2.
2) Cognitive system for sympathetic HR interaction	• Online planning and decision-making of the hybrid control system to be intuitive and sympathetic to the user. • Supporting HRI with patients by integrating multiple interaction modalities in a high-level control system.	○ Non-verbally interaction with a user in more than 70% of the pre-defined use cases. ○ Positive evaluation on the sympathetic interaction by more than 90% of the users.	› Ensuring a seamless and pleasant interaction between human and robot necessitates precise models. › High-level interaction modalities such as game theoretic approaches require the agents to be described as dynamics systems.
3) Cooperative control strategies under uncertain system dynamics	• To develop a shared control strategy capable of handling unknown influences due to highly nonlinear, time-varying, and uncertain physical interactions. • To research novel modular control scheme for self-regulation of the hybrid exoskeleton, which allows safe execution of the assistive force.	○ Controller adapting online to time-varying changes for 80% of use cases. ○ Theoretical control guarantees adherence to safety constraints in the pre-defined use cases.	› Online adaptation of controllers can be achieved by updating models from data. › Deployment of appropriate machine learning techniques allows the derivation of theoretical model error guarantees, which can be used for control.

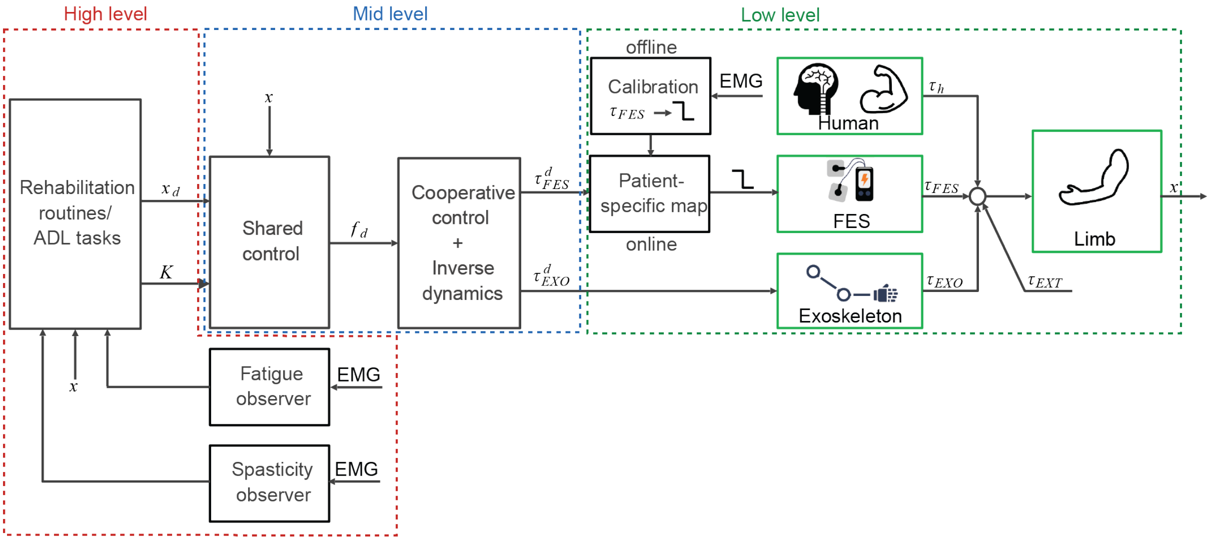

Figure 2.1: General control architecture for the ReHyb system with conceptual division in high-, mid-, and low-level components.

Once the interaction strategies are defined, the middle-level architecture deals with allocating the control between the user and the hybrid exoskeleton, as well as the distribution of load for FES and the robotic system. For the shared control block, there are two conceptual approaches. First, an adaptive impedance control that generated behavior by modulating the interaction with the environment. Second, a game-theoretical framework, which achieves optimal interaction between the human and hybrid exoskeleton by a cost-function representation of both agents. While the game-theoretic approach potentially leads to an improved interaction and provides more flexibility in generating interaction paradigms, such as assist-as-needed, it also assumes an existing model of the human-exoskeleton dynamics, which is generally difficult to obtain due to complex interaction dynamics. On the other hand, the load-distribution, represented by the cooperative control block in Figure 2.1, assigns the desired force from the shared controller to both the FES and exoskeleton. In order to do this effectively, again a dynamics model is required as represented by the ‘inverse dynamics’ component in Figure 2.1.

Lastly, the low-level modules of the general control architecture facilitates the execution of the computed assistance on an actuator level, i.e., the torques generated by the FES and exoskeleton device. Here, particular challenges arise since not all of the required information to do so is observable and instead need to be estimated. For an appropriate generation of the exoskeleton torques during the physical human-robot-interaction (pHRI) it is necessary to know the volitional torques that the human provides. As there are no expensive force-torque sensors present to measure the interaction forces, the use of model-based torque observers becomes necessary. This torque observer requires the exoskeleton dynamic model, motor dynamics, and human arm dynamics. Precise models are highly important for a successful torque estimation, which are challenging to obtain because of complex behaviors, e.g., due to friction. On the other hand, a patient-specific map is needed to convert desired torques to a tangible FES stimulation pattern. This mapping has to be learned, as there are no viable analytical models to describe the input-output relation. Additionally, the learning process should ideally be done online due to time-varying behaviors such as muscle-fatigue.

Given the above described embedding of personalized models in different facets of the general architecture and the underlying functional requirements for the overall ReHyb system, it is possible to derive technical requirements for the deployed data-driven system identification methods. These fundamental requirements are detailed in the following section.

3 Technical requirements

In order to achieve the above-described objectives and goals in the ReHyb project, it is necessary to utilize personalized models of the patient. Given the particular use case and the functional requirements, certain technical requirements emerge for the deployed data-driven system identification methods in order to ensure that the resulting models are suitable for the envisioned application.

Furthermore, the encountered identification challenges, and consequently the prioritization of requirements, change depending on the variables of interest. In the following, a table of properties that need to be estimated in a model-based fashion is provided. For each of the variables of interest the key challenges are listed and technical requirements for the potential learning method are derived. For example, both spasticity and fatigue are time-varying phenomena, since the state of the patient changes during the course of a training session. Therefore, real-time capable learning techniques are needed in order to cope with this challenge. Differently, the modelling of human neuromechanics for example is more useful, if the results are interpretable for the clinicians, hence, requiring the learning methods to be physically consistent. A more detailed overview is given in Table 2. The derived technical requirements are expanded on in Subsections 3.1 to 3.5. Furthermore, the technical requirements identified in this section are conferred to in case studies, which will be presented in Section 5.

Table 2 - Variables of interest, their respective key challenges from a learning perspective and the derived technical requirements

Variable of interest	Key challenges	Technical requirements
• Spasticity	○ Time-varying	› Real-time capabilities (cf. 3.2)
• Fatigue	○ Time-varying	› Real-time capabilities (cf. 3.2)
• Human-exoskeleton dynamics	○ Highly nonlinear ○ Uncertainties	› Safety and robustness (cf. 3.1) › Expressiveness (cf. 3.3)
• Human neuromechanics	○ Interpretability for clinicians	› Physical consistency (cf. 3.5)
• Learning for ADL scenarios	○ Portable system	› Resource constraints (cf. 3.4)
• Human volitional torque	○ Short calibration	› Data-efficiency (cf. 3.2)
• FES stimulation mapping	○ Low amplitudes ○ Time-varying ○ Short calibration	› Safety and robustness (cf. 3.1) › Real-time capabilities (cf. 3.2) › Data-efficiency (cf. 3.2)

3.1 Safety and robustness of learned models

The most important requirement to consider for the used models is that the safety of the overall system is not compromised. Therefore, machine learning methods are preferred that, besides the mean prediction, also provide upper bounds on the model error. These confidence bounds allow to interpret the learned dynamics in a robust form, which facilitates the development of algorithms with safety guarantees. Since the models are used, among others, as an input to the control of the hybrid exoskeleton, facilitating safety is paramount.

3.2 Real-time capable and data-efficient learning

In order to facilitate an intuitive interaction of the ReHyb system with the human it is crucial to estimate the current state and intention of the user. However, since the system in question is time-varying due to the involvement of the human, it is necessary to continuously update the deployed models with new measurements to ensure that the produced estimates properly reflect reality. Here, time-varying behaviors are introduced for example due to fatigue of the patient or commencing spasticity. Therefore, one of the key requirements set to the considered data-driven system identification methods is that they allow for close to real-time updates and prediction rates of the learned model. As a consequence, the deployed machine learning techniques also have to be able to produce viable estimates even with very little training data.

3.3 Flexibility and expressiveness of data-driven modelling

For most of the required input-outputs relations in the system it is desired to model them using first principles. This is reasonable as analytical models generalize well to previously unobserved situations and inherently contain desirable properties, such as, conservation of energy. The use and development of these models based on first principles is attempted in T5.1 and T5.2 of WP5. However, as there are dynamics present in the system that, due to their complexity and limited understanding, cannot be described with analytical models, data-driven system identification methods are to be used. Consequently, the machine learning techniques deployed in this deliverable have to be very flexible and expressive by design, as they would not be able to explain the aforementioned complex behaviors otherwise.

3.4 Learning with computational and memory constraints

Besides the application for clinical rehabilitation of stroke patients, which may take place in a stationary setting, the envisioned ReHyb system should also be deployed for assistance during ADL at home. These ADL scenarios ideally take place in a non-stationary manner to maximize the utility of the system and the benefit for the user. However, making the system portable produces several issues from a modelling perspective. On the one hand, it increases the complexity, since the environment becomes less structured and therefore more uncertain. This issue is exacerbated by the fact that a portable system needs to be lightweight and therefore the resulting hardware limitations automatically lead to restrictions regarding computational and memory resources of the system. Learning models, which can be used for control in safety-critical systems under these resource constraints is a challenging problem.

3.5 Interpretability and physical consistency

As stated above, models based on first principles are inherently beneficial and contain desirable properties. Besides the fact that they generalize well, they also have the added benefit that they are easily interpretable and guarantee physical consistency. This is particularly important in the use case of health care, as the utility of deployed models is directly linked to their reliability and intelligibility of the models. Additionally, interpretability increases the acceptance of model-based computational rehabilitation methods, since the clinicians using the models can easily integrate the model estimates into their decision-making process. Therefore, deploying system identification techniques that can provide interpretable and physically consistent estimates is very beneficial for the ReHyb system.

3.6 Conclusions from requirement evaluation

Data-driven system identification methods enable us to deal with a system that entail a lot of complexity and are operated under uncertain conditions, which is exactly the case in the ReHyb project. Due to the physical human-exoskeleton-interaction, the use of FES, and the uncertain behavior of the human, the dynamics governing the system become highly nonlinear, time-varying and uncertain. While this level of complexity is challenging from a modelling perspective, it also allows the ReHyb project to tackle use cases that would otherwise not be achievable.

However, in order for the machine learning techniques to be suitable for the particular use cases considered in the ReHyb project, certain technical requirements have to be considered, which have been described in detail above. Some of these requirements may be understood as conditions that have be fulfilled at all times and for all models, while others rather depend on the specific functional use of the model. Safety and robustness, for instance, is a hard constraint that each model should fulfil independent of the use case in the overall architecture. Differently, interpretability is more important when it comes to the modeling of human neuromechanics and may be less important in other aspects.

4 Machine learning methods

Given the technical requirements identified in the previous section, we survey the literature for different machine learning techniques and their use for rehabilitation robotics in this section. A particular focus of this brief overview is, how well the methods apply for the use in the ReHyb project and how to advance existing methods to maximize their utility for our use case.

4.1 Learning-based modelling techniques

Generally, we differentiate between three principal learning-based modelling techniques to perform system identification [1]:

- **Parametric models**: Here, a predefined structure is employed for the system and the learning process is limited to learning the parameters of the model. The functions giving structure to the model can either be defined by first principles, such as an Euler-Lagrange system for a robot arm, or differently by generic feature functions like radial basis functions.
- **Neural networks (NNs):** Differently to standard parametric models, NNs do not impose a specific structure in modelling the system at hand. However, in principle NNs can be understood as a kind of parametric model, since the system is completely described by learning appropriate parameters/weights. Typically, NNs have a lot of parameters, which is especially the case in deep NNs (DNNs), where the models are very large.
- **Nonparametric models:** Lastly, we consider nonparametric models, sometimes also referred to by the term data-driven models. Data-driven models also avoid assumptions regarding the structure of the system but instead build structure from data. Here, particular focus will be given to Gaussian processes (GPs) as an instance of data-driven models.

In principle, each of the presented machine learning model classes have their distinct advantages and disadvantages. For example, the imposed structure in parametric models greatly simplifies the learning process, thereby, allowing a straight-forward way of incrementally updating the learned model with new data. A common approach to do so is recursive least-squares regression [2], which can be used to update the parametric model in real-time. However, while this strict structure of parametric models enables online learning and fast adaptations to a time-varying system, it comes at the cost that the models are usually inaccurate, since they cannot capture the complexity of the real system. Since the ReHyb system is a particularly complex one, more sophisticated estimation methods need to be considered here.

One approach to obtain precise models from data, even if prior knowledge of the system is not available, are neural networks. Recently, NN-based modelling has gained popularity due to their impressive performance in describing highly nonlinear systems. Besides the use in previously impossible thought tasks, such as playing GO [3], NN are also deployed in rehabilitation robotics. In [4] for example, the authors used an adaptive impedance controller with individualized, NN-based adjustment of the control parameters to improve tracking performance. The advantage of this kind of adaptive control scheme is that the NN can learn from patient data while also being able to handle complex dynamics of the human-robot interaction. Differently, deep learning has also been used in [5] for automatic assessment of rehabilitation progress by training the DNN on movement indicators. While, NNs are versatile and expressive enough to handle these different use cases, there are some limitations that hinder the deployment in the ReHyb project: First, to achieve a good performance a large amount of data is required, which is typically difficult to obtain if a human is in the loop, since it cannot be simulated properly, and data acquisition is time-consuming and cumbersome. Second, while NNs are computational efficient when it comes to predictions, the training process is inefficient, therefore, limiting the possibility of providing online adaptations. Lastly, NNs tend to have overconfident estimation results and inherently do not provide measures of uncertainty for their predictions. Hence, without probabilistic confidence bounds it is difficult to develop robust algorithms with safety guarantees using NN-based models, which is particularly important in the ReHyb context.

While NNs are a powerful modeling technique due to their expressiveness, they are not the only viable learning method to model complex systems. Data-driven modelling techniques, GPs in particular, have been shown to perform similarly well to NNs w.r.t. modeling nonlinear dynamical systems [6]. Besides this high level of flexibility, GPs have the added advantage that the learning process is data-efficient, with high performances being achieved even with small training data sets [7]. This is particularly useful when working with humans, since an extensive data acquisition scheme reduces the utility of the robotic device greatly. Additionally, GPs allow to address the key technical requirement derived in the last section regarding the safety and robustness of learned models. Here, one can make use of the fact that confidence bounds for the GP predictions can be derived, which guarantee that the model error does not exceed a certain range with high probability [8] [9]. This allows the interpretation of learned models in a robust manner and facilitates the development of control algorithms with safety guarantees [10]. Furthermore, the risk of overconfident predictions is reduced, as the estimate does not only include a mean value but also the accompanying variance. In clinical use cases this is especially advantageous, because a clinician can make a well-informed decision based on a complete picture of the learning-based estimates. However, certain drawbacks still prevent the direct use of GPs in the ReHyb context. One issue is the high computational complexity of GPs, which grows quadratically in the amount of data points. This limits the utility of GPs, if continuous learning is desirable or if the target system has computational constraints. Also, it prevents online learning completely. One additional downside of data-driven modeling techniques, such as GPs, is that the structure of the model is purely built from data. While this allows for a lot of flexibility and expressiveness, it also leads to models that are not guaranteed to be physically consistent. Therefore, even though GP-based models may be viable in principle, there are relevant shortcomings that still need to be addressed.

4.2 Current limitations

All of the learning-based system identification techniques presented in the previous subsection have their unique advantages and disadvantages. While parametric methods do allow for the most straight-forward implementation from a computational and analytical perspective, their limited expressiveness does prevent them from further considerations in this task. Both NNs and GPs can potentially deal with the complexity of the ReHyb system, however, only GPs allow for a robust and safe integration of the learned model into the overall control architecture. Therefore, GPs will be considered for further developments in this deliverable.

5 Development of novel data-driven system identification methods

As described above, GPs have many beneficial properties, which make them suitable for the data-driven system identification in the ReHyb project. However, in order to increase their utility, certain challenges still need to be overcome. In the following we will address some of the problems with research developments performed within T5.4.

5.1 Case study 1 - Real-time capable and data-efficient learning [11]

Many neuromechanical parameters that describe the state of the patient, such as spasticity or fatigue, are time-varying and therefore require continuous, online estimations and model updates. However, one of the key challenges with GPs is the increasing computational complexity in the amount of data points, which limits their application in online learning contexts. Previously, modifications have been proposed, such as event-triggered online learning where data points are only added whenever the uncertainty in the system becomes too large [12] or sparse GPs [13], which sort out non-informative training points. Nonetheless, these approaches only aim at reducing the amount of data points in the training process and do not address the computational complexity problem at its root. However, recently a novel method has been introduced [14], which enables learning and predicting online.

In this work, for the first time, we deploy the novel method experimentally in a user study in which participants perform a rehabilitation exercise. Generally, in these kinds of settings, the required personalization is achieved through manual tuning by clinicians, which is cumbersome and error-prone. In this work we propose a novel online learning architecture, which is able to personalize the provided assistance at run time to each individual user. To this end, we deploy Gaussian process-based online learning with previously unseen prediction and update rates. Finally, we evaluate our method in an experimental user study, where the learning model-based controller is shown to provide personalized control, while also obtaining safe interaction forces.

5.2 Case study 2 - Flexibility and precision of data-driven modelling methods [15]

Learning in a fast and data-efficient manner has been the focus of the previous case study. However, when trying to model highly complex behaviors, i.e., the volitional torque provided by a patient during a rehabilitation exercise, a high level of flexibility and precision is expected. Particularly in the ReHyb use case, this is a challenging problem because labeled training data is sparse and expensive, while high prediction accuracy is required from models of these dynamical systems. In order to increase the prediction performance of these models, abstract prior knowledge such as stability, should be included in the learning approach. This stability property can be verified by the use of Lyapunov functions, which represent a kind of energy measure within the dynamical system.

One of the key challenges is to ensure sufficient flexibility of the models, which is typically limited by the usage of parametric Lyapunov functions to guarantee stability. Therefore, we derive an approach to learn a nonparametric Lyapunov function based on GP regression from data in this work. Furthermore, we learn a nonparametric Gaussian process state-space model from the data and show that it is capable of reproducing observed data exactly. Finally, the flexibility and efficiency of our approach is demonstrated on the benchmark problem of learning handwriting motions from a real-world dataset, where our approach achieves almost exact reproduction of the training data.

5.3 Case study 3 - Learning with computational and memory constraints [16]

During ADL scenarios, one of the most difficult challenges is in guaranteeing safety while dealing with the highly uncertain environment and the tight resource constraints imposed by the system. For a safety-critical system, such as the ReHyb system, to operate properly in unknown environments it needs to be able to adapt quickly. This can be achieved by inferring a model online from the data stream generated during operation. Here, GP-based learning is particularly well suited for safety-critical applications as it ensures bounded prediction errors.

However, while there exist computationally efficient approximations for online inference, these approaches lack guarantees for the prediction error and have high memory requirements. Hence, these methods are not applicable to safety-critical systems with tight memory constraints. These constraints are present however in a lightweight, fully portable assistive devices.

Therefore, in this work, we propose a novel networked online learning approach based on Gaussian process regression, which addresses the issue of limited local resources by employing remote data management in the cloud. Our approach formally guarantees a bounded tracking error with high probability, which is exploited to identify the most relevant data to achieve a pre-defined performance. We further propose an effective data transmission scheme between the local system and the cloud, which takes bandwidth limitations and time delay of the transmission channel into account. The effectiveness of the proposed method is successfully demonstrated in a human-exoskeleton simulation.

5.4 Case study 4 - Learning with GPs with physical consistency [17]

In the particular use case of learning human neuromechanics, ensuring the physical consistency of learned models is equally as important as achieving high prediction accuracies. Besides guiding the learning process and generating reasonable results at all times, physical consistency also facilitates the interpretability of learned models. In sensitive use cases such as clinical settings this is a necessary requirement to obtain acceptance for new methods and facilitate their deployment in conjunction with current therapy practices.

Therefore, we propose a novel, physically consistent GP enabling the identification of uncertain Lagrangian systems, which can be used to model human and robots alike. The solution space is tailored according to the energy components of the Lagrangian, thereby, analytically guaranteeing physical and mathematical properties such as energy conservation. For the approach only differential input-to-output measurements of the function map are required while Gaussian noise is permitted in torques, velocities, and accelerations. We demonstrate the effectiveness of the approach in numerical simulation.

6 Conclusions

In this deliverable, we have presented data-driven system identification methods to estimate variables of interest for the ReHyb exoskeleton including FES and human neuromechanics. Since the other tasks in the WP are focused on generating models based on first principles, here, particular focus was given to improving and augmenting these models using machine learning. To this end, first the technical requirements for the deployment of machine learning techniques are derived and mapped to the respective variables of interest. Subsequently, the literature is surveyed for suitable methods with the result that GPs are identified as a favorable technique due to their model error bounds, which facilitate the development of safe and robust model-based algorithms.

However, since the state of the art in Gaussian process literature does not provide methods that are able to meet all the technical requirements identified for the ReHyb use case, multiple extensions are proposed in this deliverable. These contributions to the current state of the art are detailed and evaluated in four scientific case studies, which are shown in the appendix of this deliverable. With these novel GP-algorithms we are able to develop data-driven system identification methods, which are suitable for the deployment in the ReHyb project and facilitate a higher degree of personalization for all model-based components of the system.

7 Definitions, Acronyms and Abbreviations

Acronyms abbreviations	Description
ReHyb	Rehabilitation based on Hybrid neuroprosthesis
T	Task
D	Deliverable
FES	Functional Electrical Stimulation
ADL	Activities of Daily Living
WP	Work Package
KPI	Key Performance Indicator
pHRI	Physical human-robot interaction
NN	Neural Network
DNN	Deep Neural Network
GP	Gaussian Process

8 Bibliography

[1] L. Ljung, System Identification, Boston: Birkhäuser, 1998.

[2] M. H. Hayes, Statistical Digital Signal Processing and Modeling, Wiley, 1996.

[3] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. v.d. Driessche, T. Graepel and D. Hassabis, "Mastering the game of Go without human knowledge," *Nature,* vol. 550, pp. 354-359, 2017.

[4] T. Tsuji and Y. Tanaka, "Tracking control properties of human-robotic systems based on impedance control," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans,* vol. 35, pp. 523-535, 2005.

[5] Y. Liao, A. Vakanski and M. Xian, "A Deep Learning Framework for Assessing Physical Rehabilitation Exercises," *IEEE Transactions on Neural Systems Rehabilitation Engineering,* vol. 28, pp. 468-477, 2020.

[6] J. Kocijan, B. Banko, B. Likar, A. Girard, R. Murray-Smith and C. Rasmussen, "A case based comparison of identification with neural network and Gaussian process models," in *Intelligent Control Systems and Signal Processing: ICONS*, 2003.

[7] C. E. Rasmussen and C. K. I. Williams, Gaussian Processes for Machine Learning, MIT Press, 2006.

[8] A. Lederer, J. Umlauft and S. Hirche, "Uniform Error Bounds for Gaussian Process Regression with Application to Safe Control," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019.

[9] N. Srinivas, A. Krause, S. M. Kakade and M. W. Seeger, "Information-Theoretic Regret Bounds for Gaussian Process Optimization in the Bandit Setting," *IEEE Transactions on Information Theory,* vol. 58, pp. 3250-3265, 2012.

[10] J. Umlauft, T. Beckers, M. Kimmel and S. Hirche, "Feedback linearization using Gaussian processes," in *IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017.

[11] S. Tesfazgi, A. Lederer, J. F. Kunz, A. J. Ordóñez-Conejo and S. Hirche, "Personalized Rehabilitation Robotics based on Online Learning Control," in *https://arxiv.org/abs/2110.00481*, 2021.

[12] J. Umlauft and S. Hirche, "Feedback Linearization Based on Gaussian Processes With Event-Triggered Online Learning," *IEEE Transactions on Automatic Control,* vol. 65, pp. 4154-4169, 2020.

[13] J. Quinonero-Candela and C. E. Rasmussen, "A Unifying View of Sparse Approximate Gaussian Process Regression," *Journal of Machine Learning Research,* vol. 6, p. 1939–1959, 2005.

[14] A. Lederer, A. J. O. Conejo, K. A. Maier, W. Xiao, J. Umlauft and S. Hirche, "Gaussian Process-Based Real-Time Learning for Safety Critical Applications," in *Proceedings of the 38th International Conference on Machine Learning*, 2021.

[15] W. Xiao, A. Lederer and S. Hirche, "Learning Stable Nonparametric Dynamical Systems with Gaussian Process Regression," in *IFAC-PapersOnLine*, 2020.

[16] A. Lederer, M. Zhang, S. Tesfazgi and S. Hirche, "Networked Online Learning for Control of Safety-Critical Resource-Constrained Systems based on Gaussian Processes," in *https://arxiv.org/abs/2202.11491*, 2022.

[17] G. Evangelisti and S. Hirche, "Physically Consistent Learning of Conservative Lagrangian Systems with Gaussian Processes," in *submitted to 61st IEEE Conference on Decision and Control*, 2022.

Personalized Rehabilitation Robotics based on Online Learning Control

Samuel Tesfazgi[1]*, Armin Lederer[1]*, Johannes F. Kunz, Alejandro J. Ordóñez Conejo, Sandra Hirche[1]

Abstract— **The use of rehabilitation robotics in clinical applications gains increasing importance, due to therapeutic benefits and the ability to alleviate labor-intensive works. However, their practical utility is dependent on the deployment of appropriate control algorithms, which adapt the level of task-assistance according to each individual patient's need. Generally, the required personalization is achieved through manual tuning by clinicians, which is cumbersome and error-prone. In this work we propose a novel online learning control architecture, which is able to personalize the control force at run time to each individual user. To this end, we deploy Gaussian process-based online learning with previously unseen prediction and update rates. Finally, we evaluate our method in an experimental user study, where the learning controller is shown to provide personalized control, while also obtaining safe interaction forces.**

I. INTRODUCTION

In recent years neurological disorders have become more dominant with an estimate of over 16 million people suffering a first stroke each year [1]. Therefore, an urgent need for rehabilitation treatments arises to ensure the quality of living for such patients. In particular high-intensity and repetition training has been shown to produce the most promising recovery results [2]. Due to these requirements, effective rehabilitation is labor intensive and both patients and healthcare professionals can benefit greatly from robotic-assisted rehabilitation strategies [3]. However, the control of these devices presents certain challenges, which can limit their applicability in practice. Different factors, such as level of assistance, patient engagement and task success, have to be considered when designing the controller. These requirements are particularly difficult to fulfill, due to the uncertain interaction dynamics between human and robot. This problem is exacerbated by the variety of patient behaviors and needs, which require a high degree of personalization.

A control approach that is widely used in the literature is impedance control, which has previously been shown to be applicable for robot-based arm rehabilitation [4]. Impedance control is particular popular for human-robot interaction, since it provides compliant behavior for appropriately chosen parameters, therefore, ensuring limited and safe interaction forces. In [5], a position-dependent stiffness is used to assist rehabilitation tasks, where the stability of the human-robot interaction is guaranteed using system passivity. Differently, in [6], a desired impedance model is defined, which is subsequently achieved through an iterative learning scheme. However, despite these application examples of impedance control for robot-based neurorehabilitation, they generally hinge on properly chosen impedance parameters, which first have to be tuned by the manufacturer a-priori and are then readjusted by clinicians [5]. This procedure is time-consuming and error prone, due to the variety of target personas. Additionally, the tuning needs to be performed cautiously in order to retain the compliant behavior needed for safe interaction. To address these issues other works have developed control architectures that specifically take the complete dynamics into consideration, e.g., via feed-forward control and disturbance observers [7]. While this approach decreases interaction forces, it requires measurements of the patient-robot interaction wrench at each interaction point, such that undesirable force-torque sensors become necessary, which increase system costs and may even lead to stability concerns [8].

Alternatively, learning-based controllers can be deployed to fulfill the personalization requirements. To deal with the unknown interaction model and to adapt the patient-robot interaction, He et al. [9] propose a neural network control approach for rehabilitation robotics. Differently, the authors in [10] focus on obtaining human motor control models during physical interaction by deploying Gaussian processes (GPs) to learn the human arm impedance. Despite these approaches showing promising results, they cannot account for behavior changes of the patient, i.e., due to fatigue, since the learning is performed offline. However, facilitating this adaptation can be highly beneficial for the personalization of control strategies in rehabilitation robotics [11].

In this work we present a novel learning-based control architecture that facilitates highly personalized assistance in a human-robot collaboration task, while requiring no parameter tuning. To the best of our knowledge, this is the first time that a GP is used directly in the control loop to generate assistive forces during physical human-robot interaction. Additionally, we evaluate our learning-based control architecture experimentally, where update and prediction rates with online-generated data are achieved that are orders of magnitude higher than previous GP approaches. The remainder of the paper is structured as follows: First the problem statement is introduced in Section II, while the online learning approach is presented in Section III. Subsequently, in Section IV, the online learning control architecture is proposed. Finally, the evaluation of our method follows in Section V.

II. PROBLEM FORMULATION

We model the physical-human robot interaction in rehabilitation robotics using Euler-Lagrange systems of the form

$$\boldsymbol{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{f}_i(\boldsymbol{x}) = \boldsymbol{u}, \qquad (1)$$

[1] Samuel Tesfazgi, Armin Lederer and Sandra Hirche are with the Chair of Information-oriented Control (ITR), Department of Electrical and Computer Engineering, Technical University of Munich, Germany {armin.lederer,samuel.tesfazgi,hirche}@tum.de

*These authors contributed equally.

where $\boldsymbol{x} = \begin{bmatrix}\boldsymbol{q}^T & \dot{\boldsymbol{q}}^T & \ddot{\boldsymbol{q}}^T \; t\end{bmatrix}^T$ is the concatenation of joint angles $\boldsymbol{q} \in \mathbb{R}^d$, angular velocities $\dot{\boldsymbol{q}} \in \mathbb{R}^d$, angular accelerations $\ddot{\boldsymbol{q}} \in \mathbb{R}^d$, and the time $t \in \mathbb{R}_{0,+}$. The matrix $\boldsymbol{H} : \mathbb{R}^d \to \mathbb{R}^{d\times d}$ denotes the symmetric and positive definite generalized inertia of the robotic system, $\boldsymbol{C} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d\times d}$ is the generalized Coriolis matrix, $\boldsymbol{g} : \mathbb{R}^d \to \mathbb{R}^d$ are the torques resulting from gravitation, $\boldsymbol{f}_i : \mathbb{R}^{3d+1} \to \mathbb{R}^d$ describes the interaction torques generated by the i-th patient, and $\boldsymbol{u} \in \mathbb{R}^d$ are torques applied to the system as control input.

Remark 1: We account for the intra-patient variation of interaction dynamics, e.g., caused by unobserved internal dynamics in the patient such as fatigue, by considering time-dependent functions $\boldsymbol{f}_i(\cdot)$. Hence, all behavioral changes of the human are modeled through time-dependency.

In order to reflect the practical availability of models, we make the following assumption.

Assumption 1: All parameters of the robotic system are known, hence, $\boldsymbol{H}(\cdot)$, $\boldsymbol{C}(\cdot,\cdot)$ and $\boldsymbol{g}(\cdot)$ are available. In contrast, the individual dynamics of patients $\boldsymbol{f}_i(\cdot)$ are unknown.

This assumption reflects the fact that accurate models of robotic systems can typically be obtained using classical identification procedures [12], which can be directly applied in control design. In contrast, the identification of models of human motor dynamics in physical interaction is a challenging problem and often limited to simple scenarios [10], making them generally inapplicable in rehabilitation robotics.

In order to overcome these limitations of conventional modeling techniques, we employ a non-parametric, data-driven approach for learning the human induced dynamics $\boldsymbol{f}_i(\cdot)$. For the inference of an individual model for each patient, we consider access to the following measurements.

Assumption 2: The control input $\boldsymbol{u}$, the joint angles $\boldsymbol{q}$, and the angular velocities $\dot{\boldsymbol{q}}$ and accelerations $\ddot{\boldsymbol{q}}$ of each patient can be observed for learning a personalized model.

As the control input $\boldsymbol{u}$ is determined by the employed control law, it can be directly observed. The joint angles $\boldsymbol{q}$ are usually measured, such that the angular velocities $\dot{\boldsymbol{q}}$ and accelerations $\ddot{\boldsymbol{q}}$ can be obtained through numerical differentiation. It is important to note that we do not require force torque sensors to determine interaction forces, which often suffer from high measurement noise and are expensive.

The task of each patient is the execution of a rehabilitation exercise described by a reference trajectory $\boldsymbol{q}_{\mathrm{ref}}$ for system (1), for which we require the following property.

Assumption 3: The bounded reference trajectory $\boldsymbol{q}_{\mathrm{ref}}$ is twice continuously differentiable with bounded derivatives.

Since abrupt movements must be avoided in physical human-robot interaction, in particular when dealing with impaired patients, the required smoothness of the reference is a natural assumption. Moreover, the assumed boundedness of reference trajectories directly follows from the compact work spaces robots and humans operate in, such that Assumption 3 is not restrictive in practice.

In order to execute the task by tracking the reference trajectory with the human-robot system (1), a control law needs to be defined to determine the control inputs $\boldsymbol{u}$. On the one hand, the applicability in real-world rehabilitation robotic scenarios requires this control law to ensure safe interaction forces by avoiding excessively high control inputs $\boldsymbol{u}$. On the other hand, the successful execution of the rehabilitation exercise requires a satisfactory tracking accuracy. These requirements generally pose conflicting goals, which must be traded-off in a control gain tuning phase. However, this tuning process cannot be performed for each patient individually, such that control gains must be employed, which are expected to perform well on a wide range of patients, but can yield rather poor performance for some of them. In order to overcome this issue, we consider the problem of designing a control law which adapts online to the observed behavior of individual patients using non-parameteric machine learning, such that a highly personalized treatment can be realized.

III. GAUSSIAN PROCESS-BASED ONLINE LEARNING

In order to develop control laws achieving the posed design goals, we employ Gaussian process-based machine learning to infer a model of the human motor behavior. The foundations of Gaussian process regression are introduced in Section III-A, before a Gaussian process based online learning algorithm relying on the aggregation of local Gaussian process models is presented in Section III-B.

A. Gaussian Process Regression

Gaussian process (GP) regression bases on the assumption that any finite number of evaluations $\{f(\boldsymbol{x}^{(n)})\}_{n=1}^N$ of a scalar function $f : \mathbb{R}^\rho \to \mathbb{R}$, $\rho \in \mathbb{N}$, at inputs $\boldsymbol{x}^{(n)} \in \mathbb{R}^\rho$ follows a joint Gaussian distribution [13]. This distribution, denoted as $\mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}'))$, is defined in terms of a prior mean function $m : \mathbb{R}^\rho \to \mathbb{R}$, which can be used to incorporate prior knowledge such as parametric models, and a covariance function $k : \mathbb{R}^\rho \times \mathbb{R}^\rho \to \mathbb{R}$, capturing more abstract information such as differentiability or periodicity of $f(\cdot)$. If no specific knowledge about the unknown function $f(\cdot)$ is available, $m(\cdot)$ is commonly set to 0, which we also assume in the remainder of the work. The most frequently used covariance function is the squared exponential kernel

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f^2 \exp\left(-\sum_{i=1}^{\rho} \frac{(x_i - x_i')^2}{2l_i^2}\right), \qquad (2)$$

whose shape depends on the signal standard deviation $\sigma_f \in \mathbb{R}_+$ and the length scales $l_i \in \mathbb{R}_+$. The signal standard deviation σ_f and the kernel length scales l_i, $i = 1, \ldots, \rho$, form the hyperparameters $\boldsymbol{\theta} = \begin{bmatrix}\sigma_f & l_1 & \ldots & l_\rho & \sigma_{\mathrm{on}}\end{bmatrix}^T$ together with an assumed target noise standard deviation $\sigma_{\mathrm{on}} \in \mathbb{R}_+$. The hyperparameters $\boldsymbol{\theta}$ are commonly obtained by maximizing the log-likelihood

$$\begin{aligned}\log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) = & -\frac{1}{2}\boldsymbol{y}^T(\boldsymbol{K} + \sigma_{\mathrm{on}}^2 \boldsymbol{I}_N)^{-1}\boldsymbol{y} \\ & -\frac{1}{2}\log(\det(\mathbf{K} + \sigma_{\mathrm{on}}^2 \boldsymbol{I}_N)) - \frac{n}{2}\log(2\pi), \quad (3)\end{aligned}$$

where we define the elements of the kernel matrix $\boldsymbol{K} \in \mathbb{R}^{N\times N}$ as $K_{i,j} = k(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)})$, and concatenate the training inputs and targets into $\boldsymbol{X} = \begin{bmatrix}\boldsymbol{x}^{(1)} & \ldots & \boldsymbol{x}^{(N)}\end{bmatrix}$ and $\boldsymbol{y} =$

$\begin{bmatrix} y^{(1)} & \dots & y^{(N)} \end{bmatrix}^T$, respectively. Although this maximization involves a non-convex optimization problem, it is typically solved using gradient based optimization methods [13].

After the hyperparameters have been optimized, the posterior distribution can be exactly calculated under the assumption of training targets perturbed by zero mean Gaussian noise with variance σ_{on}^2. This posterior is again Gaussian with mean and variance

$$\mu(\boldsymbol{x}) = \boldsymbol{y}^T \left(\boldsymbol{K} + \sigma_{\mathrm{on}}^2 \boldsymbol{I}_N\right)^{-1} \boldsymbol{k}(\boldsymbol{x}) \tag{4}$$

$$\sigma^2(\boldsymbol{x}) = k(\boldsymbol{x}, \boldsymbol{x}) - \boldsymbol{k}^T(\boldsymbol{x}) \left(\boldsymbol{K} + \sigma_{\mathrm{on}}^2 \boldsymbol{I}_N\right)^{-1} \boldsymbol{k}(\boldsymbol{x}), \tag{5}$$

where we define the elements of the kernel vector $\boldsymbol{k}(\boldsymbol{x}) \in \mathbb{R}^N$ as $k_i(\boldsymbol{x}) = k(\boldsymbol{x}^{(i)}, \boldsymbol{x})$.

B. Locally Growing Random Trees of Gaussian Processes

When applying Gaussian process regression in a control application, data becomes available sequentially as time proceeds. While Gaussian process regression can be straightforwardly applied to such data streams using rank one updates in principle [14], the computational complexity of this approach scales quadratically with the number of training samples, such that exact inference typically becomes too slow for online learning in control loops.

A common approach in robotics applications to overcome this issue of GPs bases on the divide and conquer principle of splitting up the data set and training multiple local GP models [15]. Locally growing random trees of Gaussian processes (Log-GPs) are a recently proposed method following this idea, which have been demonstrated to achieve update and prediction rates necessary for online learning within control loops, while preserving many beneficial properties of exact GP inference [16]. This method constructs a tree, whose leaf nodes contain locally active Gaussian process models with a maximum number $\bar{N} \in \mathbb{N}$ of training samples. Given a tree consisting only of the root node and no training samples, a binary tree is iteratively constructed with incoming training data $(\boldsymbol{x}, y)$ using the following procedure:

1) Given a new training pair $(\boldsymbol{x}, y)$, the tree is traversed until a leaf is reached by going to the child of node n, which is sampled from a Bernoulli distribution with probability $p_n(\boldsymbol{x})$.
2) If the leaf contains fewer than $\bar{N}$ training pairs continue with 5), else go to 3)
3) add two child nodes to the current leaf node n and distribute its training data by sampling from $p_n(\boldsymbol{x}^{(i)})$ for all $i = 1, \dots, \bar{N}$.
4) Go to the child node sampled from $p_n(\boldsymbol{x})$.
5) Add the training pair $(\boldsymbol{x}, y)$ to the local GP model in the current leaf node and update its hyperparameters.

In this procedure, the Bernoulli distributions with probability functions $p_n(\cdot)$ define the regions where local models are active. In order to quickly determine the probabilities $p_n(\boldsymbol{x})$, saturating linear functions of the form

$$p_n(\boldsymbol{x}) = \begin{cases} 0 & \text{if } x_{j_n} < s_n - \frac{o_n}{2} \\ \frac{x_{j_n} - s_n}{o_n} + \frac{1}{2} & \text{if } s_n - \frac{o_n}{2} \leq x_{j_n} \leq s_n + \frac{o_n}{2} \\ 1 & \text{if } s_n + \frac{o_n}{2} < x_{j_n} \end{cases} \tag{6}$$

have been proposed [16], where j_n denotes the dimension in which the state space is split, s_n denotes the value of the splitting plane, and o_n corresponds to the size of the region where both local GPs are active to ensure a smooth global model. The splitting dimension j_n can be chosen, e.g., as the dimension with the maximum spread in the training data, while a simple choice for the position of the splitting plane s_n is the mean of the data in dimension j_n. Finally, the size of the overlapping region o_n is typically chosen to be fixed ratio of the extension the local models.

In order to retain a low computational complexity in the hyperparameter optimization in step 5), we merely perform a single gradient-based optimization step of the log-likelihood, whose partial derivatives are given by

$$\frac{\partial}{\partial \theta_i} \log p(\boldsymbol{y} | \boldsymbol{X}, \boldsymbol{\theta}) = \frac{1}{2} \left(\boldsymbol{y}^T \tilde{\boldsymbol{K}} \frac{\partial \boldsymbol{K}}{\partial \theta_i} \tilde{\boldsymbol{K}} \boldsymbol{y} - \mathrm{tr} \left(\tilde{\boldsymbol{K}} \frac{\partial \boldsymbol{K}}{\partial \theta_i} \right) \right). \tag{7}$$

Hence, hyperparameters are adapted online in step 5) using update rules of the form

$$\tilde{\boldsymbol{\theta}}^{n+1} = \tilde{\boldsymbol{\theta}}^n + \boldsymbol{\psi} \left(\nabla_{\tilde{\boldsymbol{\theta}}} \log p(\boldsymbol{y} | \boldsymbol{X}, \boldsymbol{\phi}(\tilde{\boldsymbol{\theta}})) \right) \Delta_t \tag{8}$$

where the step width $\Delta_t \in \mathbb{R}_+$ and the direction function $\boldsymbol{\psi} : \mathbb{R}^\rho \to \mathbb{R}^\rho$ can be defined to realize commonly used optimization schemes, e.g., steepest ascent or conjugate gradient [17]. Since the computational complexity of the update step (8) only depends on the number of training samples in a local model but is independent of the overall amount of data, the update complexity of LoG-GPs remains logarithmic with this online hyperparameter adaptation scheme.

Based on the constructed binary tree, the predictions of local GP models can be efficiently aggregated. For this purpose, given a test point $\boldsymbol{x}$, the probabilities $p_n(\boldsymbol{x})$ along a branch of the tree are multiplied to obtain the weight $w_m(\boldsymbol{x})$ of leaf node m. Then, standard aggregation schemes such as mixtures of experts [18]

$$\tilde{\mu}(\boldsymbol{x}) = \sum_{m=1}^{M} w_m(\boldsymbol{x}) \mu_m(\boldsymbol{x}) \tag{9}$$

where M denotes the number of leaf nodes, can be employed to calculate the aggregated mean $\tilde{\mu}(\boldsymbol{x})$. Since the probabilities $p_n(\cdot)$ are usually chosen such that only few local models are active at the same test point, the tree structure can be again exploited for computing the weights $w_m(\boldsymbol{x})$, yielding a $\mathcal{O}(\log^2(N))$ complexity for the computation of predictions under weak assumptions [16]. Thereby, LoG-GPs can achieve the high prediction and update rates as required for learning in many control loops.

IV. INDIVIDUALIZED CONTROL USING GAUSSIAN PROCESS BASED ONLINE LEARNING

In order to allow for an individualized treatment of patients, it is necessary to infer personalized models. Even though these models could be learned offline using data

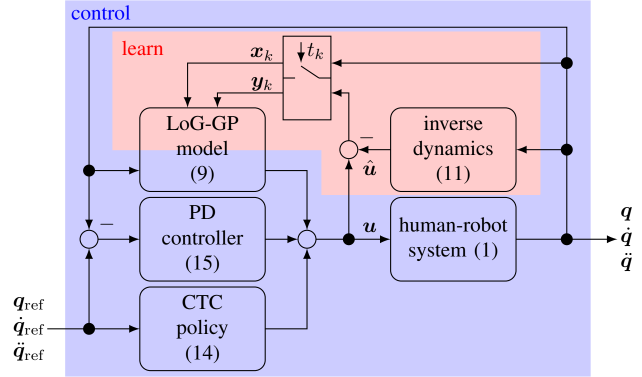

Fig. 1. Online learning control architecture with control components highlighted in blue and learning blocks illustrated in red. Noisy measurements of the system outputs and torque errors $\hat{\boldsymbol{u}} - \boldsymbol{u}$ are taken at times t_k and used to infer a model online using LoG-GPs. Predictions of the LoG-GPs are employed as feedforward term in a CTC control law.

obtained during calibration phases conducted before the actual rehabilitation exercises, this is often perceived as cumbersome by patients. Therefore, we propose an online learning control law based on LoG-GPs which achieves an adaptation to individual patients in virtually real-time.

The proposed control architecture consisting of the online data generation and model inference as well as the feedforward and feedback control components is outlined in Fig. 1. Detailed information about the learning procedure can be found in Section IV-A, while explanations regarding the individualized control law are provided in Section IV-B.

A. Online Data Generation and Model Inference

In order to account for the individual human-induced dynamics in the controller, it is necessary to infer a model online from data. In an idealized scenario, direct measurements of the torques generated by humans would be measured, i.e., $\boldsymbol{y} = \boldsymbol{f}_i(\boldsymbol{x})$, such that a model of the patient's dynamics can be learned from the training samples $\boldsymbol{x}$, $\boldsymbol{y}$. However, due to the considered lack of force torque sensors for measuring the interaction forces as stated in Assumption 2, we cannot directly employ measurements of $\boldsymbol{f}_i(\cdot)$ for training GP models. This problem can straightforwardly be overcome by rearranging (1), such that

$$\boldsymbol{f}_i(\boldsymbol{x}) = \boldsymbol{u} - \hat{\boldsymbol{u}}(\boldsymbol{x}), \tag{10}$$

where

$$\hat{\boldsymbol{u}}(\boldsymbol{x}) = \boldsymbol{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) \tag{11}$$

denotes the inverse dynamics model. Since the applied torque $\boldsymbol{u}$ is determined by the employed control law and the parameters of the Euler-Lagrange system describing the robot dynamics are known as stated in Assumption 1, $\hat{\boldsymbol{u}}(\boldsymbol{x})$ can directly be computed. Therefore, it remains to define a sampling rate $1/\tau$, $\tau \in \mathbb{R}_+$ at which measurements of $\boldsymbol{x}$ and $\boldsymbol{u}$ are taken, such that a training data set

$$\{(\boldsymbol{x}^{(k)} = \boldsymbol{x}(k\tau), \boldsymbol{y}^{(k)} = \boldsymbol{u}(k\tau) - \hat{\boldsymbol{u}}(\boldsymbol{x}(k\tau)))\}_{k=0}^{K} \tag{12}$$

for $K = \lfloor t/\tau \rfloor$ is aggregated based on the online measurements. Using these online data, we can update an independent GP for each target dimension of $\boldsymbol{y}^{(k)}$, i.e., for each $i = 1, \ldots, d$ a LoG-GP is updated using a training pair $(\boldsymbol{x}^{(k)}, y_i^{(k)})$ as explained in Section III-B. In order to employ these LoG-GPs in the control loop, their predictions are concatenated into the vector $\tilde{\boldsymbol{\mu}}(\boldsymbol{x}) = [\mu_1(\boldsymbol{x}) \ \cdots \ \mu_d(\boldsymbol{x})]^T$.

Remark 2: The proposed approach for learning a model of the individual patient dynamics in physical human-robot interaction online is independent of a particular control law. Therefore, the choice of a particular control law does not affect the model inference approach.

B. Individualized Control Law

By exploiting the online learned model in the form of a feedforward control, it is straightforward to achieve a flexible adaptation to the individual dynamics of each patient without the need for additional calibration phases. Therefore, we propose the individualized control law

$$\boldsymbol{u} = \boldsymbol{u}_{\mathrm{CTC}}(\boldsymbol{p}) + \boldsymbol{u}_{PD}(\boldsymbol{e}, \dot{\boldsymbol{e}}) + \tilde{\boldsymbol{\mu}}(\boldsymbol{x}), \tag{13}$$

where the computed torque control

$$\boldsymbol{u}_{\mathrm{CTC}}(\boldsymbol{p}, \boldsymbol{p}_{\mathrm{ref}}) = \boldsymbol{H}(\boldsymbol{q})\ddot{\boldsymbol{q}}_{\mathrm{ref}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}_{\mathrm{ref}} + \boldsymbol{g}(\boldsymbol{q}) \tag{14}$$

is used to compensate the nonlinear dynamics of the robotic system, and the PD controller

$$\boldsymbol{u}_{PD}(\boldsymbol{e}, \dot{\boldsymbol{e}}) = -\boldsymbol{K}_p\boldsymbol{e} - \boldsymbol{K}_d\dot{\boldsymbol{e}} \tag{15}$$

with control gains $\boldsymbol{K}_p, \boldsymbol{K}_d \in \mathbb{R}^{d\times d}$ ensures the convergence of the tracking error $\boldsymbol{e} = \boldsymbol{q} - \boldsymbol{q}_{\mathrm{ref}}$ to a neighborhood of $\boldsymbol{0}$.

The PD gains allow a flexible trade-off between compliant behavior and high tracking accuracy, with low gains $\boldsymbol{K}_p$, $\boldsymbol{K}_d$ leading to small control inputs of the PD controller (15) in practice. Since the computed torque control (14) is practically bounded under Assumption 3 and the magnitude of LoG-GP predictions is restricted through the observed torques [19], which are applied by the patient, it is straightforward to see that the individualized control law (13) with a sufficiently compliant PD controller can ensure safe interaction forces. This dependency of safety on the control gains is demonstrated experimentally in Section V-B.

Remark 3: Since the individualized control law (13) is based on a classical computed torque approach and uniform error bounds for predictions with LoG-GPs can be shown under weak assumptions [16], it is straightforward to analyze the stability of the closed-loop human-robot system using Lyapunov theory analogously to, e.g., [20], [21]. For reasons of brevity, a formal stability analysis is omitted here.

V. EXPERIMENTAL EVALUATION

For the evaluation of the proposed learning-based controller, we perform experiments with a two DoF human-robot interaction setup, which is explained in Section V-A. First, the capacity of the method to successfully assist during a rehabilitation exercise whilst applying safe control outputs is shown in Section V-B. After this initial validation, the method is contrasted with a controller tuned for one individual, in Section V-C. Thereby, we demonstrate the learners ability to adapt to different human operators and provide personalized assistance.

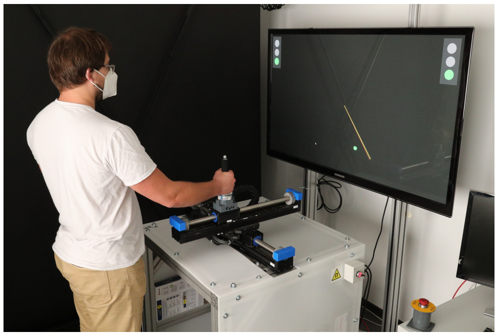

Fig. 2. Reenactment of an individual performing the experiment task with the manipulandum. Consent for the publication of the image was obtained.

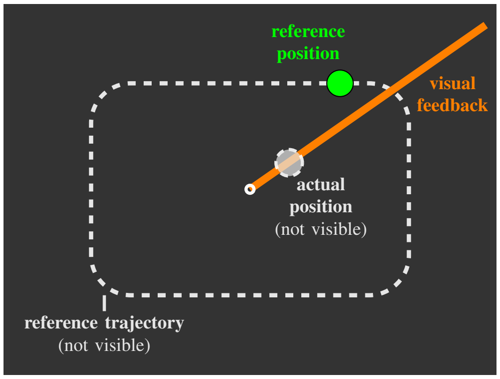

Fig. 3. Exemplary depiction of the task design. The rounded rectangle drawn with the gray, dashed line represents the reference trajectory of the green circle, which the participant is instructed to track. Instead of the actual current position, depicted by the gray circle, the subject can only

A. *Experiment Setup and Task Design*

The experiments are executed on a two DoF manipulandum, which consists of two orthogonally mounted single rail stages (Copley Controls Thrustube Module), each driven by linear servo motors. Both rail stages are equipped with optical encoders that measure the position of a cart on the upper rail with $1\,\mu\mathrm{m}$ precision. Additionally, a six DoF force-torque sensor (JR3-75M25) is mounted below the handle, through which the human interacts with the system, to measure forces in the horizontal plan. The force-torque sensor is strictly required to operate the device and is not used for the computation of control policies. Additionally, an inherent output force limitation is integrated as a safety measure, which guarantees that the interaction force applied to the human remains in safe regions, therefore, enabling experiments with aggressively tuned controllers. The device runs at $4\,\mathrm{kHz}$ and the workspace of the complete apparatus spans $\pm 0.20\,\mathrm{m}$ in both DoF. Visual feedback to perform the task is provided to the user through a screen placed behind the apparatus. The complete experiment setup and apparatus is shown in Fig. 2. The task itself is designed as follows:

Standing in front of the apparatus and facing the screen, the subjects are instructed to track a green dot by moving the handle on top of the cart. In addition the participants are informed that different controllers will support them during task execution. Since the experiments are performed by healthy subjects, the provided visual feedback is artificially modified, therefore, limiting the participants' ability to successfully perform the task and thereby mimicking the physical limitations of an impaired patient. Specifically, the subjects do not see their current position in the task space entirely, but instead only the angle from the origin is visualized through a pointer. However, despite the limited feedback, their tracking performance is still evaluated on the position error in Cartesian coordinates. The task design and the visual feedback is depicted in Fig. 3. Each run of the experiment begins at the same starting position for the green circle and consists of five repetitions of the reference trajectory. The complete experimental procedure can be split into two parts; first a training phase, followed by a test phase. During the initial training phase the participants get accustomed to the task and the assistance by performing one experiment run with each controller. Subsequently, the test phase begins, which consists of four experiment runs per controller. At every run a random controller variation is selected for assistance. If during any trial the workspace limit is reached, the device shuts down as a safety precaution and the run is evaluated as a failure. The failed runs are not repeated subsequently.

TABLE I
PD CONTROL GAINS USED IN THE EXPERIMENTS.

	LOW GAIN	HIGH-GAIN	GP	TUNED GAINS
k_p	1	600	1	35
k_d	0.1	60	0.1	3.5

B. *Compliance and Accuracy with Learning Control*

In order to demonstrate the applicability of our approach, we conduct a user study with 9 healthy, right-handed participants between the age of 22 and 35. Participants signed a written informed consent, approved by the ethics committee of the medical faculty of the Technical University of Munich. During the experiment the operators perform a task whilst being assisted by three different variations of the proposed control architecture. A high-gain, low-gain and GP variation with PD control-parametrization according to Table I and diagonal gain matrices $\boldsymbol{K}_p = k_p \boldsymbol{I}$, $\boldsymbol{K}_d = k_d \boldsymbol{I}$, $k_p, k_d \in \mathbb{R}_+$. For the high-gain and low-gain controller there is no learning GP and they only differ with regards to the gains used for the PD controller (15). Differently, the GP controller uses the individualized control law (13) including the proposed learning-based controller with small gains for the PD controller. The online hyperparameter adaptation (8) is realized using RPROP [22] since it has been demonstrated to exhibit lower computational complexity and faster convergence compared to other gradient-based optimization schemes [23]. All three controllers have the same CTC policy (14) and the GP runs at a update and prediction rate of $200\,\mathrm{Hz}$, resulting in approximately 10000 training samples at the end of one experimental run.

The analysis of the experimental results are shown in Fig. 4. The top bar plots depict the tracking performance by comparing the mean and standard deviation of the summed absolute error for each controller. It is clearly visible that the low-gain controller leads to the worst tracking accuracy and exhibits a notable variation between subjects, which is appar-

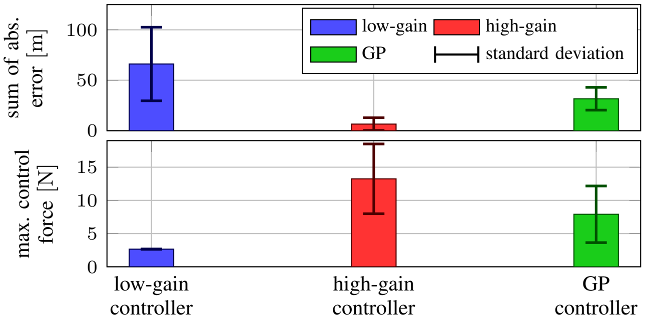

Fig. 4. Top: Mean and standard deviation of the summed absolute error for all participants over low-gain, high-gain and GP controller configuration. Bottom: Mean and standard deviation of the maximum applied control force norm at each run. The GP controller trades-off applied maximum control force and tracking performance, while the low-gain and high-gain controller either exhibit low tracking performance or overly large control forces.

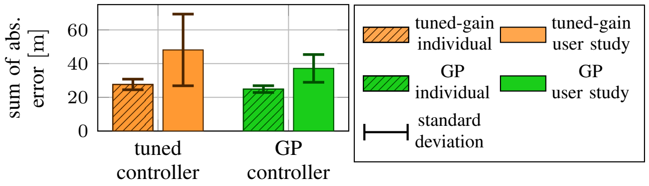

Fig. 5. Mean and standard deviation of the summed absolute error for one surrogate participant and the complete user study. While the tuned PD controller and GP controller perform comparably for the surrogate individual, only the GP controller adapts in the user study to personalize the assistance required for acceptable tracking performance.

ent due to the large standard deviation. In fact, four subjects reached the workspace limit at least once when they were assisted by the low-gain controller, resulting in a total of eight failed runs, which demonstrates the issues of highly compliant controllers in ensuring a successful task execution. Since the failed runs are particularly short they are not included in the analysis depicted in Fig. 4. While the high gain controller does not suffer from failed trials and exhibits the best control performance, it can result in uncomfortable interaction forces, which may ultimately become unsafe. This can be observed at the bottom of Fig. 4 depicting the applied maximum forces, which are largest for the high gain configuration. In contrast to these PD control laws, the proposed online learning controller adapts itself to each participant. Since the labels used during training of the GP correspond to the human-generated torques, it is highly unlikely that higher control forces are generated than the human operator can manage. This is also demonstrated by the experiments as shown in Fig. 4. The applied maximum control forces of the GP controller remain significantly smaller compared to the high-gain configuration, while the tracking performance is strongly improved over the low-gain controller. Therefore, it is demonstrated that the proposed online learning control scheme is capable of successful task execution while being safer than high-gain controllers due to smaller interaction forces.

C. Personalized Assistance through Online Learning

In order to demonstrate the benefits of the proposed learning controller, we compare it to a PD controller with tuned gains. Due to the lack of a better procedure for adaptation in human-robot interaction, the gains of the PD controller are tuned heuristically to balance the applied forces and the resulting tracking performance. As practical considerations prevent a tuning with all participants, one individual is chosen instead. However, since the PD controller needs to be safe for all users, a cautious tuning is preferred, which tends to result in lower control gains. The best trade-off is obtained for the parametrization depicted on the right side of Table I.

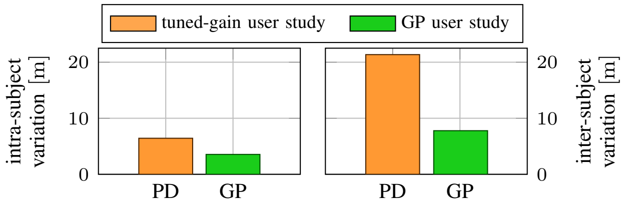

Fig. 6. Left: Average of intra-subject standard deviation in tracking error. Right: Inter-subject standard deviation of participant specific average tracking error. The adaptability of the GP controller is reflected in the smaller intra- and inter-subject variation.

The tuned PD controller and the proposed online learning control law are evaluated as described in Section V-B, which leads to the results depicted in Fig. 5. For the individual, the two controllers perform comparably well with regards to tracking performance, with the GP controller leading to slightly better tracking. However, the observed difference in tracking error is insignificant, since it lies within the statistical variation, and can be attributed to the cautious tuning of the PD controller. When deploying the tuned PD controller to previously unobserved individuals and comparing the performance to the learning-based GP controller in a user study, it can be seen that the tuned PD controller performs significantly worse. Similarly, the GP controller on average results in higher tracking errors in the user study than for the surrogate individual. However, the increase in mean tracking error is larger for the tuned PD controller with a substantial growth in the standard deviation. Therefore, the tuned PD controller leads to inconsistent tracking results, which can be attributed to the different levels of task proficiency of the participants. This becomes even clearer when looking at the intra- and inter-subject variation of the tracking error as depicted in Fig. 6. While the participants exhibit a similar variation of the tracking error among the experiment runs for both controllers, the variation of the tracking error between different subjects is significantly larger for the tuned PD controller. This is due to the inability of the PD control law to adapt to unproficient participants, which require more guidance to execute the task properly. In contrast, the GP controller provides personalized support for each subject, such that the variation between different participants can be reduced.

VI. Conclusion

This paper introduces a novel online learning control method for personalized rehabilitation robotics. This is the first time that GPs are used directly in the control loop during physical human-robot collaboration. Furthermore, the proposed method facilitates personalized assistance via online learning with previously unseen update rates. The approach is validated in an experimental user study, which confirms the adaptation capabilities of the learning-based controller.

VII. ACKNOWLEDGMENTS

This work has received funding from the Horizon 2020 research and innovation programme of the European Union under grant agreement n° 871767 of the project ReHyb: Rehabilitation based on hybrid neuroprosthesis. A. L. gratefully acknowledges financial support from the German Academic Scholarship Foundation.

REFERENCES

[1] V. L. Feigin, M. H. Forouzanfar, R. Krishnamurthi, G. A. Mensah, M. Connor, D. A. Bennett, A. E. Moran, R. L. Sacco, L. Anderson, T. Truelsen, M. O'Donnell, N. Venketasubramanian, S. Barker-Collo, C. M. Lawes, W. Wang, Y. Shinohara, E. Witt, M. Ezzati, and M. Naghavi, "Global and regional burden of stroke during 1990-2010: Findings from the Global Burden of Disease Study 2010," *The Lancet*, vol. 383, no. 9913, pp. 245–255, 2014. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/24449944/

[2] P. A. Ringleb, M. G. Bousser, G. Ford, P. Bath, M. Brainin, V. Caso, Á. Cervera, A. Chamorro, C. Cordonnier, L. Csiba, A. Davalos, H. C. Diener, J. Ferro, W. Hacke, M. Hennerici, M. Kaste, P. Langhorne, K. Lees, D. Leys, J. Lodder, H. S. Markus, J. L. Mas, H. P. Mattle, K. Muir, B. Norrving, V. Obach, S. Paolucci, E. B. Ringelstein, P. D. Schellinger, J. Sivenius, V. Skvortsova, K. S. Sunnerhagen, L. Thomassen, D. Toni, R. Von Kummer, N. G. Wahlgren, M. F. Walker, and J. Wardlaw, "Guidelines for management of ischaemic stroke and transient ischaemic attack 2008," pp. 457–507, 2008. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/18477843/

[3] H. I. Krebs, J. J. Palazzolo, L. Dipietro, M. Ferraro, J. Krol, K. Rannekleiv, B. T. Volpe, and N. Hogan, "Rehabilitation robotics: Performance-based progressive robot-assisted therapy," *Autonomous Robots*, vol. 15, no. 1, pp. 7–20, 2003.

[4] Y. Yang, L. Wang, J. Tong, and L. Zhang, "Arm rehabilitation robot impedance control and experimentation," in *2006 IEEE International Conference on Robotics and Biomimetics, ROBIO 2006*, 2006, pp. 914–918.

[5] J. Zhang and C. C. Cheah, "Passivity and Stability of Human-Robot Interaction Control for Upper-Limb Rehabilitation Robots," *IEEE Transactions on Robotics*, vol. 31, no. 2, pp. 233–245, 2015.

[6] X. Li, Y. H. Liu, and H. Yu, "Iterative learning impedance control for rehabilitation robots driven by series elastic actuators," *Automatica*, vol. 90, no. April, pp. 1–7, 2018.

[7] F. Just, O. Özen, P. Bösch, H. Bobrovsky, V. Klamroth-Marganska, R. Riener, and G. Rauter, "Exoskeleton transparency: Feed-forward compensation vs. disturbance observer," *at - Automatisierungstechnik*, vol. 66, pp. 1014–1026, 11 2018.

[8] C. An and J. Hollerbach, "Dynamic stability issues in force control of manipulators," in *Proceedings. 1987 IEEE International Conference on Robotics and Automation*, vol. 4, 1987, pp. 890–896.

[9] W. He, S. S. Ge, Y. Li, E. Chew, and Y. S. Ng, "Neural Network Control of a Rehabilitation Robot by State and Output Feedback," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 80, no. 1, pp. 15–31, 2015.

[10] J. R. Medina, H. Borner, S. Endo, and S. Hirche, "Impedance-Based Gaussian Processes for Modeling Human Motor Behavior in Physical and Non-Physical Interaction," *IEEE Transactions on Biomedical Engineeringngineering*, vol. 66, no. 9, pp. 2499–2511, 2019.

[11] P. Beckerle, G. Salvietti, R. Unal, D. Prattichizzo, S. Rossi, C. Castellini, S. Hirche, S. Endo, H. B. Amor, M. Ciocarlie, F. Mastrogiovanni, B. D. Argall, and M. Bianchi, "A human-robot interaction perspective on assistive and rehabilitation robotics," *Frontiers in Neurorobotics*, vol. 11, no. MAY, pp. 1–6, 2017.

[12] J. Hollerbach, W. Khalil, and M. Gautier, "Model Identification," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Springer Berlin Heidelberg, 2008, ch. 14, pp. 321–344.

[13] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA: The MIT Press, 2006.

[14] D. Nguyen-Tuong and J. Peters, "Incremental sparsification for real-time online model learning," *Journal of Machine Learning Research*, vol. 9, pp. 557–564, 2010.

[15] 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.

[16] A. Lederer, A. Odonez Conejo, K. Maier, W. Xiao, J. Umlauft, and S. Hirche, "Gaussian Process-Based Real-Time Learning for Safety Critical Applications," in *International Conference on Machine Learning*, 2021, pp. 6055–6064.

[17] P. Pedregal, *Introduction to Optimization*. New York, NY: Springer Science+Business Media, 2004.

[18] V. Tresp, "Mixtures of Gaussian processes," *Advances in Neural Information Processing Systems*, 2001.

[19] A. Capone, A. Lederer, and S. Hirche, "Gaussian Process Uniform Error Bounds with Unknown Hyperparameters for Safety-Critical Applications," 2021. [Online]. Available: http://arxiv.org/abs/2109.02606

[20] M. K. Helwa, A. Heins, and A. P. Schoellig, "Provably Robust Learning-Based Approach for High-Accuracy Tracking Control of Lagrangian Systems," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1587–1594, 2019.

[21] J. Umlauft and S. Hirche, "Feedback Linearization based on Gaussian Processes with event-triggered Online Learning," *IEEE Transactions on Automatic Control*, vol. 65, no. 10, pp. 4154–4169, 2019.

[22] M. Riedmiller and H. Braun, "A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm," in *IEEE International Conference on Neural Networks*, 1993, pp. 586–591.

[23] M. Blum and M. Riedmiller, "Optimization of gaussian process hyperparameters using Rprop," in *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2013, pp. 339–344.

Learning Stable Nonparametric Dynamical Systems with Gaussian Process Regression

Wenxin Xiao*, Armin Lederer[1]* and Sandra Hirche[1]

Abstract— **Modelling real world systems involving humans such as biological processes for disease treatment or human behavior for robotic rehabilitation is a challenging problem because labeled training data is sparse and expensive, while high prediction accuracy is required from models of these dynamical systems. Due to the high nonlinearity of problems in this area, data-driven approaches gain increasing attention for identifying nonparametric models. In order to increase the prediction performance of these models, abstract prior knowledge such as stability should be included in the learning approach. One of the key challenges is to ensure sufficient flexibility of the models, which is typically limited by the usage of parametric Lyapunov functions to guarantee stability. Therefore, we derive an approach to learn a nonparametric Lyapunov function based on Gaussian process regression from data. Furthermore, we learn a nonparametric Gaussian process state space model from the data and show that it is capable of reproducing observed data exactly. We prove that stabilization of the nominal model based on the nonparametric control Lyapunov function does not modify the behavior of the nominal model at training samples. The flexibility and efficiency of our approach is demonstrated on the benchmark problem of learning handwriting motions from a real world dataset, where our approach achieves almost exact reproduction of the training data.**

I. INTRODUCTION

Identification of models for systems involving humans is a highly relevant problem in many fields such as medicine, where dynamical systems can be used to model the progression of a disease, and robotic rehabilitation, where models of the human behavior can be used to maximize the training efficiency. Major difficulties in these modelling problems typically are a high nonlinearity of real world systems, the absence of first principle models and sparsity of the expensive data [1]. Therefore, parametric models are generally not capable of representing these complex system appropriately.

As a more flexible solution, data-driven approaches, which can extract necessary information automatically from training data, have gained increasing attention for modeling nonlinear systems, since they exhibit sufficient flexibility to adapt their complexity to the observed data and only require marginal prior knowledge. Although classical system identification literature has considered the problem of determining stable models, see, e.g, [2], the combination of machine learning techniques and control theory has led to a variety of new approaches recently. A common method is to adapt standard machine learning approaches using Lyapunov stability constraints during model parameter optimization. Using a quadratic Lyapuonv function, this method has been applied to Gaussian mixture models in the stable estimator of dynamical systems approach proposed by [3], and is further improved in [4] by employing additional prior distributions, which ensure physical consistency. The constrained optimization method has also been used in combination with neural networks [5], where the flexibility of the model can be improved by learning the Lyapunov function with a separate neural network [6]. Since this constrained training approach can have negative effects on the learning performance, it has been proposed to learn a possibly unstable nominal model and a control Lyapunov function (CLF) separately, such that a virtual control can be determined based on the CLF to stabilize the nominal model [7]. This approach has been pursued with different Lyapunov functions, such as the weighted sum of asymmetric quadratic functions [7] and sums of squares [8]. Furthermore it has been extended to achieve risk-sensitive behavior by considering the model uncertainty due to sparsity of data [9].

Although the existing methods ensure stable trajectories and achieve low reproduction errors on many practical examples, there are no guarantees on the achievable expressiveness using a certain model. As this issue arises mainly due to the use of parametric Lyapunov functions, we develop a novel, nonparametric Lyapunov function which can be learned from data using Gaussian process regression. We employ a Gaussian process state space model (GP-SSM) as nominal model, and show that it can learn dynamical systems accurately on training data. By stabilizing the GP-SSM based on the nonparametric control Lyapunov function, we prove that the resulting model is capable of reproducing observed data exactly, while being globally asymptotically stable. The flexibility of the approach is demonstrated in learning dynamical systems from a real world dataset, and compared to existing methods.

The remainder of this paper is organized as follows. In Section II, we describe the considered problem. Section III explains our approach to learn a stable dynamical system which reproduces observed data. The method is compared to existing approaches on real world data in Section IV.

II. PROBLEM STATEMENT

Consider a nonlinear, discrete-time dynamical system[1]

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k) \tag{1}$$

*These authors contributed equally.

[1] Armin Lederer and Sandra Hirche are with the Chair of Information-oriented Control (ITR), Department of Electrical and Computer Engineering, Technical University of Munich, Germany {armin.lederer,hirche}@tum.de

[1] **Notation:** Lower/upper case bold symbols denote vectors/matrices, respectively, $\boldsymbol{I}_n$ the $n \times n$ identity matrix, $\mathbb{R}_+$ all positive real numbers, $\|\cdot\|$ the Euclidean norm and $E[\cdot]$ the expectation operator.

which is asymptotically stable on the continuous valued state space $\mathbb{X} \subset \mathbb{R}^d$. Furthermore, assume that the function $\boldsymbol{f}(\cdot)$ is unknown and that consecutive measurements of the states are taken such that we obtain a training data set $\mathbb{D} = \{(\boldsymbol{x}_k^{(m)}, \boldsymbol{x}_{k+1}^{(m)})\}_{m=1}^M$ with $M \in \mathbb{N}$ data pairs. We will make use of the following assumption.

Assumption 1: The function $\boldsymbol{f}(\cdot)$ defines an asymptotically stable system (1) on the compact set $\mathbb{X} \subset \mathbb{R}^d$.

We want to estimate a model based on the observed data, which exhibits the posed assumptions on stability. Therefore, the goal is to derive a stable model of the unknown dynamical system, which maximizes the accuracy of the reproduced training trajectories by reproducing the observed data $(\boldsymbol{x}_k^{(m)}, \boldsymbol{x}_{k+1}^{(m)})$ exactly.

III. Stabilization of Gaussian Process State Space Models

For learning stable dynamical systems capable of reproducing observations, we follow the control Lyapunov function approach proposed in [8]. For this virtual stabilization method, we separately learn a nominal system model $\boldsymbol{\mu} : \mathbb{X} \to \mathbb{R}^d$ and a control Lyapunov function $V : \mathbb{X} \to \mathbb{R}_+$ from the training data. For a prediction, we determine the optimal, stabilizing virtual control $\boldsymbol{u}^*(\boldsymbol{x})$ for the nominal model $\boldsymbol{\mu}(\cdot)$ based on the control Lyapunov function $V(\cdot)$, which minimally modifies the nominal model, and define the stable model as

$$\hat{\boldsymbol{f}}(\boldsymbol{x}) = \boldsymbol{\mu}(\boldsymbol{x}) + \boldsymbol{u}^*(\boldsymbol{x}). \tag{2}$$

Since we consider scenarios with sparse data, we employ Gaussian process (GP) regression, whose implicit bias-variance trade-off avoids overfitting and hence, provides high prediction accuracy with few training samples. We consider deterministic systems and therefore, we use noise-free Gaussian process state space models as nominal model in contrast to the approach proposed in [8]. We show that the noise-free GP-SSMs are capable of reproducing the training data exactly under weak assumptions in Section III-A. In Section III-B we propose a novel method to learn a nonparametric control Lyapunov function from training data based on Gaussian process regression, which is guaranteed to converge along the training data. Finally, we show that a stabilizing control can be obtained via a constrained optimization and equals zero for all training data in Section III-C. Therefore, we obtain an asymptotically stable model (2), which is capable of reproducing observed data exactly.

A. Gaussian Process State Space Models

Gaussian processes are a powerful machine learning tool for approximating nonlinear functions [10]. A GP is a stochastic process on the continuous input domain $\mathbb{X}$ such that each finite subset $\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\} \subset \mathbb{X}$ is assigned a joint Gaussian distribution. This view is equal to a consideration as distribution over functions, which is typically expressed through

$$f(\boldsymbol{x}) \sim \mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')) \tag{3}$$

with prior mean and covariance function

$$m(\boldsymbol{x}) = E[f(\boldsymbol{x})] \tag{4}$$

$$k(\boldsymbol{x}, \boldsymbol{x}') = E[(f(\boldsymbol{x}) - m(\boldsymbol{x}))(f(\boldsymbol{x}') - m(\boldsymbol{x}'))]. \tag{5}$$

A GP is completely specified by its mean function $m(\cdot)$ and covariance kernel $k(\cdot, \cdot)$. The mean function allows to include prior knowledge in the form of approximate or parametric models. While such models exist for some applications, we do not assume their availability in the following and set the prior mean function to $m(\boldsymbol{x}) = 0$ without loss of generality. The covariance kernel $k(\cdot, \cdot)$ is used to encode more abstract prior knowledge such as information about the smoothness of the regressed function and determines which functions can be approximated properly with a Gaussian process. Probably the most commonly used kernel is the squared exponential (SE) kernel with automatic relevance determination

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f^2 \exp\left(-\frac{1}{2} \sum_{i=1}^d \left(\frac{x_i - x_i'}{l_i} \right)^2 \right), \tag{6}$$

where $\sigma_f^2 \in \mathbb{R}_+$ is the signal variance and $l_i \in \mathbb{R}_+, \forall i = 1, \dots, d$ are the length-scale parameters. These variables are concatenated in a hyperparameter vector $\boldsymbol{\psi} = [l_1 \dots l_d \; \sigma_f]^T$. The squared exponential kernel is a universal kernel in the sense of [11] which means that it allows to approximate continuous functions arbitrarily well. Therefore, Gaussian process regression with this kernel is capable of learning many typical dynamics.

We employ d independent GPs to model a dynamical system with d-dimensional state space, such that the i-th component is denoted by

$$f_i(\boldsymbol{x}) \sim \mathcal{GP}(0, k_i(\boldsymbol{x}, \boldsymbol{x}')). \tag{7}$$

Predictions with this model can be calculated by conditioning the prior GPs (7) on the given training set $\mathbb{D} = \{\boldsymbol{x}_k^{(n)}, \boldsymbol{x}_{k+1}^{(n)}\}_{i=1}^N$. The conditional expectation $\mu(\cdot)$ can be calculated analytically using linear algebra. For this reason, we define target vectors

$$\boldsymbol{y}_i = [x_{k+1,i}^{(1)} \; \dots \; x_{k+1,i}^{(N)}]. \tag{8}$$

Then, the predictive mean is given by

$$\mu_i(\boldsymbol{x}) = \boldsymbol{k}_i(\boldsymbol{x}) \boldsymbol{K}_i^{-1} \boldsymbol{y}_i \tag{9}$$

with $K_{i,n,m} = k_i(\boldsymbol{x}_k^{(n)}, \boldsymbol{x}_k^{(m)})$ and $k_{i,n} = k_i(\boldsymbol{x}, \boldsymbol{x}_k^{(n)})$.

Remark 1: The matrix inverse in (9) theoretically always exists for the squared exponential kernel if there is no repeated entry in the input data, i.e., $\boldsymbol{x}_k^{(n)} \neq \boldsymbol{x}_k^{(m)}$, $\forall n \neq m$, due to the fact that this kernel is universal [11]. However, a small regularizer, typically called observation noise variance σ_{on}^2, can be added on the diagonals of $\boldsymbol{K}_i$ in order to avoid numerically ill-conditioned inversions. This regularizer has typically a small effect on the prediction since the resulting mean squared prediction error is smaller than the noise variance [10].

The hyperparameters $\boldsymbol{\psi}_i$ of the Gaussian processes $f_i(\cdot)$ can be obtained by independently maximizing the log-likelihood

$$\log p(\boldsymbol{y}_i | \boldsymbol{X}_i) = -\frac{1}{2} \boldsymbol{y}_i^T \boldsymbol{K}_i^{-1} \boldsymbol{y}_i - \frac{1}{2} \log \det(\boldsymbol{K}_i) - \frac{N}{2} \log 2\pi, \tag{10}$$

where $\boldsymbol{X} = [\boldsymbol{x}_k^{(1)} \; \dots \; \boldsymbol{x}_k^{(N)}]$ denotes the input training data matrix. This optimization problem is typically solved using gradient based approaches [10], even though it is generally non-convex.

We use the posterior mean function $\boldsymbol{\mu}(\cdot)$ defined through (9) to define a nominal dynamical model

$$\boldsymbol{x}_{k+1} = \boldsymbol{\mu}(\boldsymbol{x}_k) \tag{11}$$

which is generally not asymptotically stable. However, training samples are reproduced exactly such that we obtain the following result.

Lemma 1: Consider a training data set $\mathbb{D} = \{(\boldsymbol{x}_k^{(m)}, \boldsymbol{x}_{k+1}^{(m)})\}_{m=1}^{M}$ generated by an unknown dynamical system (1), which has a stable equilibrium at the origin. Furthermore, assume that the training data set is augmented by adding the pair $(\boldsymbol{0}, \boldsymbol{0})$. Then, a Gaussian process state space model trained with this training data set reproduces the training data exactly and has an equilibrium at the origin.

Proof 1: Performing the prediction for all training inputs $\boldsymbol{x}_k^{(m)}$ jointly yields

$$\mu_i(\boldsymbol{X}_k) = \boldsymbol{K}_i \boldsymbol{K}_i^{-1} \boldsymbol{y}_i = \boldsymbol{y}_i,$$

where $\mu_i(\boldsymbol{X}_k) = [\mu_i(\boldsymbol{x}_k^{(1)}) \ \ldots \ \mu_i(\boldsymbol{x}_k^{(M)}) \ \mu_i(\boldsymbol{0})]$ and the inverse is well defined due to the fact that we consider a deterministic function $\boldsymbol{f}(\cdot)$ such that $\boldsymbol{x}_{k+1}^{(m)} \neq \boldsymbol{x}_{k+1}^{(m')}$, $\forall m \neq m'$. Furthermore, we have the identity

$$[\boldsymbol{y}_1 \ \ldots \ \boldsymbol{y}_d]^T = [\boldsymbol{x}_{k+1}^{(1)} \ \ldots \ \boldsymbol{x}_{k+1}^{(M)} \ \boldsymbol{0}]$$

due to the definition of the data set $\mathbb{D}$. Therefore, the training data is reproduced exactly by the nominal system (11). Finally, the equilibrium at the origin follows from the additional training pair $(\boldsymbol{0}, \boldsymbol{0})$ due to [12].

The exact reproduction of data regardless of their complexity is a major advantage of the nonparametric GP modeling approach. However, this reproduction is only possible, if the training data can be considered noise-free, which is exploited in the proof as the property $\boldsymbol{x}_{k+1}^{(m)} \neq \boldsymbol{x}_{k+1}^{(m')}$. In applications with few training data such as medical applications or human-robot interaction, this condition is typically satisfied due to the sparsity of the data. Therefore, it is not a severe restriction.

Remark 2: Since we only focus on deterministic systems in our approach, the variance of the next state $\boldsymbol{x}_{k+1}$ is not of primary interest in this paper. However, it could be used to determine regions of the state space $\mathbb{X}$, which require more training data in order to provide a good model of the dynamical system.

B. Learning Nonparametric Control Lyapunov Functions

Although exact reproduction of the data is possible using GP-SSMs, this does not imply that the stabilized system (2) also exhibits superior reproduction performance. This is due to the fact that an insufficiently flexible parameterization of the control Lyapunov function $V(\cdot)$ might not allow the decrease of $V(\cdot)$ along all training samples. However, the required flexibility is difficult to determine a priori with parametric functions such as sums of squares or weighted sum of asymmetric quadratic functions [8]. Therefore, we propose to learn a control Lyapunov function from data based on Gaussian process regression to exploit the flexibility of a fully nonparametric approach. Since we do not have any target values for the supervised learning, we cannot directly apply the GP regression approach. Therefore, we approximate the infinite horizon cost $\tilde{V}_\infty(\boldsymbol{x}) = \sum_{k=1}^{\infty} l(\boldsymbol{f}^k(\boldsymbol{x}))$, where $\boldsymbol{f}^k(\cdot)$ denotes the k-times application of the dynamics $\boldsymbol{f}(\cdot)$ and $l : \mathbb{R}^d \to \mathbb{R}_+$ is a chosen stage cost, by transforming the Bellman equation at training points into a regression problem as proposed in [13]. This is formalized in the following lemma.

Lemma 2: Consider the approximate infinite horizon cost

$$V_\infty(\boldsymbol{x}) = \boldsymbol{\lambda}^T(\boldsymbol{k}(\boldsymbol{X}_k, \boldsymbol{x}) - \boldsymbol{k}(\boldsymbol{X}_{k+1}, \boldsymbol{x})), \tag{12}$$

with positive definite stage cost $l : \mathbb{R}^d \to \mathbb{R}_+$, training points $\boldsymbol{X}_k = [\boldsymbol{x}_k^{(1)} \ \ldots \ \boldsymbol{x}_k^{(M)}]$, $\boldsymbol{X}_{k+1} = [\boldsymbol{x}_{k+1}^{(1)} \ \ldots \ \boldsymbol{x}_{k+1}^{(M)}]$ and

$$\boldsymbol{k}(\boldsymbol{X}, \boldsymbol{x}) = \left[k(\boldsymbol{x}^{(1)}, \boldsymbol{x}) \ \ldots \ k(\boldsymbol{x}^{(M)}, \boldsymbol{x})\right]^T \tag{13}$$

$$\boldsymbol{\lambda} = \boldsymbol{\kappa}^{-1}[l(\boldsymbol{x}_{k+1}^{(1)}) \ \ldots \ l(\boldsymbol{x}_{k+1}^{(M)})]^T, \tag{14}$$

where the elements of the invertible matrix $\boldsymbol{\kappa} \in \mathbb{R}^{M \times M}$ are defined using the squared exponential kernel $k(\cdot,\cdot)$ as

$$\begin{aligned} \kappa_{mn} = {} & k\left(\boldsymbol{x}_k^{(m)}, \boldsymbol{x}_k^{(n)}\right) - k\left(\boldsymbol{x}_{k+1}^{(m)}, \boldsymbol{x}_k^{(n)}\right) \\ & - k\left(\boldsymbol{x}_k^{(m)}, \boldsymbol{x}_{k+1}^{(n)}\right) + k\left(\boldsymbol{x}_{k+1}^{(m)}, \boldsymbol{x}_{k+1}^{(n)}\right). \end{aligned} \tag{15}$$

Then, the control Lyapunov function

$$V(\boldsymbol{x}) = l(\boldsymbol{x}) + \max\{0, V_\infty(\boldsymbol{x}) + V_\infty(\boldsymbol{0})\} \tag{16}$$

is positive definite and decreasing along the training data, i.e., $V(\boldsymbol{x}_k^{(m)}) \geq V(\boldsymbol{x}_{k+1}^{(m)})$, $\forall m = 1, \ldots, M$.

Proof 2: Since $l(\cdot)$ is a positive definite function, $V(\cdot)$ is positive due to its definition (16). Hence, it remains to show the decrease along the training data. For this reason, we first consider the exact infinite horizon cost function $\tilde{V}_\infty(\cdot)$, which satisfies the Bellman equation

$$\tilde{V}_\infty(\boldsymbol{x}) - \tilde{V}_\infty(\boldsymbol{f}(\boldsymbol{x})) = l(\boldsymbol{f}(\boldsymbol{x})).$$

Due to [13], a function satisfying this equation on a finite set of pairs $(\boldsymbol{x}, \boldsymbol{f}(\boldsymbol{x}))$ can be obtained through noiseless GP regression with the kernel

$$\tilde{k}(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}, \boldsymbol{x}') - k(\boldsymbol{f}(\boldsymbol{x}), \boldsymbol{x}') - k(\boldsymbol{x}, \boldsymbol{f}(\boldsymbol{x}')) + k(\boldsymbol{f}(\boldsymbol{x}), \boldsymbol{f}(\boldsymbol{x}'))$$

and output training data $\boldsymbol{y} = [l(\boldsymbol{x}_{k+1}^{(1)}) \ \ldots \ l(\boldsymbol{x}_{k+1}^{(M)})]^T$, where invertibility of the matrix $\boldsymbol{\kappa}$ defined through (15) is guaranteed due to the usage of the universal squared exponential kernel. This follows directly from a representation of $\tilde{V}_\infty(\cdot)$ in the feature space associated with the kernel $k(\cdot,\cdot)$ and the linearity of this representation. Substituting the obtained regression result into the original feature space representation of $\tilde{V}_\infty(\cdot)$ directly yields (12). Since the approximate infinite horizon cost (12) is guaranteed to satisfy the Bellman equation on the training pairs, the approximated cost $l(\boldsymbol{x}) + V_\infty(\boldsymbol{x})$ is decreasing along the training data. Since this property is shift invariant, i.e., adding a constant to (12) does not change the decrease along training data, (12) is not guaranteed to be positive for all $\boldsymbol{x} \neq \boldsymbol{0}$. Therefore, we enforce $V(\boldsymbol{0}) = 0$ by adding the value of (12) evaluated at the origin and exclude obvious regression errors by setting negative values of the shifted approximate infinite horizon $V_\infty(\boldsymbol{x}) + V_\infty(\boldsymbol{0})$ to 0. Since regression errors do not occur on the training samples, the decrease along training data is guaranteed for (16) and the theorem is proven.

The hyperparameters of the control Lyapunov function $V(\cdot)$ can be obtained via the standard approach of maximizing the log-likelihood (10). However, if we assume a parameterized stage cost $l_\theta(\cdot)$, we can optimize jointly with respect to hyperparameters $\boldsymbol{\psi}$ and cost parameters $\boldsymbol{\theta}$

$$\min_{\boldsymbol{\psi},\boldsymbol{\theta}} \log p(\boldsymbol{l}_\theta(\boldsymbol{X}_{k+1})|\boldsymbol{X}_{k+1},\boldsymbol{X}_k), \tag{17}$$

where the log-likelihood is given by

$$\log p(\boldsymbol{l}_\theta(\boldsymbol{X}_{k+1})|\boldsymbol{X}_{k+1},\boldsymbol{X}_k) = \\ -\frac{1}{2}\boldsymbol{l}_\theta^T(\boldsymbol{X}_{k+1})\boldsymbol{\kappa}^{-1}\boldsymbol{l}_\theta(\boldsymbol{X}_{k+1}) - \frac{1}{2}\log(|\boldsymbol{\kappa}|) - \frac{N}{2}\log(2\pi) \tag{18}$$

with the abbreviation $\boldsymbol{l}_\theta(\boldsymbol{X}_{k+1}) = \left[l_\theta(\boldsymbol{x}_{k+1}^{(1)}) \ \ldots \ l_\theta(\boldsymbol{x}_{k+1}^{(M)})\right]$. This approach exhibits the advantage that the highly local approximate infinite horizon cost $V_\infty(\cdot)$, which is typically nonzero only in the proximity of training data, and the global parametric stage cost $l(\cdot)$ are jointly adapted to the data.

Remark 3: While we assume GPs with squared exponential kernels in this article, all theoretical results are directly applicable to arbitrary universal kernels [11].

Remark 4: Although the Lyapunov function $V(\cdot)$ depends on the hyperparameters $\boldsymbol{\psi}$ and the stage cost parameters $\boldsymbol{\theta}$, the fundamental properties such as the decreasing value along the training data are not influenced by them. Therefore, the Lyapunov function $V(\cdot)$ is considered nonparametric. However, the behavior away from the training data crucially depends on the hyperparameters such that the hyperparameter optimization (18) is an important step in obtaining suitable hyperparameters.

C. Reproductivity Preserving Stabilization

We pursue the optimization based approach proposed in [8] to virtually stabilize the nominal system (11) with minimal modification. Within this approach we obtain the stabilizing control $\boldsymbol{u}(\boldsymbol{x})$ through

$$\boldsymbol{u}^*(\boldsymbol{x}) = \arg\min_{\boldsymbol{u}} \frac{1}{2}\boldsymbol{u}^T\boldsymbol{u}, \tag{19a}$$

subject to:

$$\begin{aligned} V(\boldsymbol{\mu}(\boldsymbol{x})+\boldsymbol{u}) &< V(\boldsymbol{x}) \qquad \forall \boldsymbol{x} \neq \boldsymbol{0} \\ V(\boldsymbol{\mu}(\boldsymbol{x})+\boldsymbol{u}) &= V(\boldsymbol{x}) \qquad \forall \boldsymbol{x} = \boldsymbol{0}, \end{aligned} \tag{19b}$$

where $V(\cdot)$ is the nonparametric Lyapunov function (16). Although these non-convex constraints generally prevent guarantees for the global optimality of solutions, this is not a problem since local minima can trivially be obtained by setting $\boldsymbol{u}^*(\boldsymbol{x}) = -\boldsymbol{\mu}(\boldsymbol{x})$. Therefore, asymptotic stability is not affected by the non-convexity of the optimization problem which is exploited in the following theorem.

Theorem 1: The model (2) with nominal model defined through (9) and stabilizing control obtained in (19) based on the nonparametric control Lyapunov function (16) with radially unbounded stage cost $l(\cdot)$ is globally asymptotically stable and reproduces training data exactly, i.e., $\hat{\boldsymbol{f}}(\boldsymbol{x}_k^{(m)}) = \boldsymbol{x}_{k+1}^{(m)}$, for all $m = 0, \ldots, M$.

Proof 3: The function $V(\cdot)$ is positive definite and radially unbounded since the stage cost $l(\cdot)$ also satisfies these conditions. The optimization problem is always feasible since $\boldsymbol{u}^*(\boldsymbol{x}) = -\boldsymbol{\mu}(\boldsymbol{x})$ is a trivial solution and $||\boldsymbol{u}||^*$ is bounded since each mean function of the Gaussian process state space model is bounded, i.e.,

$$|\mu_i(\boldsymbol{x})| \leq \sigma_f^2\sqrt{N}||\boldsymbol{K}_{\boldsymbol{i}}^{-1}\boldsymbol{y}_i|| \quad \forall i = 1,\ldots,d,$$

with $\boldsymbol{y}_i$ from (8). Because the training set is fixed and generated by a deterministic function $\boldsymbol{f}(\cdot)$ the norm of $\boldsymbol{K}_i^{-1}\boldsymbol{y}_i$ is a finite constant. Hence, $V(\cdot)$ is a Lyapunov function and the system (2) is globally asymptotically stable. Finally, reproduction of observed training data follows from the fact that $V(\cdot)$ is decreasing along training data as shown in Lemma 2 and the exact reproduction of training data with the nominal model (11) as proven in Lemma 1.

Although we use the trivially feasible control $\boldsymbol{u}^*(\boldsymbol{x}) = -\boldsymbol{\mu}(\boldsymbol{x})$ to prove asymptotic stability, it might not lead to good local optima as starting point of numerical optimization. Therefore, we propose to choose as initial point for the numerical optimization the closest training point in the training data set (including the origin) which satisfies the stability conditions. This approach results in weak convergence to the training data as local optima in the proximity of data are more likely to be found.

Remark 5: If the stability conditions are already satified by the uncontrolled GP-SSM, the optimal control $\boldsymbol{u}^*$ is $\boldsymbol{0}$. Therefore, there is no need to solve the optimization (19) numerically and computation time can be reduced by directly using the nominal model (11).

IV. Experimental Evaluation

In order to demonstrate the flexibility of the proposed nonparametric (NP) Lyapunov function, we compare its performance to the weighted sum of asymmetric quadratic functions (WSAQF) V_{WSAQF} [7] and the sum of squares (SOS) Lyapunov function [8]. We evaluate the performance in learning the motions of the LASA handwriting dataset[2] because it is a well-established benchmark for stable nonlinear dynamical systems, which fosters comparability of the methods. The setting of our simulations is described in Section IV-A, while the results are presented in Section IV-B and discussed in Section IV-C.

A. Experimental Setting

The LASA data set consists of 24 handwriting shapes recorded with a tablet computer. For each shape 3 to 15 recordings of the same motion are in the data set with a single trajectory consisting of 150 or 250 data points. Since some of the trajectories of a single shape intersect and practically exhibit a stochastic behavior, our approach is not directly applicable to the original data. In order to ensure comparability of the control Lyapunov functions, we downsample the training data by a factor 10 to resolve this issue and obtain sparse data. For learning the GP-SSMs we add a regularizer 10^{-14} to the diagonal of the kernel matrices $\boldsymbol{K}_i$ and $\boldsymbol{\kappa}$ in order to improve numerical stability of the matrix inversion in (9) and (14), respectively.

Control Lyapunov functions: We compare the flexibility of three different Lyapunov functions:

[2] Data set is available at https://bitbucket.org/khansari/seds

- The WSAQF Lyapunov function proposed in [7] which is given by

$$V_{\mathrm{WSAQF}}(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{P}_0 \boldsymbol{x} + \sum_{l=1}^{L} \beta_l(\boldsymbol{x}) (\boldsymbol{x}^T \boldsymbol{P}_l (\boldsymbol{x} - \boldsymbol{\epsilon}_l))^2, \tag{20}$$

  where

$$\beta_l(\boldsymbol{x}) = \begin{cases} 0 & \text{if } \boldsymbol{x}^T \boldsymbol{P}_l(\boldsymbol{x} - \boldsymbol{\epsilon}_l) < 0 \\ 1 & \text{otherwise,} \end{cases} \tag{21}$$

  with positive definite matrices $\boldsymbol{P}_l$, $l = 0, \ldots, L$. We set $L = 3$ in our simulations resulting in 18 parameters.
- The SOS Lyapunov function proposed in [8] which is defined as

$$V_{\mathrm{SOS}}(\boldsymbol{x}) = \boldsymbol{m}(\boldsymbol{x})^T \boldsymbol{P}_0 \boldsymbol{m}(\boldsymbol{x}), \tag{22}$$

  where $\boldsymbol{m}(\cdot)$ is a vector of monomials and $\boldsymbol{P}_0$ is a positive definite matrix, see [14] for a detailed explanation on the SOS technique. We use monomials up to degree 2 which results in 15 free parameters.
- The proposed nonparametric Lyapunov function defined in (16) which is denoted as $V_{\mathrm{NP}}(\cdot)$ in the sequel. We employ a quadratic stage cost $l(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{P}_0 \boldsymbol{x}$ with positive definite matrix $\boldsymbol{P}_0$ such that the conditions of Theorem 1 for global asymptotic stability are met.

The parameters $\boldsymbol{P}_l$ of the WSAQF and SOS control Lyapunov function are optimized to fit the data through the minimization problem

$$\min_{\boldsymbol{P}_l} \sum_{m=1}^{M} \max\left\{0, V(\boldsymbol{x}_{k+1}^{(m)}) - V(\boldsymbol{x}_k^{(m)})\right\}. \tag{23}$$

The positive definiteness of the matrices $\boldsymbol{P}_l$ is enforced using a Cholesky decomposition and constraining the eigenvalues of it to be larger than 0.01 in all approaches.

Simulation of the stabilized models: In order to compare the flexibility in reproducing the training data exactly we simulate the dynamical systems stabilized with the different control Lyapunov functions starting at the initial points of each trajectory. The optimization (19) is solved using an interior point algorithm where the strict inequality constraint (19b) is enforced through

$$V(\boldsymbol{\mu}(\boldsymbol{x}) + \boldsymbol{u}) - V(\boldsymbol{x}) \leq -\rho \log(1 + V(\boldsymbol{x})) \tag{24}$$

with $\rho = 0.01$ in order to improve numerical robustness. The simulation of trajectories is stopped, if they reach a neighborhood $\|\boldsymbol{x}_k\| \leq 10$ or exceed 1000 steps. We measure the reproduction error Δ_{rep} between the control Lyapunov functions using the total area between the training trajectory and the simulated trajectory. In addition to these simulations, we compare the computational efficiency of different approaches. For this reason we measure the average time $\bar{t}_{\mathrm{train}}$ it takes to fit the control Lyapunov functions to the data. Furthermore, we predict the stabilized models on a uniformly spaced 100×100 grid and compare the average computation time $\bar{t}_{\mathrm{test}}$ for a non-trivial control $\boldsymbol{u}^*(\boldsymbol{x}) \neq \boldsymbol{0}$.

B. Results

The training data $\mathbb{D}$, the stabilized GP-SSMs $\hat{\boldsymbol{f}}(\boldsymbol{x})$ and the simulated trajectories $\hat{\boldsymbol{x}}_k^{(m)}$ for the S-shape of the LASA dataset are shown in Fig. 1. The square root of the control Lyapunov functions $V(\boldsymbol{x})$ are visualized by colormaps with red denoting highest and dark blue lowest values. In addition to the stabilized models, the GP-SSM without stabilization is depicted which reproduces the training trajectories exactly. The quantitative results regarding computation times and reproduction errors for the S-shape as well as the whole data set are shown in Table I. It can be clearly seen that the nonparametric Lyapunov function provides a lower reproduction error and allows even a faster optimization, while it takes significantly more time to train.

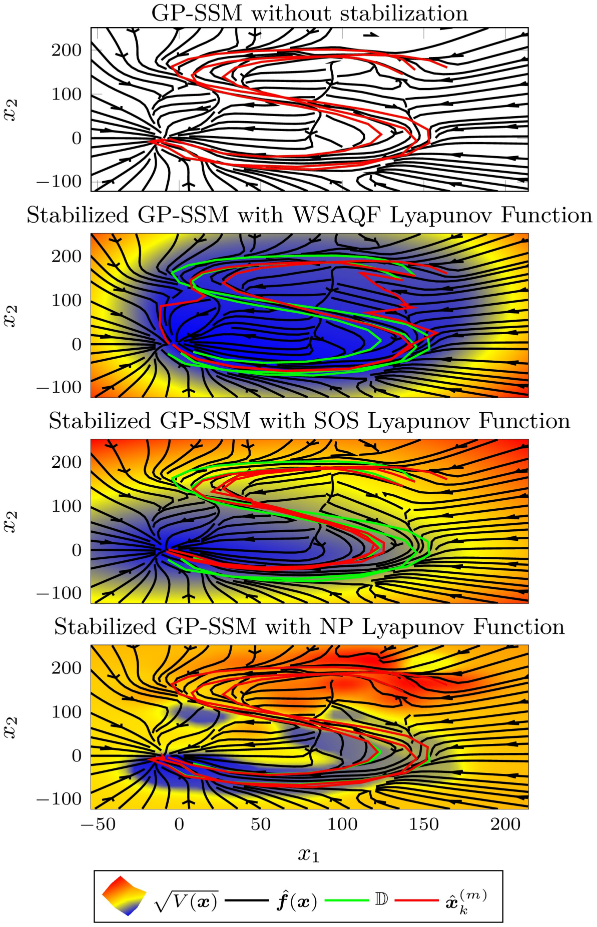

Fig. 1. GP-SSM stabilized by different control Lyapunov functions together with the training data for the LASA S-shape movement.

$V(\boldsymbol{x}_k)$	Δ_{rep} S shape	Δ_{rep} all 24	$\bar{t}_{\mathrm{train}}$	$\bar{t}_{\mathrm{test}}$
WSAQF	14276	2377	$0.7468s$	$0.0101s$
SOS	6107	1819.8	**0.4655**s	$0.0083s$
Our method ”NP”	**280.99**	**415.6**	$2.9536s$	**0.0056**s

TABLE I

REPRODUCTION ERROR AND AVERAGE COMPUTATION TIMES FOR LYAPUNOV FUNCTION TRAINING AND STABILIZING CONTROL COMPUTATION.

C. Discussion

The simulations clearly show that the nonparametric control Lyapunov function in combination with a noise-free GP-SSM allows the precise reproduction of observed training data. Since existing approaches such as SOS or WSAQF are limited by the number of used parameters, not all training samples satisfy the stability conditions. Therefore, the stabilizing control $\boldsymbol{u}(\boldsymbol{x})$ computed based on the SOS and WSAQF control Lyapunov functions can cause a deviation from the observed trajectories. In contrast, our nonparametric approach adapts its flexibility to the data. Although the nonparametric Lyapunov function exhibits local minima, this does not cause an increasing Lyapunov function along trajectories. Instead, a local minimum leads to discrete-time dynamics, which can have large differences between consecutive states, since the system must move from the local minimum to a state with smaller Lyapunov function within a single time step. Moreover, the nonparametric Lyapunov function approach relies on a precise nominal model: when the nominal model is too imprecise such that the nominal trajectories deviate significantly from the training data, the approximate infinite horizon cost $V_\infty(\cdot)$ is almost zero such that the quadratic stage cost $l(\cdot)$ dominates. Therefore, trajectories do generally not converge to training data which would be necessary for a risk-sensitive behavior with awareness of the sparsity of data. However, this does not affect stability of the obtained model and could be overcome by employing the approach proposed in [9]. Furthermore, the dominance of the quadratic cost $l(\cdot)$ exhibits also advantages regarding the computation time of the optimal controls $\boldsymbol{u}^*(\boldsymbol{x})$ such that the nonparametric Lyapunov function is the fastest on average (see Table I).

V. Conclusion

In this paper, we develop a novel approach for learning a fully nonparametric, asymptotically stable model, which is capable of precisely reproducing observed data. We show that deterministic training data can be learned exactly with GP-SSMs, and employ a nonparametric control Lyapunov function learned from the data to stabilize the nominal GP-SSM without modifying the nominal model at training points. In a comparison to existing GP-SSM stabilization approaches on a real world dataset the superior flexibility and precision of the nonparametric control Lyapunov function is demonstrated. In order to extend the applicability of the approach to systems with noisy data, we will modify the approach in future work, such that stochastic stability conditions can be considered for learning the nonparametric Lyapunov function.

VI. Acknowledgments

This work has received funding from the Horizon 2020 research and innovation programme of the European Union under grant agreement n° 871767 of the project ReHyb: Rehabilitation based on hybrid neuroprosthesis. A. L. gratefully acknowledges financial support from the German Academic Scholarship Foundation.

References

[1] A. Pentland and A. Liu, "Modeling and prediction of human behavior," *Neural Computation*, vol. 11, no. 1, pp. 229–242, 1999.

[2] S. L. Lacy and D. S. Bernstein, "Subspace identification with guaranteed stability using constrained optimization," *IEEE Transactions on Automatic Control*, vol. 48, no. 7, pp. 1259–1263, 2003.

[3] S. M. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with Gaussian mixture models," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, 2011.

[4] N. Figueroa and A. Billard, "A Physically-Consistent Bayesian Non-Parametric Mixture Model for Dynamical System Learning," in *Proceedings of the Conference on Robot Learning*, vol. 87, 2018, pp. 927–946.

[5] K. Neumann, A. Lemme, and J. J. Steil, "Neural learning of stable dynamical systems based on data-driven Lyapunov candidates," in *IEEE International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1216–1222.

[6] A. Lemme, K. Neumann, R. Reinhart, and J. Steil, "Neural Learning of Vector Fields for Encoding Stable Dynamical Systems," *Neurocomputing*, vol. 141, pp. 3–14, 2014.

[7] S. Mohammad Khansari-Zadeh and A. Billard, "Learning control Lyapunov function to ensure stability of dynamical system-based robot reaching motions," *Robotics and Autonomous Systems*, vol. 62, no. 6, pp. 752–765, 2014.

[8] J. Umlauft, A. Lederer, and S. Hirche, "Learning Stable Gaussian Process State Space Models," in *Proceedings of the American Control Conference*, 2017, pp. 1499–1504.

[9] L. Pöhler, J. Umlauft, and S. Hirche, "Uncertainty-based Human Motion Tracking with Stable Gaussian Process State Space Models," *IFAC-PapersOnLine*, vol. 51, no. 34, pp. 8–14, 2019.

[10] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA: The MIT Press, 2006.

[11] I. Steinwart, "On the Influence of the Kernel on the Consistency of Support Vector Machines," *Journal of Machine Learning Research*, vol. 2, pp. 67–93, 2001.

[12] J. Umlauft, L. Pöhler, and S. Hirche, "An Uncertainty-Based Control Lyapunov Approach for Control-Affine Systems Modeled by Gaussian Process," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 483–488, 2018.

[13] A. Lederer and S. Hirche, "Local Asymptotic Stability Analysis and Region of Attraction Estimation with Gaussian Processes," in *Proceedings of the IEEE Conference on Decision and Control*, 2019.

[14] A. Papachristodoulou and S. Prajna, "A Tutorial on Sum of Squares Techniques for Systems Analysis," in *Proceedings of the American Control Conference*, 2005, pp. 2686–2700.

Networked Online Learning for Control of Safety-Critical Resource-Constrained Systems based on Gaussian Processes

Armin Lederer, Mingmin Zhang, Samuel Tesfazgi and Sandra Hirche

***Abstract*— Safety-critical technical systems operating in unknown environments require the ability to quickly adapt their behavior, which can be achieved in control by inferring a model online from the data stream generated during operation. Gaussian process-based learning is particularly well suited for safety-critical applications as it ensures bounded prediction errors. While there exist computationally efficient approximations for online inference, these approaches lack guarantees for the prediction error and have high memory requirements, and are therefore not applicable to safety-critical systems with tight memory constraints. In this work, we propose a novel networked online learning approach based on Gaussian process regression, which addresses the issue of limited local resources by employing remote data management in the cloud. Our approach formally guarantees a bounded tracking error with high probability, which is exploited to identify the most relevant data to achieve a certain control performance. We further propose an effective data transmission scheme between the local system and the cloud taking bandwidth limitations and time delay of the transmission channel into account. The effectiveness of the proposed method is successfully demonstrated in a simulation.**

I. INTRODUCTION

Technical systems are required to operate increasingly autonomously in uncertain environments. For ensuring safety and high performance, these systems need to be able to infer models from observed data online, such that they can quickly adapt to new situations. This is particularly important in applications such as the safe control of autonomous underwater vehicles [1], unmanned aerial vehicles [2] and wearable robots [3], where uncertainty arising from humans in the control loop and changing environments can prevent the derivation of accurate models prior to system operation.

Gaussian process (GP) regression is a supervised machine learning method, which is commonly employed in highly nonlinear, safety-critical applications due to its high expressiveness and probabilistically bounded prediction errors [4]. Even though it admits closed-form updates allowing online learning and thereby an iterative adaptation of inferred models, it exhibits a quadratic update complexity in the number of training samples. Therefore, it becomes too slow for processing streaming data generated during system operation in real-time, since controllers often run at sampling rates in the magnitude of 10^2 Hz to 10^3 Hz and consequently measurements quickly accumulate to large data sets, which render exact inference computationally intractable [5]. In order to reduce the complexity of GPs, several approximations for online learning have been developed, which include inducing point methods [6], variational inference approaches [7] and finite feature approximations [8]. While these approaches can yield computation times low enough for online learning in control, beneficial safety-relevant theoretical properties of exact GPs such as uniform error bounds [9] do not directly extend to them, and thus, they cannot be used in safety-critical applications. In addition, those approaches exhibit a linear or even higher order polynomial memory complexity, which prohibits their application in resource-constrained technical systems such as drones, autonomous underwater vehicles or wearable robots with limited memory for storing data. In summary, there is a significant gap between the principle potential of GPs and their realistic application in safety-critical systems.

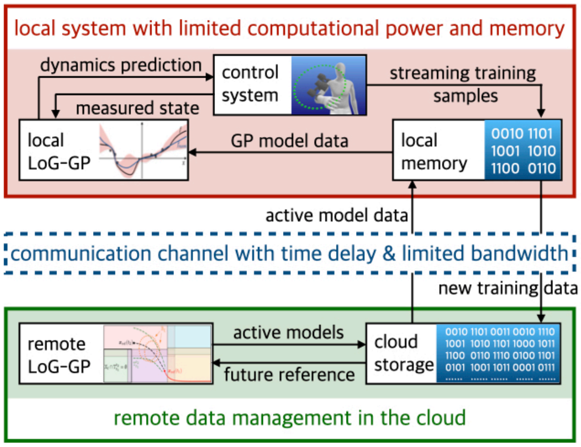

Fig. 1. Overview of the proposed networked online learning architecture: The LoG-GP predicts the unknown dynamics, e.g., of a wearable robot, for a measured state. For computing these predictions, it can only access GP model data in the local memory. Measurements of the system are continuously stored in the local memory and regularly sent to the cloud, where necessary models for a future reference trajectory are determined using a sampling based approach and corresponding data is sent to the local memory.

This paper addresses the problem of online learning control for safety-critical systems with limited computational and memory resources. We exploit the fact that our envisioned applications are able to communicate with external infrastructure including clouds with potentially unlimited data storage. Since realistic communication network restrictions such as time delays and limited bandwidth prevent the full externalization of the online model inference, we propose to exploit networked cloud computing only partially for determining and storing GP models. Our approach, which

This work was supported by the European Research Council (ERC) Consolidator Grant ”Safe data-driven control for human-centric systems (CO-MAN)” under grant agreement number 864686 and by the Horizon 2020 research and innovation programme of the European Union under grant agreement number 871767 of the project ReHyb.

The authors are with the Department of Electrical and Computer Engineering, Technical University of Munich, 80333 Munich, Germany [armin.lederer, mingmin.zhang, samuel.tesfazgi, hirche]@tum.de

is illustrated in Fig. 1, transmits data to the remote compute system, where a tree structure with localized GP models at its leaf nodes is iteratively build up using a method developed by the authors, which is called locally growing random tree of GPs (LoG-GP)[1] [10]. A sampling-based reachability analysis is executed in the cloud to determine the localized GP models which need to be communicated back to the resource-constrained system. Thereby, predictions and model updates can be computed locally without any delays. We ensure the timely availability of required data on the local system using an effective transmission scheme. This scheme provides insight on fundamental trade-offs between the bandwidth, time delays, local memory and achievable tracking error. By employing a smart aggregation method for the predictions of the local GP models, prediction error bounds of GPs are inherited. Therefore, the safety guarantees derived for control with exact GP models are maintained despite computational and memory limitations. The effectiveness of the developed networked online learning method is demonstrated in close-to-real simulations of a robotic exoskeleton.

The remainder of this paper is structured as follows: Section II formally describes the considered problem, followed by the proposed networked online learning method based on GPs in Section III. In Section IV, a tracking error bound for an online learning feedback linearizing control law is exemplarily derived, such that the effectiveness of the networked online learning approach can be demonstrated in Section V, before the paper is concluded in Section VI.

II. PROBLEM DESCRIPTION

Since accurate models for many systems such as autonomous underwater vehicles and wearable robots are often not available in practice, we consider the problem of inferring a dynamics model online from measurements generated during operation, such that the tracking performance of model-based control can be improved. Formally, we model these systems with differential equations of the form[2]

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}), \tag{1}$$

where $\boldsymbol{x} \in \mathbb{X} \subset \mathbb{R}^{d_x}$ denotes the state, $\boldsymbol{u} \in \mathbb{U} \subset \mathbb{R}^{d_u}$ is the control input, and $\boldsymbol{f} : \mathbb{X} \times \mathbb{U} \to \mathbb{R}^{d_x}$ is the unknown dynamics function. We consider the task of tracking a bounded, continuously differentiable reference trajectory $\boldsymbol{x}_{\text{ref}} : \mathbb{R}_{0,+} \to \mathbb{X}$ with the system state $\boldsymbol{x}(t)$. For this purpose, we employ a model-based control law $\boldsymbol{\pi}_{\hat{f}} : \mathbb{X} \to \mathbb{U}$, where $\hat{\boldsymbol{f}} : \mathbb{X} \times \mathbb{U} \to \mathbb{R}^d$ is a model of the unknown function $\boldsymbol{f}(\cdot)$. The tracking performance of such a control law typically depends strongly on the accuracy of the model $\hat{\boldsymbol{f}}(\cdot)$, such that we employ the following assumption on the model-based control law $\boldsymbol{\pi}_{\hat{f}}(\cdot)$, which is satisfied by many control techniques such as feedback linearization [11], backstepping [12] and adaptive control [13].

Assumption 1: The tracking error $\boldsymbol{e}(t) = \boldsymbol{x} - \boldsymbol{x}_{\text{ref}}(t)$ is ultimately bounded with monotonously increasing ultimate bound $\vartheta : \mathbb{R}_{0,+} \to \mathbb{R}_{0,+}$, i.e., for every $c \in \mathbb{R}_+$, there exists a time $T = T(c, \vartheta)$, such that it holds that

$$\|\boldsymbol{e}(0)\| \le c \qquad \Rightarrow \qquad \|\boldsymbol{e}(t)\| \le \vartheta(\kappa_t),\ \forall t \ge T, \tag{2}$$

where $\kappa_t = \max_{t' \in [0,t]} \|\boldsymbol{f}(\boldsymbol{x}(t')) - \hat{\boldsymbol{f}}(\boldsymbol{x}(t'))\|$.

For notational simplicity, we assume no knowledge of $\boldsymbol{f}(\cdot)$ before system operation, but considering a prior model $\hat{\boldsymbol{f}}_0(\cdot)$ is straightforward [11]. In order to infer a model $\hat{\boldsymbol{f}}(\cdot)$ online, we require periodical measurements of the system.

Assumption 2: Data pairs $(\boldsymbol{x}_n, \boldsymbol{y}_n = \boldsymbol{f}(\boldsymbol{x}_n, \boldsymbol{\pi}_{\hat{\boldsymbol{f}}}(\boldsymbol{x}))) + \boldsymbol{\epsilon}_n)$, where $\boldsymbol{\epsilon}_n \sim \mathcal{N}(0, \sigma_y^2 \boldsymbol{I}_{d_x})$ are i.i.d. Gaussian random variables with variance $\sigma_y^2 \in \mathbb{R}_+$, are sampled at time instances $t^{(n)} = n\tau$ with sampling time $\tau \in \mathbb{R}_+$. The data is aggregated in a time-varying training set $\mathbb{D}_t = \{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1}^{N(t) = \lfloor \frac{t}{\tau} \rfloor}$.

Assumption 2 admits training targets $\boldsymbol{y}$ perturbed by Gaussian noise, which is a frequently found assumption in literature, see, e.g., [11], [12], [13]. It also requires noise-free state measurements for training, which however, is commonly assumed in many employed control schemes such as feedback linearization and sliding mode control [14].

Since Assumption 2 ensures a continuous data stream, model updates of $\hat{\boldsymbol{f}}(\cdot)$ must be computed fast enough to avoid that data is generated at higher rates than it can be processed. Hence, the average update time T_{up} of $\hat{\boldsymbol{f}}(\cdot)$ must satisfy the computational constraint

$$T_{\text{up}} \le \tau. \tag{3}$$

Additionally, the continuous stream of data leads to a steadily growing size of the data set $\mathbb{D}_t$. Therefore, the amount of generated data will eventually reach the memory limitations, which are unavoidable on all real-world systems. Formally, this can be modelled via the memory constraint

$$|\mathbb{D}_t^{\text{loc}}| \le \bar{M}, \tag{4}$$

where $\mathbb{D}_t^{\text{loc}}$ denotes the data set stored in the memory of the technical system and $\bar{M} \in \mathbb{N}$ represents the memory limitations. Since this restriction can crucially limit the achievable control performance [15], we consider that data can be transferred to a cloud via a network connection, effectively extending the overall memory capacity. The available memory in the cloud is usually significantly larger than on the local system, such that we assume it to be infinite for simplicity. However, the data transfer between the cloud and the local system takes non-negligible time in practice due to effects such as network delays $T_d \in \mathbb{R}_+$ and finite bandwidth $B \in \mathbb{R}_+$. Therefore, data sent to the cloud cannot be immediately accessed by the local system, but the time T_{access} between requesting data $\mathbb{D}$ and using it has to satisfy the network constraint

$$T_{\text{access}} \ge \frac{|\mathbb{D}|}{B} + T_d. \tag{5}$$

Despite these restrictions, the model-based control law $\boldsymbol{\pi}_{\hat{\boldsymbol{f}}}(\cdot)$ using the model $\hat{\boldsymbol{f}}(\cdot)$ learned from the streaming data $\mathbb{D}_t$ should achieve a high tracking control performance. Therefore, we consider the problem of developing a

[1] Open-source software packages for several programming languages are available at `https://gitlab.lrz.de/online-GPs/LoG-GPs`.

[2] Notation: Lower/upper case bold symbols denote vectors/matrices, $\mathbb{R}_+$/$\mathbb{R}_{0,+}$ all real positive/non-negative numbers, $\boldsymbol{I}_n$ the $n \times n$ identity matrix, $\|\cdot\|$ the Euclidean norm, $|\mathbb{D}|$ the cardinality of a set $\mathbb{D}$, and $\lceil\cdot\rceil$/$\lfloor\cdot\rfloor$ the ceil/floor operator.

networked online learning method for inferring a highly accurate model $\hat{\boldsymbol{f}}(\cdot)$ of the unknown dynamics $\boldsymbol{f}(\cdot)$ under computational, memory and network constraints.

III. NETWORKED ONLINE LEARNING BASED ON GAUSSIAN PROCESSES

Since the time delay T_d prevents externalizing the online learning, we propose the networked online learning approach outlined in Fig. 1, which performs inference locally, but transfers unnecessary data to the cloud. The approach is based on GP regression [4] due to its strong theoretical foundation as introduced in Section III-A. For enabling online learning with GPs, we employ LoG-GPs firstly proposed in our earlier work [10], which inherit the probabilistic prediction error guarantees of exact GPs while having merely logarithmically increasing update and prediction complexities as outlined in Section III-B. In order to transmit data to the cloud without performance loss, we exploit the modular structure of LoG-GPs and determine the region, in which system states can potentially be in a given time interval, using a sampling-based approach in Section III-C. By developing a data transmission scheme in Section III-D, we ensure that necessary data is always locally available despite transmission bandwidth limitations and network delays. For notational simplicity, the proposed method is presented for scalar functions $f(\cdot)$, but can be employed for the vector-valued dynamics in (1) by applying it to each dimension individually.

A. Gaussian Process Regression

A Gaussian process is an infinite collection of random variables, any finite subset of which follows a joint Gaussian distribution [4]. The GP is usually denoted as $\mathcal{GP}(m(\cdot), k(\cdot,\cdot))$, where $m : \mathbb{R}^d \to \mathbb{R}$ is a prior mean incorporating a priori knowledge such as approximate models, and $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_{0,+}$ is a covariance function reflecting information such as periodicity. Since we assume no prior knowledge, the prior mean $m(\cdot)$ is set to 0 in the sequel. Analogously, we employ the probably most common choice for the covariance function: the squared exponential kernel $k(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f^2 \exp(-\sum_{i=1}^d (x_i - x_i')^2/(2l_i^2))$, where $\sigma_f \in \mathbb{R}_+$ denotes the signal standard deviation, and $l_i \in \mathbb{R}_+$, $i = 1, \dots, d$ are length scales [4].

Given a prior GP $\mathcal{GP}(0, k(\cdot,\cdot))$, regression is performed by conditioning on the training data $\mathbb{D}_t$ as introduced in Assumption 2. The resulting posterior distribution is again Gaussian with mean and variance given by

$$\mu(\boldsymbol{x}) = \boldsymbol{k}^T(\boldsymbol{x}) \left(\boldsymbol{K} + \sigma_y^2 \boldsymbol{I}\right)^{-1} \boldsymbol{y} \tag{6}$$

$$\sigma^2(\boldsymbol{x}) = k(\boldsymbol{x}, \boldsymbol{x}) - \boldsymbol{k}^T(\boldsymbol{x}) \left(\boldsymbol{K} + \sigma_y^2 \boldsymbol{I}\right)^{-1} \boldsymbol{k}(\boldsymbol{x}), \tag{7}$$

where the elements of $\boldsymbol{K} \in \mathbb{R}^{N \times N}$ and $\boldsymbol{k}(\boldsymbol{x}) \in \mathbb{R}^N$ are defined through $K_{i,j} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $k_i(\boldsymbol{x}) = k(\boldsymbol{x}, \boldsymbol{x}_i)$, respectively, and we concatenate training targets $\boldsymbol{y} = [y_1 \ \cdots \ y_N]^T$.

B. Locally Growing Random Tree of Gaussian Processes

Since the update complexity of Gaussian process regression scales quadratically with the number of training samples, we employ the recently proposed approach of locally growing random trees of GPs [10], which preserves beneficial properties of exact GP inference such as the existence of uniform prediction error bounds. LoG-GPs rely on the idea of iteratively constructing a tree, whose leaf nodes contain locally active GP models. In detail, the construction starts with a single GP model, which is updated with incoming streaming data until a prescribed threshold of training samples $\bar{N}$ is reached. When the GP model contains $\bar{N}$ training samples in its data set $\mathbb{D}_0$, the data set is split into 2 subsets $\mathbb{D}_i$, $i = 1, 2$, by assigning data in $\mathbb{D}_0$ to a subset $\mathbb{D}_i$ via sampling from a Lipschitz continuous probability function $p^0 : \mathbb{R}^d \to [0, 1]$. Thereby, a tree with 2 leaf nodes is generated, which contain all the data, such that individual GP models can be efficiently computed using (6) and (7). New streaming data obtained after the splitting can be assigned to the subsets $\mathbb{D}_i$ by sampling from $p^0(\cdot)$ again until either of the subsets $\mathbb{D}_i$ reaches the capacity limit $\bar{N}$. Then, a new probability function $p^i : \mathbb{R}^d \to [0, 1]$ is defined to distribute the data to new subsets, thereby extending the tree of GPs by a new layer. By repeating this procedure every time a subset $\mathbb{D}_i$ reaches $\bar{N}$ training samples, a tree of GP models is iteratively constructed with a computational complexity of $\mathcal{O}_p(\log(N))$ allowing updates with rates up to 1kHz [10], which is fast enough to satisfy the computational constraint (3) in many systems.

For computing predictions with LoG-GPs, we simply multiply the probabilities $p^i(\boldsymbol{x})$ along a path to a leaf node l to obtain the weight $\omega_l(\boldsymbol{x})$. Then, a generalized product of experts aggregation scheme [5] can be employed to obtain the approximate GP prediction

$$\tilde{\mu}(\boldsymbol{x}) = \sum_{l \in \mathbb{L}} \frac{\omega_l(\boldsymbol{x}) \tilde{\sigma}^2(\boldsymbol{x})}{\sigma_l^2(\boldsymbol{x})} \mu_l(\boldsymbol{x}), \quad \tilde{\sigma}^{-2}(\boldsymbol{x}) = \sum_{l \in \mathbb{L}} \frac{\omega_l(\boldsymbol{x})}{\sigma_l^2(\boldsymbol{x})}, \tag{8}$$

where $\mathbb{L}$ denotes the set of leaf nodes of the tree of GP models. By defining the probability functions such that only a single child node has a positive probability in most of the input domain $\mathbb{R}^d$, most of the weights $\omega_l(\boldsymbol{x})$ become 0. Since the definition of the aggregated mean $\tilde{\mu}(\cdot)$ implies that the local GP predictions $\mu_l(\boldsymbol{x})$ and $\sigma_l^2(\boldsymbol{x})$ must only be computed if $\omega_l(\boldsymbol{x}) > 0$, models with $\omega_l(\boldsymbol{x}) = 0$ can be considered locally inactive at $\boldsymbol{x}$ and therefore, aggregated predictions can be efficiently computed in $\mathcal{O}_p(\log^2(N))$ complexity. Moreover, this construction of the aggregated prediction $\tilde{\mu}(\cdot)$ ensures that uniform error bounds are directly inherited from exact GP regression.

Lemma 1 ([10]): Assume the function $f : \mathbb{R}^d \to \mathbb{R}$ is a sample from a Gaussian process $\mathcal{GP}(0, k(\cdot,\cdot))$ with a L_k-Lipschitz kernel $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. Then, the aggregated mean function (8) of a LoG-GP trained with data satisfying Assumption 2 guarantees a probabilistically, uniformly bounded prediction error on a compact domain $\Omega \subset \mathbb{R}^d$, i.e., for $\delta \in (0, 1)$ and $\rho \in \mathbb{R}_+$, we have

$$P(|f(\boldsymbol{x}) - \tilde{\mu}(\boldsymbol{x})| \leq \eta(\boldsymbol{x}), \forall \boldsymbol{x} \in \Omega) \geq 1 - \delta, \tag{9}$$

where

$$\eta(\boldsymbol{x}) = \sqrt{\beta(\delta, \rho)} \sum_{l \in \mathbb{L}} \frac{\omega_l(\boldsymbol{x}) \tilde{\sigma}^2(\boldsymbol{x})}{\sigma_l(\boldsymbol{x})} + \gamma(\rho) \tag{10}$$

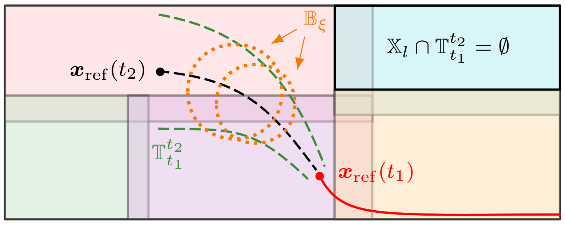
$\mathbb{B}_\xi$
$\mathbb{X}_l \cap \mathbb{T}_{t_1}^{t_2} = \emptyset$
$\boldsymbol{x}_{\mathrm{ref}}(t_2)$
$\mathbb{T}_{t_1}^{t_2}$
$\boldsymbol{x}_{\mathrm{ref}}(t_1)$

Fig. 2. A local model $l \in \mathbb{L}$ is inactive if its active region $\mathbb{X}_l$ does not intersect with the tube $\mathbb{T}_{t_1}^{t_2}$ induced by the tracking error bound $\vartheta(\kappa_t)$ as illustrated for the region in the top right. The set of active models $\hat{\mathbb{A}}$ is found by over-approximating the tube $\mathbb{T}_{t_1}^{t_2}$ with balls $\mathbb{B}_\xi$, from which random samples $\boldsymbol{x}^{(i)}$ are drawn to determine the active models $\mathbb{A}_{\boldsymbol{x}^{(i)}}$ at these states.

$$\beta(\delta,\rho) = 2\log\left(d^{\frac{d}{2}}|\mathbb{L}| \max_{\boldsymbol{x},\boldsymbol{x}'\in\mathbb{R}^d} \|\boldsymbol{x}-\boldsymbol{x}'\|_\infty^d\right) - 2\log\left(\delta 2^d \rho^d\right) \quad (11)$$

$$\gamma(\rho) = \sum_{l\in\mathbb{L}} \frac{\omega_l \tilde{\sigma}^2(\boldsymbol{x})}{\sigma_l^2(\boldsymbol{x})}\left(L_{\mu_l}\rho + \sqrt{\beta(\rho)}L_{\sigma_l}\tau\right) + L_f\rho, \quad (12)$$

and L_f, L_{μ_l}, L_{σ_l} are Lipschitz constants of $f(\cdot)$, $\mu_l(\cdot)$, $\sigma_l(\cdot)$.

This result relies on a well-calibrated prior GP, which is a rather unrestrictive assumption in practice since the sample spaces of GPs are very expressive for many frequently used kernels, e.g., the space of continuous functions for GPs with a squared exponential kernel [9]. Since the posterior variances $\sigma_l^2(\boldsymbol{x})$ are guaranteed to converge to 0 for dense data [16], this result ensures that arbitrarily accurate models can be obtained. Therefore, LoG-GPs are accompanied by strong theoretical guarantees for their prediction accuracy as required in safety-critical applications.

C. Sampling-Based Identification of Active Models

Since Lemma 1 ensures bounded prediction errors when using (8) to learn a model $\hat{\boldsymbol{f}}(\cdot)$ of the unknown dynamics $\boldsymbol{f}(\cdot)$, we can determine the system states $\boldsymbol{x}$ which can be potentially reached within a fixed time interval using the tracking error bound $\vartheta(\kappa_t)$ introduced in Assumption 1. Therefore, we can obtain the models, which need to be available in the local memory, by finding all individual GP models which are active for states $\boldsymbol{x}$ in the potentially reachable set.

In detail, this set $\mathbb{A}$ of potentially active models during a time window $\mathbb{W} = [t_1, t_2]$, $t_1, t_2 \in \mathbb{R}$, $t_2 > t_1$, is defined through the intersections between active regions $\mathbb{X}_l = \{\boldsymbol{x} : \omega_l(\boldsymbol{x}) > 0\}$ of local models $l \in \mathbb{L}$ and the tube $\mathbb{T}_{t_1}^{t_2} = \{\boldsymbol{x} \in \mathbb{R}^d : \exists t \in \mathbb{W}, \boldsymbol{x} \in \mathbb{B}_{\vartheta(\kappa_t)}\}$ based on balls $\mathbb{B}_{\vartheta(\kappa_t)} = \{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x} - \boldsymbol{x}_{\mathrm{ref}}(t)\| \leq \vartheta(\kappa_t)\}$ with radius given by Assumption 1, i.e., $\mathbb{A} = \{l \in \mathbb{L} : \mathbb{X}_l \cap \mathbb{T}_{t_1}^{t_2} \neq \emptyset\}$ as illustrated in Fig. 2. Since the computation of the intersections $\mathbb{X}_l \cap \mathbb{T}_{t_1}^{t_2}$ requires an explicit representation of the active regions $\mathbb{X}_l$ of local models $l \in \mathbb{L}$, which is not provided by LoG-GPs, the definition of $\mathbb{A}$ cannot be directly used in practice. We follow a different idea exploiting the implicit representation of the active regions $\mathbb{X}_l$ via the weights $\omega_l(\cdot)$, which allows to directly compute the set of active models $\mathbb{A}_{\boldsymbol{x}} = \{l \in \mathbb{L} : \omega_l(\boldsymbol{x}) > 0\}$ for a given state $\boldsymbol{x}$. Therefore, we can alternatively represent the set of potentially active models during the time window $\mathbb{W}$ via $\mathbb{A} = \bigcup_{t\in\mathbb{W}} \bigcup_{\boldsymbol{x}\in\mathbb{B}_{\vartheta(\kappa_t)}} \mathbb{A}_{\boldsymbol{x}}$. By approximating the unions over uncountable sets via discretization and random sampling as outlined in Algorithm 1, we can over-approximate the set $\mathbb{A}$ via $\hat{\mathbb{A}}$ and obtain

Algorithm 1: Determining Active Models

```
Function ActiveModels($N_s$, $t_1$, $t_2$, $\Delta t$):
    $\hat{\mathbb{A}} \leftarrow \emptyset$
    compute $\xi$ using (14)
    for $j = 0 : \lceil \frac{t_2 - t_1}{\Delta t} \rceil$ do
        for $i = 1 : N_s$ do
            Determine active models $\mathbb{A}_{\boldsymbol{x}^{(i)}}$ for input $\boldsymbol{x}^{(i)} \sim \mathcal{U}(\mathcal{B}_\xi)$
            $\hat{\mathbb{A}} \leftarrow \hat{\mathbb{A}} \cup \mathbb{A}_{\boldsymbol{x}^{(i)}}$
    return $\hat{\mathbb{A}}$
```

$$\hat{\mu}(\boldsymbol{x}) = \sum_{l\in\hat{\mathbb{A}}} \frac{\omega_l(\boldsymbol{x})\hat{\sigma}^2(\boldsymbol{x})}{\sigma_l^2(\boldsymbol{x})}\mu_l(\boldsymbol{x}), \quad \hat{\sigma}^{-2}(\boldsymbol{x}) = \sum_{l\in\hat{\mathbb{A}}} \frac{\omega_l(\boldsymbol{x})}{\sigma_l^2(\boldsymbol{x})}. \quad (13)$$

If sufficiently many samples are used, this approximation yields identical predictions as shown in the following result.

Theorem 1: Consider a dynamical system (1) and assume Assumptions 1 and 2 hold. Choose $\|\boldsymbol{e}(t_1)\| \leq \vartheta(\kappa_{t_1})$ and

$$\xi = 2\zeta + L_{\boldsymbol{x}_{\mathrm{ref}}}\frac{\Delta t}{2} + \vartheta(\kappa_{t_1+j\Delta t}) \quad (14)$$

for constants $\zeta, \Delta t \in \mathbb{R}_+$. Then, with probability of at least

$$1 - \|\mathbb{L}\| \left\lceil \frac{t_2 - t_1}{\Delta t} \right\rceil \left(1 - \frac{\min\{r_{\min}^{d_x}, \zeta^{d_x}\}}{\xi^{d_x}}\right)^{N_s}, \quad (15)$$

where $r_{\min}$ denotes the radius of the largest ball contained in the smallest active region of a leaf node $l \in \mathbb{L}$, the predictions $\hat{\mu}(\boldsymbol{x}(t))$ and $\tilde{\mu}(\boldsymbol{x}(t))$ are identical for all $t \in \mathbb{W}$.

Proof: See Appendix A. ■

Since this theorem ensures that (8) and (13) are identical with probability greater than (15), it ensures that using $\hat{\mu}(\cdot)$ as model in a control law $\boldsymbol{\pi}_{\hat{\boldsymbol{f}}}(\cdot)$ yields no reduction in control performance with high probability. Therefore, it allows us to determine irrelevant data for a time interval $\mathbb{W}$, which we exploit in the following section for transmitting data to the cloud, thereby reducing the local memory occupation.

D. Transmission Scheme

Due to the non-negligible time required for a data transfer, the transmission to and from the cloud must be carefully scheduled in order to ensure that the necessary data is always available locally. For simplicity, we consider that data is transmitted at regularly spaced time instances $j\Delta T$, $j = \mathbb{N}$, such that each time interval $\mathbb{W}_j = [(j-1)\Delta T, j\Delta T]$ has a length of $\Delta T \in \mathbb{R}_+$. During each time interval $\mathbb{W}_j$, we propose the transmission scheme illustrated in Fig. 3, where the idea is that the memory is divided into two parts. During each interval $\mathbb{W}_j$, half of the memory is used for updating the local data set with data from the cloud, while the other half contains the data set $\mathbb{D}_j$ necessary for computing the mean predictions $\hat{\mu}(\cdot)$ during time interval $\mathbb{W}_j$ according to the potentially active models $\hat{\mathbb{A}}_j$. For updating the local memory, the data set $\mathbb{D}_{j-1}$ from the previous interval $\mathbb{W}_{j-1}$, which contains newly measured training samples as well as data from the cloud, is sent to the cloud. Once this transmission has been completed, the cloud contains the complete data set $\mathbb{D}_{(j-1)\Delta T}$ obtained until time $(j-1)\Delta T$, such that Algorithm 1 can be employed to determine the possibly active models $\hat{\mathbb{A}}_{j+1}$ for the next time interval $\mathbb{W}_{j+1}$

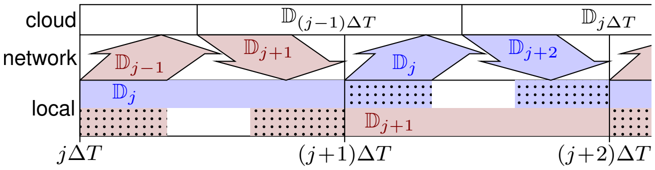

Fig. 3. During each interval $\mathbb{W}_j = [j\Delta T, (j+1)\Delta T]$, the previously necessary data $\mathbb{D}_{j-1}$ is sent to the cloud and the data $\mathbb{D}_{j+1}$ for the next interval $\mathbb{W}_{j+1}$ is fetched. While these data sets occupy memory during the interval $\mathbb{W}_j$, parts of $\mathbb{D}_{j-1}$ and $\mathbb{D}_{j+1}$ are in transmission and not available on the local system. Therefore, these data sets cannot be used for prediction, which is highlighted through the dotted pattern. The data in the cloud is updated with incoming transmissions, such that it contains the complete data set $\mathbb{D}_{(j-1)\Delta T}$ up to the end of previous interval $j-1$.

Algorithm 2: Data Transfer Scheme: Cloud

```
Function UpdateLoop($\Delta T$, $\tau$, $\Delta t$):
    for $n = 1, \ldots, \infty$ do
        if $n\tau \geq j\Delta T$ then
            $j \leftarrow j + 1$
            $\mathbb{D}_{j-1} \leftarrow$Receive ()
            $\hat{\mathbb{A}}_{j+1} \leftarrow$ActiveModels($N_s, j\Delta T, (j+1)\Delta T, \Delta t$)
            Transmit ($\mathbb{D}_{j+1}$)
```

Algorithm 3: Data Transfer Scheme: Local System

```
Function UpdateLoop($\Delta T$, $\iota$, $\tau$):
    for $n = 1, \ldots, \infty$ do
        if $n\tau \leq \iota\Delta T$ then
            $\mathbb{D}_t^{\mathrm{loc}} \leftarrow \mathbb{D}_t^{\mathrm{loc}} \cup (\boldsymbol{x}^{(n)}, y^{(n)})$
        if $n\tau \geq j\Delta T$ then
            $j \leftarrow j + 1$
            Transmit ($\mathbb{D}_{j-1}$)
            Delete ($\mathbb{D}_{j-1}$)
            $\mathbb{D}_{j+1} \leftarrow$Receive ()
```

in the cloud. The corresponding data set $\mathbb{D}_{j+1}$ is sent to the local memory, such that it is available for $t \geq (j+1)\Delta T$.

It is straightforward to see that this transmission scheme can ensure the satisfaction of the network constraint (5) for a fixed data set $\mathbb{D}_j$, if $T_{\text{access}} = \Delta T/2$ is sufficiently large. However, due to the online generation of data during system operation, it generally cannot be ensured that the data sets $\mathbb{D}_j$ have a bounded size, such that the fixed time ΔT might eventually not be sufficient to finish the transmission within the time interval $\mathbb{W}_j$. Therefore, the real-time learning with data generated online during system operation has to be stopped eventually at some interval $\mathbb{W}_\iota$, $\iota \in \mathbb{N}$ in order to upper bound the size of all sets $\mathbb{D}_j$. This leads to the data transfer scheme outlined in Algorithm 2 for the cloud and in Algorithm 3 for the local system, for which it is straightforward to prove the satisfaction of the network constraint (5) as shown in the following result.

Lemma 2: Choose $\Delta T \geq \frac{\bar{M}}{B} + 2T_d$ and $\iota \in \mathbb{N}$ such that the memory constraint (4) is satisfied. Then, Algorithms 2 and 3 ensure the satisfaction of the network constraint (5).

Proof: See Appendix B. ■

In order to apply this lemma in a real-world system, it remains to develop an approach for enforcing the memory constraint (4) by choosing a suitable value of ι. In practice, this value can be selected online using heuristics such that learning can be stopped, e.g., when the number of active models exceeds a threshold. Moreover, when the reference $\boldsymbol{x}_{\text{ref}}$ is periodic, we can determine ι based on the data sets from previous periods, as shown in the following theorem.

Theorem 2: Assume the reference trajectory is periodic with period $T_p = q\Delta T$ for $\Delta T \geq \frac{\bar{M}}{B} + 2T_d$ and $q \in \mathbb{N}$. Let

$$\iota = q + \min_{|\mathbb{D}_j| > \frac{\bar{M}-2\bar{m}}{2}} j, \quad \bar{m} = \max_{j \in \mathbb{N}} |\mathbb{D}_{(j+q)}| - |\hat{\mathbb{D}}_j| \leq \left\lceil \frac{T_p}{\tau} \right\rceil. \quad (16)$$

Then, Algorithms 2 and 3 ensure the satisfaction of the memory constraint (4) and network constraint (5).

Proof: See Appendix C. ■

This theorem allows to determine online when to stop adding new training samples to the LoG-GP by checking if $|\mathbb{D}_j| > \frac{\bar{M}}{2} - \bar{m}$, which can be performed with low complexity and can be directly implemented. Moreover, it provides valuable insight into the interrelations between achievable tracking accuracy, memory constraint $\bar{M}$, time delay T_d and limited bandwidth B. In order to see this, note that the data set size $|\mathbb{D}_j|$ usually grows almost linearly with the interval length ΔT. Since an increase in bandwidth B admits smaller ΔT, learning can continue up to higher values of ι in general. Therefore, a higher data density can be achieved, which in turn yields a lower GP variance [16] guaranteeing a smaller tracking error. In contrast, an increase in local memory $\bar{M}$ admits larger data set sizes $|\mathbb{D}_j|$, but in turn requires longer intervals ΔT, such that the achievable data density and consequently the tracking accuracy are barely affected. Finally, a reduction of the delay T_d allows smaller values of ΔT and thereby also leads to an improvement in achievable control performance. Therefore, available bandwidth B for data transmission and time delay T_d are crucial for the achievable tracking accuracy when using the networked online learning control law, while finite local memory $\bar{M}$ only has secondary relevance to enable implementation of the transmission scheme using Algorithm 2 and 3. This insight can be beneficially used for the design of autonomous systems in practice, since it allows a reduction of local memory when sufficient bandwidth for data transmission is available.

IV. Efficient Tracking Error Bounds

In order to demonstrate the applicability of the proposed networked online learning approach, we exemplarily derive a tracking error bound $\vartheta(\kappa_t)$ for a feedback linearizing control law, which can be applied to a wide range of practically relevant systems such as robotic manipulators, unmanned aerial and autonomous underwater vehicles. For the derivation of $\vartheta(\kappa_t)$ we employ Lyapunov stability theory, such that computing $\vartheta(\kappa_t)$ effectively reduces to determining the regions of the state space with decreasing Lyapunov function along system trajectories. Due to the prediction error bound $\eta(\cdot)$, this decrease condition can be efficiently decoupled for feedback linearizing control laws as illustrated in Fig. 4. Thereby, we obtain a straightforwardly implementable tracking error bound, which can be directly used in Algorithms 2 and 3.

In more detail, we consider feedback linearizable systems

$$\dot{x}_1 = x_2, \quad \dot{x}_2 = x_3, \quad \ldots \quad \dot{x}_{d_x} = f(\boldsymbol{x}) + g(\boldsymbol{x})u, \quad (17)$$

where a scalar control input u as well as scalar functions

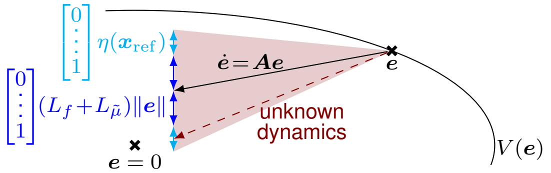
$\begin{bmatrix}0\\ \vdots\\ 1\end{bmatrix}$
$\eta(\boldsymbol{x}_{\mathrm{ref}})$
$\dot{\boldsymbol{e}}=\boldsymbol{A}\boldsymbol{e}$
$\boldsymbol{e}$
$\begin{bmatrix}0\\ \vdots\\ 1\end{bmatrix}$
$(L_f+L_{\tilde{\mu}})\|\boldsymbol{e}\|$
unknown dynamics
$\boldsymbol{e}=0$
$V(\boldsymbol{e})$

Fig. 4. Ultimate boundedness is analyzed using Lyapunov theory with Lyapunov function $V(\boldsymbol{e}) = \boldsymbol{e}^T\boldsymbol{P}\boldsymbol{e}$. For exact feedback linearization, the closed-loop dynamics are linear with $\dot{\boldsymbol{e}} = \boldsymbol{A}\boldsymbol{e}$ and ensure a decreasing Lyapunov function along system trajectories. Due to model errors, which can be expressed through the model error bound $\eta(\boldsymbol{x}_{\mathrm{ref}})$ at the reference $\boldsymbol{x}_{\mathrm{ref}}$ and the error $(L_f+L_{\tilde{\mu}})\|\boldsymbol{e}\|$ of linearization around the reference, the uncertainty in the dynamics can be bounded. Thereby, the region with decreasing Lyapunov function can be efficiently determined.

$f : \mathbb{X} \to \mathbb{R}$ and $g : \mathbb{X} \to \mathbb{R}$ are assumed only for simplicity of exposition, while all derived results straightforwardly extend to multi-input systems in the canonical form. Similar to previous work [11], we assume that $f(\cdot)$ is an unknown function, while $g(\cdot)$ is known. The knowledge of $g(\cdot)$ is merely used to streamline the presentation, but all results can be extended to unknown functions $g(\cdot)$ following the approach in [15]. In order to ensure global controllability of (1), the following assumption is needed.

Assumption 3: The function $g(\cdot)$ is positive, i.e., $g(\boldsymbol{x})>0$.

This assumption is a standard condition when designing control laws for systems in the canonical form [14, Definition 13.1], and ensures the non-singularity of $g(\cdot)$. It is naturally satisfied by many systems such as Euler-Lagrange systems, where $g(\cdot)$ corresponds to the positive inertia. Therefore, this assumption is not restrictive in practice.

Additionally, we assume that the unknown function is well-behaved, which is formalized in the following.

Assumption 4: The function $f(\cdot)$ is L_f-Lipschitz.

This assumption globally ensures a unique solution for the system (1) [14], such that it can be commonly found in control. Since it is satisfied by many systems such as Euler-Lagrange dynamics in practice, it is not restrictive.

In order to allow the accurate tracking of the reference trajectory with system (17), we consider reference trajectories

$$\boldsymbol{x}_{\mathrm{ref}}(t) = \begin{bmatrix} x_{\mathrm{ref}}(t) & \dot{x}_{\mathrm{ref}}(t) & \cdots & \frac{\mathrm{d}^{d_x-1}}{\mathrm{d}t^{d_x-1}}x_{\mathrm{ref}}(t)\end{bmatrix}^T, \quad (18)$$

where $x_{\mathrm{ref}} : \mathbb{R} \to \mathbb{R}$ is d_x times continuously differentiable. For tracking this trajectory, we employ the control law

$$u = \pi_{\mathrm{FL}}(\boldsymbol{x}) = g^{-1}(\boldsymbol{x})\Big(-\tilde{\mu}(\boldsymbol{x}) + \nu + \frac{\mathrm{d}^{d_x}}{\mathrm{d}t^{d_x}}x_{\mathrm{ref}}(t)\Big), \quad (19)$$

where the mean $\tilde{\mu}(\boldsymbol{x})$ defined in (8) is used as model. The input ν to the approximately linearized system is given by the linear feedback law $\nu = -k_c\begin{bmatrix}\lambda_1 & \cdots & \lambda_{d_x-1} & 1\end{bmatrix}\boldsymbol{e}$, where $k_c \in \mathbb{R}_+$ is the control gain and $\lambda_1, \ldots, \lambda_{d_x-1} \in \mathbb{R}$ are coefficients such that for $s \in \mathbb{C}$, the polynomial $s^{d_x-1} + \lambda_{d_x-1}s^{d_x-2} + \ldots + \lambda_1$ is Hurwitz. Due to these choices, the error dynamics can be compactly expressed by

$$\dot{\boldsymbol{e}} = \underbrace{\begin{bmatrix} \boldsymbol{0} & \boldsymbol{I} \\ -\lambda_1 k_c & -\begin{bmatrix}\lambda_2 k_c & \cdots & k_c\end{bmatrix}\end{bmatrix}}_{\boldsymbol{A}}\boldsymbol{e} + (f(\boldsymbol{x}) - \tilde{\mu}(\boldsymbol{x}))\begin{bmatrix}0\\ \vdots \\ 1\end{bmatrix}. \quad (20)$$

The matrix $\boldsymbol{A}$ defines a stable dynamical system because of the Hurwitz coefficients λ_i, which is independent of the online learning. Therefore, the second summand in (20) can be considered a disturbance depending on the online learning, which allows us to straightforwardly analyze the ultimate boundedness of this system using Lyapunov theory.

Theorem 3: Consider a dynamical system (1), where $f(\cdot)$ is a sample from a Gaussian process with L_k-Lipschitz kernel $k(\cdot,\cdot)$. Moreover, assume Assumptions 2, 3 and 4 hold, and choose a positive definite, symmetric matrix $\boldsymbol{Q} \in \mathbb{R}^{d\times d}$ and a control gain k_c such that

$$\|\boldsymbol{p}_d(k_c)\| < \frac{\lambda_{\min}(\boldsymbol{Q})}{2(L_f + L_{\tilde{\mu}})}, \quad (21)$$

where $\boldsymbol{P} = [\boldsymbol{p}_1(k_c) \ \cdots \ \boldsymbol{p}_d(k_c)]$ is the solution to the Lyapunov equation $\boldsymbol{A}^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} = -\boldsymbol{Q}$ and $L_{\tilde{\mu}}$ denotes the Lipschitz constant of the LoG-GP mean $\tilde{\mu}(\cdot)$. Then, the online learning feedback linearizing control law (19) guarantees an ultimately bounded tracking error

$$\vartheta(\kappa_t) = \frac{2\,\|\boldsymbol{p}_d(k_c)\|\,\sqrt{\lambda_{\max}(\boldsymbol{P})}\,\max_{t'\in[0,t]}\eta\left(\boldsymbol{x}_{\mathrm{ref}}(t')\right)}{\left(\lambda_{\min}(\boldsymbol{Q}) - 2\,\|\boldsymbol{p}_d(k_c)\|\,(L_f + L_{\tilde{\mu}})\right)\sqrt{\lambda_{\min}(\boldsymbol{P})}}. \quad (22)$$

Proof: See Appendix D. ■

This theorem has the advantage over previously derived results in similar settings [9], [11] that the ultimate bound $\vartheta(\kappa_t)$ in Theorem 3 depends only on the standard deviation along the reference $\boldsymbol{x}_{\mathrm{ref}}$. Thereby, the ultimate bound (22) can be efficiently computed, whereas the results in previous works provide only an implicit representation of the ultimate bound due to the dependency of the GP error bound on the state $\boldsymbol{x}$. This advantage resulting from the linearization around the reference $\boldsymbol{x}_{\mathrm{ref}}$ in (26) comes at the cost of the additional constraint (21) for the control gain k_c compared to existing approaches [9], [11]. Even though this constraint makes the application of Theorem 3 more restrictive, this weakness is strongly outweighed by the benefit of the explicit tracking error bound for determining the active models online.

Remark 1: Due to the definition of $\boldsymbol{A}$ in (20), it can be straightforwardly checked that it is always possible to ensure the satisfaction of (21) for a fixed matrix $\boldsymbol{Q}$ by choosing a sufficiently large gain k_c. Therefore, condition (21) effectively imposes a lower bound for the control gains k_c admitting ultimate tracking error bounds (22).

V. EVALUATION IN EXOSKELETON CONTROL

In order to evaluate the applicability of the proposed networked online learning approach for resource constrained systems[3], we employ it for the control of an upper-limb human-exoskeleton assisting a user in tracking a reference trajectory, which is simulated in Julia [17], a modern programming language for accelerating physics simulations. Since the exoskeleton is intended to be used in a portable manner, this scenario resembles an example for a wearable robotic system with memory and computational constraints. These constraints are particularly challenging for the control of the exoskeleton as human user data is required in practice to infer models allowing for personalized assistance.

[3] Open-source code conceptually demonstrating the proposed method is available at `https://gitlab.lrz.de/online-GPs/cloud-GPs`.

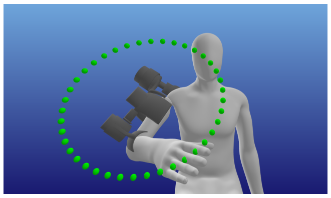

Fig. 5. Visualization of the upper-limb human-exoskeleton simulation and trajectory tracking task. The green circles depict discrete points along the elliptic reference trajectory, which must be followed with the hand.

TABLE I
BANDWIDTH B, TIME DELAY T_d, STATE MEASUREMENT STANDARD DEVIATION $\sigma_{\boldsymbol{x}}$ AND RESULTING TIME WHEN LEARNING IS STOPPED T_s

	low bandwidth	medium bandwidth	high bandwidth	large delay	state noise
B [samples/s]	1500	3000	10000	10000	1500
T_d [s]	0.1	0.1	0.1	1.0	0.1
$\sigma_{\boldsymbol{x}}$ [rad]	—	—	—	—	0.0001
T_s [s]	44.67	60.04	—	30.81	43.13

For the simulation, we assume a rigid kinematic coupling between the human and exoskeleton arm, which allows the modelling of both as one kinematic chain consisting of four DoFs. The exoskeleton model is based on the design described in [18], whilst the model parameter for the human are chosen according to anthropometric tables [19]. Here, the reference is set to $70\,\mathrm{kg}$ and $1.75\,\mathrm{m}$. As illustrated in Fig. 5, the goal is to track an elliptic trajectory with the hand of the human by employing the learning-based feedback linearizing control law (19). Each period of the ellipse takes $T_p = 6\mathrm{s}$, the simulation runs at $1\mathrm{kHz}$, and we consider a memory constraint of $\bar{M} = 4000$ data pairs for the local memory. Streaming data for online learning is generated with noise standard deviation $\sigma_y = 0.05$ at a sampling rate of $100\mathrm{Hz}$, i.e., $\tau = 10\mathrm{ms}$. Each local GP model can contain a maximum of $\bar{N} = 100$ training points and the hyperparameters are set to $\sigma_f = 1$, $l_i = 1/l_i = 3$ for inputs corresponding to joint angles/angular velocities. Algorithm 1 is run with temporal discretization $\Delta t = 10\mathrm{ms}$ and $N_s = 1000$ random samples. Finally, the control gains are set to $k_c = 400$ and $\lambda = 1$.

In order to investigate the dependency of the tracking accuracy and memory occupation on the network bandwidth B and time delay T_d, we compare networked LoG-GP controllers under different simulation conditions as outlined in Table I. In this comparison, we also consider the case of noisy state measurements to demonstrate the robustness of the proposed method against sufficiently small noise on training inputs $\boldsymbol{x}_n$. Moreover, we employ a LoG-GP without memory constraints, i.e., $\bar{M} = \infty$, as baseline to illustrate the absence of a performance loss of the networked LoG-GP when a sufficiently high bandwidth is available. The average update time for the LoG-GP is $0.3\mathrm{ms} < \tau$ in all simulations, and the resulting curves for the evolution of the local memory occupation are depicted in Fig. 6. Since the LoG-GP has low accuracy during the first period, the tracking error bound $\vartheta(\kappa)$ is large during the first 6s, such that all data is required on the local system. After this period, the different curves exhibit the behavior discussed in Section III-D: The lower the bandwidth B, the faster the local memory consumption grows. Moreover, an increase in time delay T_d causes a significantly faster growing memory occupation. Due to the limited local memory, this leads to an early stop in learning at the times depicted in Table I, after which the memory occupation stagnates. Note that the state measurement noise has effectively no impact on the local memory consumption.

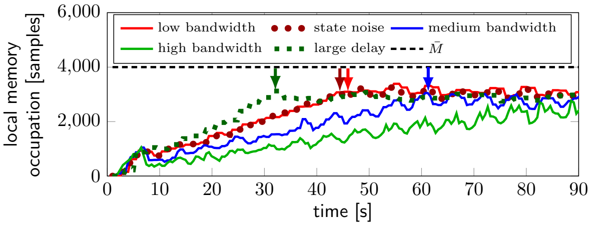

Fig. 6. The higher the bandwidth B, the longer the LoG-GP can learn before the number of training pairs in the local memory reaches the limitations. Large time delay T_d causes a significantly earlier stopping of learning, as indicated by the arrows.

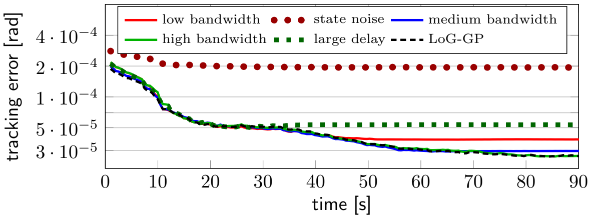

Fig. 7. When the memory limitation is reached and the learning process stops, the tracking error stagnates. Since higher bandwidths B allow learning for a longer time, larger values of B yield lower tracking errors eventually. While learning is not stopped, networked LoG-GPs ensure the same tracking accuracy as LoG-GPs without any constraints. Overall, online learning significantly improves the tracking accuracy over the baseline case without model learning, which is not depicted since it permanently exceeds $2 \cdot 10^{-2}$.

The stagnation has an immediate effect on the evolution of the tracking error, as illustrated in Fig. 7. While the feedback linearizing control law (19) with networked LoG-GP model achieves the same improvement in tracking accuracy as with the unconstrained LoG-GP when model updates are performed, the tracking performance ceases to improve and effectively remains constant after the learning has stopped. The same behavior can be observed with noisy state measurements, but the tracking error exhibits an offset since the noise also affects the linear feedback control ν. Due to the continual learning of the networked LoG-GP with a high bandwidth connection, the corresponding evolution of the tracking error is visually identical to the curve resulting from usage of the LoG-GP without memory constraint. This clearly demonstrates that the proposed approach allows a transfer of data to the cloud without any loss in performance when sufficient transmission bandwidth is available. Moreover, even when online learning has to be stopped early, it still yields a significant improvement in tracking accuracy compared to the baseline case without model learning, where a stationary error of $\approx 2 \cdot 10^{-2}$ rad has been observed. This strongly underlines the practical advantages of online model inference for model-based control despite resource constraints.

VI. CONCLUSION

This paper presents a novel networked online learning approach for control of safety-critical systems with local resource constraints based on Gaussian process regression. By employing a tree-structured local GP approximation, relevant local models for control can be efficiently determined in a sampling-based fashion. This is exploited in the design of an effective data transmission scheme, which ensures the timely availability of data in the local computing unit. The effectiveness of the proposed networked online learning approach is demonstrated in a simulation of a robotic exoskeleton.

Appendix

A. Proof of Theorem 1

Proof: Due to Assumption 1, at each time t, the tracking error $\boldsymbol{e}$ is bounded by $\vartheta(\kappa_t)$. It is straightforward to see that $\boldsymbol{x}_{\mathrm{ref}}(\cdot)$ is Lipschitz continuous, such that

$$\|\boldsymbol{e}(t)\| \leq \xi - 2\zeta \quad \forall t \in \left[t_1 + \frac{2j-1}{2}\Delta t, t_1 + \frac{2j+1}{2}\Delta t\right] \tag{23}$$

and consequently $\mathbb{T}_{t_1}^{t_2} \subset \bigcup_{j=1}^{\lceil \frac{t_2-t_1}{\Delta t} \rceil} \mathbb{B}_{\xi-2\zeta}$. Therefore, it remains to show that the set of active models for time $t_1 + j\Delta t$ defined as $\mathbb{A}_{t_1+j\Delta t} = \bigcup_{\boldsymbol{x}\in\mathcal{B}_{\xi-2\zeta}} \mathbb{A}_{\boldsymbol{x}}$ is overapproximated by Algorithm 1. For this purpose, choose any model $l \in \mathbb{A}_{t+j\Delta t}$. Then, the intersection between the active region $\mathbb{X}_l$ of this model and the ball $\mathcal{B}_\xi$ has a volume of at least $\pi^{d_x/2}(\min\{r_{\min}, \zeta\})^{d_x}/\Gamma(\frac{d_x}{2}+1)$, where $\Gamma : \mathbb{R}_+ \to \mathbb{R}_+$ denotes Euler's gamma function. Therefore, the probability of a sample $\boldsymbol{x}^{(i)} \sim \mathcal{U}(\mathcal{B}_\xi)$ being in the active region of model l can be bounded by

$$P(\omega_l(\boldsymbol{x}^{(i)}) > 0 | l \in \mathbb{A}_{t_1+j\Delta t}) \geq \min\{r_{\min}^{d_x}, \zeta^{d_x}\}/\xi^{d_x}. \tag{24}$$

The probability of none of the N_s samples falling into the active region $\mathbb{X}_l$ is consequently upper bounded by $(1 - P(\omega_l(\boldsymbol{x}^{(i)}) > 0 | l \in \mathbb{A}_{t_1+j\Delta t}))^{N_s}$, such that (15) follows from the union bound over all time steps and all models $l \in \mathbb{L}$. ■

B. Proof of Lemma 2

Proof: Satisfaction of the memory constraint (4) implies that the transmission of $\mathbb{D}_j$, $j \in \mathbb{N}$, can be achieved with time $T_{\mathrm{trans}} \leq \frac{\bar{M}}{2B} + T_d$. Hence, we have $T_{\mathrm{access}} = \frac{\Delta T}{2} \geq T_{\mathrm{trans}}$, guaranteeing satisfaction of the network constraint (5). ■

C. Proof of Theorem 2

Proof: Since the cardinality of $\mathbb{D}_{j+q}$ can be bounded by $|\mathbb{D}_{j+q}| \leq |\mathbb{D}_j| + \bar{m}$, memory constraints are satisfied as long as $|\mathbb{D}_j| \leq \bar{M}/2 - \bar{m}$. Therefore, ι as defined in (16) ensures that the memory constraint (4) is satisfied, which implies the satisfaction of the network constraint (5) due to Lemma 2. ■

D. Proof of Theorem 3

Proof: In order to prove the ultimate bound, we employ the Lyapunov function $V(\boldsymbol{e}) = \boldsymbol{e}^T\boldsymbol{P}\boldsymbol{e}$, where $\boldsymbol{P}$ is a positive definite matrix as $\boldsymbol{\lambda}$ is a Hurwitz vector. The derivative of the Lyapunov function is guaranteed to satisfy

$$\dot{V}(\boldsymbol{e}) = -\boldsymbol{e}^T\boldsymbol{Q}\boldsymbol{e} + 2\boldsymbol{e}^T\boldsymbol{p}_d(k_c)\left(f(\boldsymbol{x}) - \tilde{\mu}(\boldsymbol{x})\right). \tag{25}$$

Due to Lipschitz continuity, we obtain

$$\begin{aligned}\dot{V}(\boldsymbol{e}) \leq & -\lambda_{\min}(\boldsymbol{Q})\|\boldsymbol{e}\|^2 + 2\|\boldsymbol{e}\|\|\boldsymbol{p}_d(k_c)\|(L_f + L_{\tilde{\mu}})\|\boldsymbol{x} - \boldsymbol{x}_{\mathrm{ref}}\| \\ & + 2\|\boldsymbol{e}\|\,\|\boldsymbol{p}_d(k_c)\|\,|(f(\boldsymbol{x}_{\mathrm{ref}}) - \tilde{\mu}\,(\boldsymbol{x}_{\mathrm{ref}}))|\,,\end{aligned} \tag{26}$$

where the Lipschitz constant $L_{\tilde{\mu}}$ in (21) follows directly from Lipschitz continuity of the individual mean functions resulting from the L_k-Lipschitz kernel $k(\cdot,\cdot)$ [9]. Due to Lemma 1, the error between the unknown function $f(\cdot)$ and the LoG-GP mean $\tilde{\mu}(\cdot)$ can be bounded, such that we obtain

$$\begin{aligned}\dot{V}(\boldsymbol{e}) \leq & -\left(\lambda_{\min}(\boldsymbol{Q}) - 2\left\|\boldsymbol{p}_d(k_c)\right\|\left(L_f + L_{\tilde{\mu}}\right)\right)\|\boldsymbol{e}\|^2 \\ & + 2\|\boldsymbol{e}\|\,\|\boldsymbol{p}_d(k_c)\|\,\eta\left(\boldsymbol{x}_{\mathrm{ref}}\right).\end{aligned} \tag{27}$$

Due to (21), the Lyapunov derivative is negative for

$$\|\boldsymbol{e}\| > \frac{2\left\|\boldsymbol{p}_d(k_c)\right\|\eta\left(\boldsymbol{x}_{\mathrm{ref}}\right)}{\lambda_{\min}(\boldsymbol{Q}) - 2\left\|\boldsymbol{p}_d(k_c)\right\|\left(L_f + L_{\tilde{\mu}}\right)}. \tag{28}$$

Since the ultimately bounded set is given by the smallest sub-level set of $V(\cdot)$ which contains the ball defined through (28), we can directly determine it as $\{\boldsymbol{e} : V(\boldsymbol{e}) \leq \vartheta(\kappa_t)^2\lambda_{\min}(\boldsymbol{P})\}$ due to the quadratic structure of $V(\cdot)$. Over-approximating this set by a ball concludes the proof. ■

References

[1] A. Sahoo, S. K. Dwivedy, and P. Robi, "Advancements in the field of autonomous underwater vehicle," *Ocean Eng*, vol. 181, pp. 145–160, 2019.

[2] O. Andersson, M. Wzorek, and P. Doherty, "Deep learning quadcopter control via risk-aware active learning," in *AAAI Conf Artif Intell*, 2017, pp. 3812–3818.

[3] U. Martinez-Hernandez, B. Metcalfe, T. Assaf, L. Jabban, J. Male, and D. Zhang, "Wearable assistive robotics: A perspective on current challenges and future trends," *Sensors*, vol. 21, no. 20, pp. 1–19, 2021.

[4] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA: The MIT Press, 2006.

[5] M. Deisenroth and J. W. Ng, "Distributed Gaussian processes," in *Int Conf Mach Learn*, 2015, pp. 1481–1490.

[6] M. F. Huber, "Recursive Gaussian process: On-line regression and learning," *Pattern Recognit Lett*, vol. 45, pp. 85–91, 2014.

[7] T. Bui, C. Nguyen, and R. Turner, "Streaming sparse Gaussian process approximations," in *Adv Neural Inf Process Syst*, 2017, pp. 3300–3308.

[8] A. Gijsberts and G. Metta, "Real-time model learning using incremental sparse spectrum Gaussian process regression," *Neural Networks*, vol. 41, pp. 59–69, 2013.

[9] A. Lederer, J. Umlauft, and S. Hirche, "Uniform error bounds for Gaussian process regression with application to safe control," in *Adv Neural Inf Process Syst*, 2019, pp. 659–669.

[10] A. Lederer, A. Ordóñez Conejo, K. Maier, W. Xiao, J. Umlauft, and S. Hirche, "Gaussian process-based real-time learning for safety-critical applications," in *Int Conf Mach Learn*, 2021, pp. 6055–6064.

[11] J. Umlauft and S. Hirche, "Feedback linearization based on Gaussian processes with event-triggered online learning," *IEEE Trans Automat Contr*, vol. 65, no. 10, pp. 4154–4169, 2019.

[12] A. Capone and S. Hirche, "Backstepping for partially unknown nonlinear systems using Gaussian processes," *IEEE Control Syst Lett*, vol. 3, no. 2, pp. 416–421, 2019.

[13] A. Gahlawat, P. Zhao, A. Patterson, N. Hovakimyan, and E. A. Theodorou, "$\mathcal{L}_1$-$\mathcal{GP}$: $\mathcal{L}_1$ adaptive control with Bayesian learning," in *Learn Dyn & Cont*, 2020, pp. 1–15.

[14] H. K. Khalil, *Nonlinear Systems*, 3rd ed. Upper Saddle River, NJ: Prentice-Hall, 2002.

[15] A. Lederer, A. Capone, J. Umlauft, and S. Hirche, "How training data impacts performance in learning-based control," *IEEE Control Syst Lett*, vol. 5, no. 3, pp. 905–910, 2021.

[16] A. Lederer, J. Umlauft, and S. Hirche, "Posterior variance analysis of Gaussian processes with application to average learning curves," 2019. [Online]. Available: http://arxiv.org/abs/1906.01404

[17] J. Bezanson, A. Edelman, S. Karpinski, and V. Shah, "Julia: A fresh approach to numerical computing," *SIAM Rev*, vol. 59, no. 1, pp. 65–98, 2017.

[18] E. Trigili, S. Crea, M. Moise, A. Baldoni, M. Cempini, G. Ercolini, D. Marconi, F. Posteraro, M. Carrozza, and N. Vitiello, "Design and experimental characterization of a shoulder-elbow exoskeleton with compliant joints for post-stroke rehabilitation," *IEEE ASME Trans Mechatron*, vol. 24, no. 4, pp. 1485–1496, 2020.

[19] R. Drillis, R. Contini, and M. Bluestein, "Body segment parameters: A survey of measurement techniques," *Artif Limbs*, vol. 8, pp. 44–66, 1964.

Physically Consistent Learning of Conservative Lagrangian Systems with Gaussian Processes

Giulio Evangelisti and Sandra Hirche

Abstract— **This paper proposes a physically consistent Gaussian Process (GP) enabling the data-driven modelling of uncertain Lagrangian systems. The function space is tailored according to the energy components of the Lagrangian and the differential equation structure, analytically guaranteeing properties such as energy conservation and quadratic form. The novel formulation of Cholesky decomposed matrix kernels allow the probabilistic preservation of positive definiteness. Only differential input-to-output measurements of the function map are required while Gaussian noise is permitted in torques, velocities, and accelerations. We demonstrate the effectiveness of the approach in numerical simulation.**

I. INTRODUCTION

Euler-Lagrange (EL) systems represent a wide variety of physical systems and are the foundation of a large class of theoretical concepts in control, such as stability, passivity, and in feedback law design. The application of Gaussian Processes (GPs) in learning-based control holds great promise for the control of uncertain systems, improving important aspects like performance [1], safety [2], and data-efficiency [3]. Yet, in general GP regression does not account for physically consistency, hampering the provision of safety guarantees when used in model-based control approaches.

We address this issue by formulating GPs consistent with the relevant physical laws and mathematical models, thus implicitly performing a trade-off between the flexibility of learning and physics-imposed symmetries. Traditionally, parametric techniques [4] have been applied to the identification of EL systems, as reviewed in [5]. However, mostly assuming linearity in the parameters, these methods eventually become infeasible with increasingly nonlinear, and coupled, dynamics, as for example in soft robots [6], aircrafts [7], and marine vehicles [8].

Based on Bayesian principles, GPs allow the inclusion of analytical prior knowledge, thus forming a bridge to parametric model-based approaches. The authors of [9] propose the regression of Hamiltonian systems with GPs, preserving the symplectic structure, but without any guarantees and requiring the availability of noiseless impulse measurements. A Lagrangian kernel is derived in [10] within the framework of Reproducing Kernel Hilbert Spaces (RKHS). While analogies to our stochastic GP-based approach naturally exist, the work lacks a further specification of the associated function space, prohibiting a comprehensive analysis of control relevant properties closely intertwined with physical consistency. In [11], the EL residual dynamics are compensated by employing independent GPs per degree of freedom, neglecting correlations and structure. This leads to an implicit differential equation system, the necessity of noiseless acceleration measurements and the curse of dimensionality. Guaranteeing physical consistency of data-driven methods still represents a major open problem.

The main contribution of this work is the derivation of a physically consistent Lagrangian-Gaussian Process (L-GP) constructed by linear operators, embedding the differential equation structure into the model. Combined with quadratic kernels, we extend the GP concept from scalar function to symmetric matrix spaces. Based on a Cholesky decomposition of the underlying matrix kernel, we are able to provide a probabilistic guarantee for the positive definiteness of the estimates. Further physical consistencies of the L-GP analytically preserving equilibria, passivity, and conservatism are formally shown.

After defining the problem setting in Section II, we review the framework of GPs in Section III and describe our proposed unification with EL systems in Section IV. Section V presents our main results. Numerical illustrations validate theory and effectiveness of the approach in Section VI.

II. PROBLEM FORMULATION

In this paper, we consider the class of conservative EL systems whose equations of motion are given by[1]

$$\frac{d}{dt}\left(\nabla_{\dot{\boldsymbol{q}}} L\right) - \nabla_{\boldsymbol{q}} L = \boldsymbol{\tau} \tag{1}$$

based on the set of generalized coordinates $\boldsymbol{q} \in \mathbb{R}^N$ and the unknown Lagrangian L. The generalized forces $\boldsymbol{\tau} \in \mathbb{R}^N$ are the measurable, possibly underactuated, inputs to the system.

Assumption 1: The unknown Lagrangian is autonomous with $L \equiv L(\boldsymbol{q}, \dot{\boldsymbol{q}}) = T(\boldsymbol{q}, \dot{\boldsymbol{q}}) - V(\boldsymbol{q})$ and composed of the difference between unknown kinetic energy $T \colon \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$ and unknown potential energy $V \colon \mathbb{R}^N \to \mathbb{R}$.

Both authors are members of the Chair of Information-oriented Control, Department of Electrical and Computer Engineering, Technical University of Munich, D-80333 Munich, Germany [giulio.evangelisti, hirche]@tum.de

[1] **Notation:** Vectors $\boldsymbol{a}$ and matrices $\boldsymbol{A}$ are denoted with bold lower and upper case characters, respectively. The nabla operator ∇ multiplied with a differentiable scalar field $f(\boldsymbol{x})$ gives its gradient w.r.t. $\boldsymbol{x} \in \mathcal{X}$. The partial gradient w.r.t. a subset of variables $\boldsymbol{y} \in \mathcal{Y}$, $\mathcal{Y} \subset \mathcal{X}$, is described by $\nabla_{\boldsymbol{y}} f$. $\boldsymbol{I}$ denotes the identity, $\boldsymbol{0}$ the zero and $\boldsymbol{1}$ the ones matrix. A collection of D indexed vectors $\boldsymbol{x}_i \in \mathbb{R}^N$ is stacked row-wise in the matrix $\boldsymbol{X} = [\boldsymbol{x}_i^\top]_{i=1}^D \in \mathbb{R}^{D \times N}$ such that x_{ij} refers to the j-th component of $\boldsymbol{x}_i$. For notational convenience, we omit the indexing in the matrix expressions writing $[x_{ij}] = \boldsymbol{X} = [\boldsymbol{x}_i^\top] = [\boldsymbol{y}_j] = \boldsymbol{Y}^\top = [y_{ij}]$. $\mathrm{E}[\cdot]$, $\mathrm{Var}[\cdot]$ and $\mathrm{Cov}[\cdot]$ denote the expectation, variance and covariance operators w.r.t. random variables. The Hadamard product and square are denoted by $\boldsymbol{x} \circ \boldsymbol{y}$ and $\boldsymbol{x}^{\circ 2} = \boldsymbol{x} \circ \boldsymbol{x}$, respectively, the Kronecker product with $\otimes$, and the direct sum with $\oplus$. With $\lambda_k(\cdot)$ we denote the k-th largest eigenvalue corresponding to the components of the decreasingly ordered eigenvalue vector $\boldsymbol{\lambda}^{\downarrow}(\cdot)$.

Assumption 2: The kinetic energy is of the form $T(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \frac{1}{2}\dot{\boldsymbol{q}}^T \boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}}$ where $\boldsymbol{M}\colon \mathbb{R}^N \to \mathbb{R}^{N\times N}$ is the unknown, (symmetric) positive definite, inertia matrix.

Assumption 3: The potential energy is decomposable into the sum $V(\boldsymbol{q}) = G(\boldsymbol{q}) + U(\boldsymbol{q})$ of gravitational and elastic potential energies G and U, respectively, where the former has an equilibrium, $G(\boldsymbol{0}) = 0$, $\nabla_{\boldsymbol{q}} G(\boldsymbol{0}) = \boldsymbol{0}$, and the latter is given by $U(\boldsymbol{q}) = \frac{1}{2}\boldsymbol{q}^T \boldsymbol{S}(\boldsymbol{q})\boldsymbol{q}$ with unknown, (symmetric) positive definite, stiffness matrix $\boldsymbol{S}\colon \mathbb{R}^N \to \mathbb{R}^{N\times N}$.

Assumptions 1–3 essentially describe the considered system class and represent its physical properties. Assumption 1 restricts the systems described by (1) to be only implicitly dependent on time and free from disturbances caused by other external forces such as dissipation. Extending to handle dissipative and time-variant systems is possible and topic of current research. Assumption 2 is a valid approach when dealing, e.g., with classical mechanical rigid body systems, while Assumption 3 constrains chosen coordinates to have an equilibrium at the origin.

Exploiting Assumptions 1–3 and applying the chain rule to (1), we obtain the well-known matrix-vector expression

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}\,, \tag{2}$$

introducing the (non-unique) generalized Coriolis matrix $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \coloneqq \frac{1}{2}[\nabla_{\dot{\boldsymbol{q}}}\nabla_{\boldsymbol{q}}^\top T + \dot{\boldsymbol{M}}(\boldsymbol{q}) - \nabla_{\boldsymbol{q}}\nabla_{\dot{\boldsymbol{q}}}^\top T]$ as in [12] and the vector of generalized potential forces $\boldsymbol{g}(\boldsymbol{q}) \coloneqq \nabla_{\boldsymbol{q}} V$, having exploited $\nabla_{\dot{\boldsymbol{q}}}\nabla_{\dot{\boldsymbol{q}}}^\top T = \boldsymbol{M}(\boldsymbol{q})$, $\nabla_{\boldsymbol{q}} T = \frac{1}{2}(\nabla_{\boldsymbol{q}}\nabla_{\dot{\boldsymbol{q}}}^\top T)\dot{\boldsymbol{q}}$ as well as $\left(\nabla_{\dot{\boldsymbol{q}}}\nabla_{\boldsymbol{q}}^\top T\right)\dot{\boldsymbol{q}} = \dot{\boldsymbol{M}}(\boldsymbol{q})\dot{\boldsymbol{q}}$. Note that there are other ways to define the matrix $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$. This particular choice, however, has the unique property that $\dot{\boldsymbol{M}} - 2\boldsymbol{C}$ is skew-symmetric.

Our objective is to approximate the unknown Lagrangian function $L(\boldsymbol{q}, \dot{\boldsymbol{q}})$ by a data-based estimate $\hat{L}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ which is physically consistent, i.e., maximally aligned with the existing physical knowledge, in our case Assumptions 1–3. We assume access to noisefree positional observations $\boldsymbol{q}_i$, $i = 1, \dots, D$ with $D \in \mathbb{N}$, and to potentially noisy velocity, acceleration and torque measurements

$$\dot{\boldsymbol{x}}_i = \begin{bmatrix} \dot{\boldsymbol{q}}_i \\ \ddot{\boldsymbol{q}}_i \end{bmatrix} + \begin{bmatrix} \boldsymbol{\omega}_i \\ \boldsymbol{\alpha}_i \end{bmatrix}, \quad \begin{bmatrix} \boldsymbol{\omega}_i \\ \boldsymbol{\alpha}_i \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{\Sigma}_{\boldsymbol{\omega}_i} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Sigma}_{\boldsymbol{\alpha}_i} \end{bmatrix}\right), \tag{3a}$$

$$\boldsymbol{y}_i = \boldsymbol{\tau}_i(\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i, \ddot{\boldsymbol{q}}_i) + \boldsymbol{\varepsilon}_i\,, \quad \boldsymbol{\varepsilon}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}_i})\,. \tag{3b}$$

The noise processes $\{\boldsymbol{\omega}_i\}$, $\{\boldsymbol{\alpha}_i\}$ and $\{\boldsymbol{\varepsilon}_i\}$ are white, zero-mean, uncorrelated, and have known covariance matrices $\boldsymbol{\Sigma}_{\boldsymbol{\omega}_i}$, $\boldsymbol{\Sigma}_{\boldsymbol{\alpha}_i}$ and $\boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}_i}$, respectively. After collecting all D observations at positions $\boldsymbol{Q} = [\boldsymbol{q}_i^\top]$ with analogous matrices $\dot{\boldsymbol{X}} = [\dot{\boldsymbol{x}}_i^\top]$ and $\boldsymbol{Y} = [\boldsymbol{y}_i^\top]$, we obtain the training data set

$$\mathcal{D} = \{\boldsymbol{Q}, \dot{\boldsymbol{X}}, \boldsymbol{Y}\}\,.$$

Remark 1: Extending to noisy positional inputs is possible via certain GP-based methods as in, e.g., [13], [14], yet they are mostly based on first-order Taylor approximations.

III. GAUSSIAN PROCESS FRAMEWORK

This section gives a brief overview of the mathematical framework of GPs based on [15]. For a more complete introduction, the reader is referred to the literature [16]–[18].

A. Inference with GPs

For $\boldsymbol{x}, \boldsymbol{x}' \in \boldsymbol{\mathcal{X}}$ in a continuous domain $\boldsymbol{\mathcal{X}} \subseteq \mathbb{R}^{2N}$, a GP with mean $m(\boldsymbol{x})$ and covariance $k(\boldsymbol{x}, \boldsymbol{x}')$, denoted by

$$f(\boldsymbol{x}) \sim \mathcal{GP}\left(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')\right) \tag{4}$$

and with scalar $f(\boldsymbol{x}) \in \mathbb{R}$, is a stochastic process extending the Gaussian probability distribution from random variables to functions. As such, it inherits the convenient mathematical properties of the Normal distribution. In particular, any finite marginalization $\boldsymbol{z} = [z_i]$ of function observations $z_i = f(\boldsymbol{x_i}) + \zeta_i$, corrupted by the zero-mean white-noise process $\zeta_i \in \mathbb{R}$ and corresponding to the finite subset of function evaluations at $\{\boldsymbol{x}_i\} \subset \boldsymbol{\mathcal{X}}$, is jointly Gaussian distributed.

A GP is fully characterized by its mean and kernel function $m(\boldsymbol{x}) = \mathrm{E}[f(\boldsymbol{x})]$ and $k(\boldsymbol{x}, \boldsymbol{x}') = \mathrm{Cov}[f(\boldsymbol{x}), f(\boldsymbol{x}')]$, respectively. The former allows the analytical inclusion of prior knowledge about the unknown function, whereas the latter determines higher level functional properties such as smoothness. Estimation of the covariance describing hyperparameters is mostly done via optimization of the marginal likelihood to maximize the probability of observing the measured outputs.

For regression, GPs exploit the joint Gaussian distribution of observations $\boldsymbol{z}$ at measurement locations $\boldsymbol{X} = [\boldsymbol{x}_i^\top]$ and prediction $f(\boldsymbol{x})$ w.r.t. the test input $\boldsymbol{x} \in \boldsymbol{\mathcal{X}}$ given by

$$\begin{bmatrix} f(\boldsymbol{x}) \\ \boldsymbol{z} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(\boldsymbol{x}) \\ \boldsymbol{m} \end{bmatrix}, \begin{bmatrix} k(\boldsymbol{x}, \boldsymbol{x}) & \boldsymbol{k}^\top(\boldsymbol{x}) \\ \boldsymbol{k}(\boldsymbol{x}) & \boldsymbol{K} + \boldsymbol{\Sigma} \end{bmatrix}\right), \tag{5}$$

where we have introduced the mean $\boldsymbol{m} = [m(\boldsymbol{x}_i)]$ and covariance $\boldsymbol{k}(\boldsymbol{x}) = [k(\boldsymbol{x}_i, \boldsymbol{x})]$ vectors, the Gram matrix $\boldsymbol{K} = [k(\boldsymbol{x}_i, \boldsymbol{x}_j)]$ as well as the noise covariance matrix $\boldsymbol{\Sigma} = \mathrm{Var}[\boldsymbol{\zeta}]$ with $\boldsymbol{\zeta} = [\zeta_i]$. Here, we have dropped the dependencies of $\boldsymbol{m}(\boldsymbol{X})$, $\boldsymbol{k}(\boldsymbol{X}, \boldsymbol{x})$ and $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X})$ on $\boldsymbol{X}$ for notational simplicity. Conditioning on the observations yields a conditional Gaussian distribution with posterior mean $\hat{f}(\boldsymbol{x}) \equiv \mathrm{E}[f(\boldsymbol{x})|\boldsymbol{z}]$ given analytically by

$$\hat{f}(\boldsymbol{x}) = m(\boldsymbol{x}) + \boldsymbol{k}^\top(\boldsymbol{x})\left(\boldsymbol{K} + \boldsymbol{\Sigma}\right)^{-1}(\boldsymbol{z} - \boldsymbol{m})\,. \tag{6}$$

B. Linear Operators and GPs

Derivatives and integrals of GPs remain GPs, since these operations are linear. Applying a linear transformation operator $\boldsymbol{\mathcal{T}}_{\boldsymbol{x}}$ to the GP of (4) leads to a new, possibly multidimensional, GP in the form of

$$\boldsymbol{\mathcal{T}}_{\boldsymbol{x}} f(\boldsymbol{x}) \sim \boldsymbol{\mathcal{GP}}\left(\boldsymbol{m}_{\boldsymbol{\mathcal{T}}}(\boldsymbol{x}), \boldsymbol{K}_{\boldsymbol{\mathcal{T}}}(\boldsymbol{x}, \boldsymbol{x}')\right) \tag{7}$$

with transformed mean $\boldsymbol{m}_{\boldsymbol{\mathcal{T}}}(\boldsymbol{x}) = \boldsymbol{\mathcal{T}}_{\boldsymbol{x}} m(\boldsymbol{x})$ and kernel

$$\boldsymbol{K}_{\boldsymbol{\mathcal{T}}}(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{\mathcal{T}}_{\boldsymbol{x}} k(\boldsymbol{x}, \boldsymbol{x}') \boldsymbol{\mathcal{T}}_{\boldsymbol{x}'}^\top\,. \tag{8}$$

Here, $\boldsymbol{\mathcal{T}}_{\boldsymbol{x}'}^\top$ is applied from the right [9] as is necessary when considering matrix operators.

IV. REGRESSION OF LAGRANGIAN SYSTEMS

Our key idea is to consistently model the unknown Lagrangian using a GP. We propose a structural approach for the application of GP regression (4)–(6) to EL systems (1)–(2): deriving specific GPs and kernels, we ensure consistency with physical requirements, as in Assumptions 1–3, and embed the structure of the differential equations using Nabla operators ∇, thereby inducing multidimensional L-GPs according to (7)–(8) based on the linearity of differentiation.

A. Energy Structuring

Firstly, we model the Lagrangian as a composite GP $L(\boldsymbol{q}, \dot{\boldsymbol{q}}) = T(\boldsymbol{q}, \dot{\boldsymbol{q}}) - V(\boldsymbol{q})$ writing

$$L(\boldsymbol{q}, \dot{\boldsymbol{q}}) \sim \mathcal{GP}\left(m_L(\boldsymbol{q}, \dot{\boldsymbol{q}}), k_L(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{q}', \dot{\boldsymbol{q}}')\right) \tag{9}$$

with the subjacent kinetic and potential energy GPs

$$T(\boldsymbol{q}, \dot{\boldsymbol{q}}) \sim \mathcal{GP}\left(m_T(\boldsymbol{q}, \dot{\boldsymbol{q}}), k_T(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{q}', \dot{\boldsymbol{q}}')\right) , \tag{10a}$$

$$V(\boldsymbol{q}) \sim \mathcal{GP}\left(m_V(\boldsymbol{q}), k_V(\boldsymbol{q}, \boldsymbol{q}')\right) . \tag{10b}$$

Here, the means $m_L = m_T - m_V$ and covariances $k_L = k_T + k_V$ specify the L-GP (9) in a structurally consistent manner. In particular, the potential energy GP (10b) depends only on the position $\boldsymbol{q}$ by construction. It is split up further into the sum $V(\boldsymbol{q}) = G(\boldsymbol{q}) + U(\boldsymbol{q})$ of independent gravitational and elastic potential GPs

$$G(\boldsymbol{q}) \sim \mathcal{GP}\left(m_G(\boldsymbol{q}), k_G(\boldsymbol{q}, \boldsymbol{q}')\right) , \tag{11a}$$

$$U(\boldsymbol{q}) \sim \mathcal{GP}\left(m_U(\boldsymbol{q}), k_U(\boldsymbol{q}, \boldsymbol{q}')\right) . \tag{11b}$$

B. Quadratic and Positive Definite Structures

Consistency of kinetic and elastic GPs is ensured by according design of their mean and covariance functions. If available, we impose on the according priors the same quadratic structure based on, e.g., for the kinetic energy, the (symmetric) positive definite a-priori matrix $\boldsymbol{M}_0(\boldsymbol{q}) \succ \boldsymbol{0}$, $\forall \boldsymbol{q} \in \mathbb{R}^N$, as described in the following.

Assumption 4: The priors $m_{f_k} \in \{m_U, m_T\}$ are in the same form as the unknown energy functions

$$f_k(\boldsymbol{q}) := \frac{1}{2} \overset{(k)}{\boldsymbol{q}}{}^\top \boldsymbol{H}_k(\boldsymbol{q}) \overset{(k)}{\boldsymbol{q}} \tag{12}$$

for $k \in \{0, 1\}$, where $m_{f_0} \equiv m_U$ with $\boldsymbol{H}_{00} \equiv \boldsymbol{S}_0$ and $m_{f_1} \equiv m_T$ with $\boldsymbol{H}_{10} \equiv \boldsymbol{M}_0$. In addition, the gravitational prior fulfills the equilibrium condition such that $m_G(\boldsymbol{0}) = 0$ and $\nabla_{\boldsymbol{q}} m_G(\boldsymbol{0}) = \boldsymbol{0}$ hold.

Having specified the means in (10a) and (11b), we now move on to the kernels. Consider the quadratic functional

$$\kappa_k(\boldsymbol{q}, \boldsymbol{q}') = \frac{1}{4} \Big(\overset{(k)}{\boldsymbol{q}} \circ \overset{(k)}{\boldsymbol{q}}{}' \Big)^\top \boldsymbol{\Theta}_k(\boldsymbol{q}, \boldsymbol{q}') \Big(\overset{(k)}{\boldsymbol{q}} \circ \overset{(k)}{\boldsymbol{q}}{}' \Big) \tag{13}$$

with differential-operational index $k \in \{0, 1\}$ and the (symmetric) positive definite Cholesky decomposed matrix kernel

$$\boldsymbol{\Theta}_k(\boldsymbol{q}, \boldsymbol{q}') = \boldsymbol{R}_k^\top(\boldsymbol{q}, \boldsymbol{q}') \boldsymbol{R}_k(\boldsymbol{q}, \boldsymbol{q}') . \tag{14}$$

The Cholesky factor $\boldsymbol{R}_k$ is an upper-right triangular matrix

$$\boldsymbol{R}_k(\boldsymbol{q}, \boldsymbol{q}') = \begin{bmatrix} r_{k11}(\boldsymbol{q}, \boldsymbol{q}') & \dots & r_{k1N}(\boldsymbol{q}, \boldsymbol{q}') \\ & \ddots & \vdots \\ \boldsymbol{0} & & r_{kNN}(\boldsymbol{q}, \boldsymbol{q}') \end{bmatrix} \tag{15}$$

consisting for $0 \le n \le m \le N$ of kernels $r_{knm}(\boldsymbol{q}, \boldsymbol{q}')$. We now use (13) to set the covariances in (10a) and (11b) to:

$$k_U \equiv k_{f_0} := \kappa_0 , \tag{16a}$$

$$k_T \equiv k_{f_1} := \kappa_1 . \tag{16b}$$

Thus, we implicitly model the stiffness and inertia matrices denoted in accordance with (12) by $\boldsymbol{H}_k := [h_{knm}]$ as symmetric, matrix-valued, GPs with entries

$$h_{knm}(\boldsymbol{q}) \sim \mathcal{GP}\left(\eta_{knm}(\boldsymbol{q}), \theta_{knm}(\boldsymbol{q}, \boldsymbol{q}')\right) ,$$

where $h_{knm} = h_{kmn}$, and analagously constructing $\boldsymbol{H}_{k0} := [\eta_{knm}]$. The kernels $\theta_{knm} = \boldsymbol{\rho}_{kn}^\top \boldsymbol{\rho}_{km}$ stem from the inner product of the non-zero columns from (15), i.e, $\boldsymbol{\rho}_{km} := [r_{klm}] \in \mathbb{R}^P$ for $P = \min\{n, m\}$. Note that, due to the nonlinearity of multiplication, the r_{klm} may not be associated any further with underlying GPs, or more specifically, Gaussian Random Variables (GRVs). However, the covariances θ_{knm} remain valid kernel functions according to [15].

C. Differential Equation Embedding

Let us now use the chain rule to expand $\boldsymbol{\mathcal{L}}_{\boldsymbol{q}} \equiv \frac{d}{dt} \nabla_{\dot{\boldsymbol{q}}} - \nabla_{\boldsymbol{q}}$ from (1) by writing

$$\boldsymbol{\mathcal{L}}_{\boldsymbol{q}} = \left(\nabla_{\dot{\boldsymbol{q}}}^\top \ddot{\boldsymbol{q}} + \nabla_{\boldsymbol{q}}^\top \dot{\boldsymbol{q}} \right) \nabla_{\dot{\boldsymbol{q}}} - \nabla_{\boldsymbol{q}} := \boldsymbol{\mathcal{L}}_{\boldsymbol{q}, \dot{\boldsymbol{q}}}(\dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) . \tag{17}$$

Applying the Lagrangian-differential vector operator (17) to the GP from (9), we obtain for the torques (1)–(2) a vector-valued GP $\boldsymbol{\tau} \equiv \boldsymbol{\tau}(\boldsymbol{q}, \boldsymbol{q}'; \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}', \ddot{\boldsymbol{q}}') \in \mathbb{R}^N$ denoted by

$$\boldsymbol{\tau} \sim \boldsymbol{\mathcal{GP}}\left(\boldsymbol{m}_{\boldsymbol{\tau}}(\boldsymbol{q}; \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}), \boldsymbol{K}_{\boldsymbol{\tau}}(\boldsymbol{q}, \boldsymbol{q}'; \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}', \ddot{\boldsymbol{q}}') \right) \tag{18}$$

with mean vector $\boldsymbol{m}_{\boldsymbol{\tau}} = \boldsymbol{\mathcal{L}}_{\boldsymbol{q}} m_L$ and kernel matrix $\boldsymbol{K}_{\boldsymbol{\tau}} = \boldsymbol{\mathcal{L}}_{\boldsymbol{q}} \boldsymbol{\mathcal{L}}_{\boldsymbol{q}'}^\top k_L$. As indicated in (18) via the semicolon, $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$ have the role of regressors, whereas $\boldsymbol{q}$ remains a conventional input.

D. Joint GP Distribution

Finally, we assert a joint GP for the energies (10a)–(10b) and torques (18). Including for $\boldsymbol{\gamma}^\top(\boldsymbol{q}) := [V(\boldsymbol{q}), \nabla_{\boldsymbol{q}}^\top V(\boldsymbol{q})]$ the equilibrium condition

$$\boldsymbol{\gamma}_{\boldsymbol{0}} := \boldsymbol{\gamma}(\boldsymbol{0}) = \boldsymbol{0}$$

in alignment with Assumption 3, we write

$$\begin{bmatrix} T \\ V \\ \boldsymbol{\gamma}_{\boldsymbol{0}} \\ \boldsymbol{y} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m_T \\ m_V \\ \boldsymbol{0} \\ \boldsymbol{m}_{\boldsymbol{y}} \end{bmatrix}, \begin{bmatrix} \sigma_T^2 & 0 & \boldsymbol{0}^\top & \boldsymbol{k}_{\boldsymbol{y}T}^\top \\ 0 & \sigma_V^2 & \boldsymbol{k}_{\boldsymbol{0}V}^\top & \boldsymbol{k}_{\boldsymbol{y}V}^\top \\ \boldsymbol{0} & \boldsymbol{k}_{\boldsymbol{0}V} & \boldsymbol{K}_{\boldsymbol{0}} & \boldsymbol{K}_{\boldsymbol{y}\boldsymbol{0}}^\top \\ \boldsymbol{k}_{\boldsymbol{y}T} & \boldsymbol{k}_{\boldsymbol{y}V} & \boldsymbol{K}_{\boldsymbol{y}\boldsymbol{0}} & \boldsymbol{K}_{\boldsymbol{y}} \end{bmatrix} \right) \tag{19}$$

dropping dependencies on all variables for notational brevity. Also, we have introduced: the stacked vector of outputs $\boldsymbol{y} = \operatorname{vec}(\boldsymbol{Y}^\top)$ with mean $\boldsymbol{m}_{\boldsymbol{y}} = [\boldsymbol{m}_{\boldsymbol{\tau}}(\boldsymbol{q}_i; \dot{\boldsymbol{q}}_i, \ddot{\boldsymbol{q}}_i)]$ and covariance

$$\boldsymbol{K}_{\boldsymbol{y}} = \left[\boldsymbol{K}_{\boldsymbol{\tau}}(\boldsymbol{q}_i, \boldsymbol{q}_j; \dot{\boldsymbol{q}}_i, \ddot{\boldsymbol{q}}_i, \dot{\boldsymbol{q}}_j, \ddot{\boldsymbol{q}}_j) \right] + \oplus_i \boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}_i} , \tag{20}$$

the energy variances $\sigma_T^2(\boldsymbol{q}, \dot{\boldsymbol{q}}) = k_T(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{q}, \dot{\boldsymbol{q}})$ and $\sigma_V^2(\boldsymbol{q}) = k_V(\boldsymbol{q}, \boldsymbol{q})$, the equilibrium variance

$$\boldsymbol{K}_{\boldsymbol{0}} = \begin{bmatrix} k_V(\boldsymbol{0}, \boldsymbol{0}) & \nabla_{\boldsymbol{q}'}^\top k_V(\boldsymbol{0}, \boldsymbol{0}) \\ \nabla_{\boldsymbol{q}} k_V(\boldsymbol{0}, \boldsymbol{0}) & \nabla_{\boldsymbol{q}} \nabla_{\boldsymbol{q}'}^\top k_V(\boldsymbol{0}, \boldsymbol{0}) \end{bmatrix}$$

and covariance $\boldsymbol{K}_{\boldsymbol{y}\boldsymbol{0}} = [\boldsymbol{k}_{\boldsymbol{y}V}(\boldsymbol{0}), \nabla_{\boldsymbol{q}}^{\top}\boldsymbol{k}_{\boldsymbol{y}V}(\boldsymbol{0})]$, as well as the remaining equilibrium-potential $\boldsymbol{k}_{\boldsymbol{0}V}(\boldsymbol{q})$ and Lagrangian-differential energy $\boldsymbol{k}_{\boldsymbol{y}T}(\boldsymbol{q},\dot{\boldsymbol{q}})$, $\boldsymbol{k}_{\boldsymbol{y}V}(\boldsymbol{q})$ covariances

$$\boldsymbol{k}_{\boldsymbol{0}V}(\boldsymbol{q}) = \begin{bmatrix} k_V(\boldsymbol{0},\boldsymbol{q}) \\ \nabla_{\boldsymbol{q}_i} k_V(\boldsymbol{0},\boldsymbol{q}) \end{bmatrix} , \tag{21a}$$

$$\boldsymbol{k}_{\boldsymbol{y}T}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \left[\boldsymbol{\mathcal{L}}_{\boldsymbol{q}_i,\dot{\boldsymbol{q}}_i}(\dot{\boldsymbol{q}}_i,\ddot{\boldsymbol{q}}_i) k_T(\boldsymbol{q}_i,\dot{\boldsymbol{q}}_i,\boldsymbol{q},\dot{\boldsymbol{q}})\right] , \tag{21b}$$

$$\boldsymbol{k}_{\boldsymbol{y}V}(\boldsymbol{q}) = \left[\nabla_{\boldsymbol{q}_i} k_V(\boldsymbol{q}_i,\boldsymbol{q})\right] . \tag{21c}$$

The fully formulated L-GP (19) is the first part of our contribution. Before moving on to analyze its physical consistency, we propose a method for partial input noise compensation.

E. Nonlinear Noise Compensation

The regressor structure stemming from the application of (17) to (16b) allows for the compensation of noise in the differential inputs $\dot{\boldsymbol{q}}_i$ and $\ddot{\boldsymbol{q}}_i$. Therefore, as a last step, we combine (2) and (18) in order to transform (3b) into

$$\boldsymbol{y}_i = \boldsymbol{\tau}(\boldsymbol{q}_i;\dot{\boldsymbol{x}}_i) + \tilde{\boldsymbol{\varepsilon}}_i$$

with the transformed composite noise variable

$$\tilde{\boldsymbol{\varepsilon}}_i = -(\nabla_{\dot{\boldsymbol{q}}_i}\nabla_{\dot{\boldsymbol{q}}_i}^{\top}T)\boldsymbol{\alpha}_i - (\nabla_{\dot{\boldsymbol{q}}_i}\nabla_{\boldsymbol{q}_i}^{\top}T)\boldsymbol{\omega}_i + \boldsymbol{\varepsilon}_i \,.$$

Approximating the product of two GRVs to remain Gaussian, we obtain a heteroscedastic (state-dependent) noise model

$$\tilde{\boldsymbol{\varepsilon}}_i \sim \mathcal{N}\left(\boldsymbol{0}, \boldsymbol{\Sigma}_{\tilde{\boldsymbol{\varepsilon}}_i}(\boldsymbol{q}_i,\dot{\boldsymbol{q}}_i)\right) , \tag{22}$$

where $\boldsymbol{\Sigma}_{\tilde{\boldsymbol{\varepsilon}}_i} = \boldsymbol{\Sigma}_{\tilde{\boldsymbol{\alpha}}_i}(\boldsymbol{q}_i) + \boldsymbol{\Sigma}_{\tilde{\boldsymbol{\omega}}_i}(\boldsymbol{q}_i,\dot{\boldsymbol{q}}_i) + \boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}_i}$. After extensive computations, it can be verified that the transformed $\boldsymbol{\Sigma}_{\tilde{\boldsymbol{\alpha}}_i} := \mathrm{Var}[(\nabla_{\dot{\boldsymbol{q}}_i}\nabla_{\dot{\boldsymbol{q}}_i}^{\top}T)\boldsymbol{\alpha}_i]$ and $\boldsymbol{\Sigma}_{\tilde{\boldsymbol{\omega}}_i} := \mathrm{Var}[(\nabla_{\dot{\boldsymbol{q}}_i}\nabla_{\boldsymbol{q}_i}^{\top}T)\boldsymbol{\omega}_i]$, exploiting stochastic independencies, are given by

$$\begin{aligned} \boldsymbol{\Sigma}_{\tilde{\boldsymbol{\alpha}}_i} &= \boldsymbol{M}_0\boldsymbol{\Sigma}_{\boldsymbol{\alpha}_i}\boldsymbol{M}_0 + (\boldsymbol{1}-\boldsymbol{I})\circ\boldsymbol{\Sigma}_{\boldsymbol{\alpha}_i}\circ\boldsymbol{\Theta}_{1ii} + \mathrm{diag}(\boldsymbol{\Theta}_{1ii}\boldsymbol{\sigma}_{\boldsymbol{\alpha}_i}) \\ \boldsymbol{\Sigma}_{\tilde{\boldsymbol{\omega}}_i} &= \boldsymbol{C}_0\boldsymbol{\Sigma}_{\boldsymbol{\omega}_i}\boldsymbol{C}_0^{\top} + \mathrm{diag}\left(\boldsymbol{\Gamma}(\boldsymbol{q}_i,\dot{\boldsymbol{q}}_i)\boldsymbol{\sigma}_{\boldsymbol{\omega}_i}\right) \end{aligned} \tag{23}$$

with the inertial variance $\boldsymbol{\Theta}_{1ii} := \boldsymbol{\Theta}_1(\boldsymbol{q}_i,\boldsymbol{q}_i)$, the Coriolis mean $\boldsymbol{C}_0(\boldsymbol{q}_i,\dot{\boldsymbol{q}}_i) = [\boldsymbol{M}_0(\boldsymbol{q}_i)\dot{\boldsymbol{q}}_i]\nabla_{\boldsymbol{q}_i}^{\top}$ and variance $\boldsymbol{\Gamma}(\boldsymbol{q}_i,\dot{\boldsymbol{q}}_i) = [\boldsymbol{\Theta}_1(\boldsymbol{q}_i,\boldsymbol{q}_i)\dot{\boldsymbol{q}}_i^{\circ 2}](\nabla_{\boldsymbol{q}_i}^{\circ 2})^{\top}$ matrices as well as the main diagonals vectors $\boldsymbol{\sigma}_{\boldsymbol{\alpha}_i}$ and $\boldsymbol{\sigma}_{\boldsymbol{\omega}_i}$ of $\boldsymbol{\Sigma}_{\boldsymbol{\alpha}_i}$ and $\boldsymbol{\Sigma}_{\boldsymbol{\omega}_i}$, respectively.

Proposition 1: The distribution (22) with state-dependent variances (23) is a stochastically consistent approximation which becomes exact for the limit case $\boldsymbol{\Sigma}_{\boldsymbol{\alpha}_i}, \boldsymbol{\Sigma}_{\boldsymbol{\omega}_i} \to \boldsymbol{0}$, or the case of certainly known $T = m_T$ with $k_T \equiv 0$.

Proof: The limit case $\boldsymbol{\Sigma}_{\boldsymbol{\alpha}_i}, \boldsymbol{\Sigma}_{\boldsymbol{\omega}_i} \to \boldsymbol{0}$ leads to noisefree inputs with $\tilde{\boldsymbol{\varepsilon}}_i = \boldsymbol{\varepsilon}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}_i})$. The case of known T with certainty $k_T \equiv 0$ leads to $\boldsymbol{M}_0 \equiv \boldsymbol{M}$ and $\boldsymbol{\Sigma}_{\tilde{\boldsymbol{\alpha}}_i} = \boldsymbol{M}_0\boldsymbol{\Sigma}_{\boldsymbol{\alpha}_i}\boldsymbol{M}_0^{\top}$, respectively. Since the transformation $\tilde{\boldsymbol{\alpha}}_i = -\boldsymbol{M}\boldsymbol{\alpha}$ is linear, $\tilde{\boldsymbol{\alpha}}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_{\tilde{\boldsymbol{\alpha}}_i})$ holds exactly. Proceeding analgously for $\tilde{\boldsymbol{\omega}}_i = -(\nabla_{\dot{\boldsymbol{q}}_i}\nabla_{\boldsymbol{q}_i}^{\top}T)\boldsymbol{\omega}$, we arrive at (22). ■

By casting from the temporal into the spatial domain, (22) unifies the L-GP framework with techniques from nonlinear Kalman filtering [19]. For effective noise compensation in the inputs $\dot{\boldsymbol{q}}_i$ and $\ddot{\boldsymbol{q}}_i$, we replace the block matrices $\boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}_i}$ in (20) by $\boldsymbol{\Sigma}_{\tilde{\boldsymbol{\varepsilon}}_i}(\boldsymbol{q}_i,\boldsymbol{\xi}_i)$ from (22), where $\boldsymbol{\xi}_i := [\dot{x}_{in}]$. Also, we compute the torque covariance matrix (20) and the Lagrangian-differential kinetic covariance vector (21b) by setting $\dot{\boldsymbol{x}}_i^{\top} = [\dot{\boldsymbol{q}}_i^{\top}, \ddot{\boldsymbol{q}}_i^{\top}]$, $\forall i,j \in \{1,\ldots,D\}$.

V. Invariance Properties

Having introduced the novel framework of L-GPs, we now investigate their properties and derive certain guarantees.

A. Quadratic Form

Employing the kernel structure (13), quadratic form of the energy GPs (10a) and (11b) can be guaranteed analytically. In order to specify our result in Lemma 1, we require the following definition: for $\delta\boldsymbol{x}_i \in \mathbb{R}^N$ and fixed $\boldsymbol{q}$, the directional derivative of $\boldsymbol{\Theta}_k(\boldsymbol{q}_i,\boldsymbol{q})$ along $\delta\boldsymbol{x}_i$ at $\boldsymbol{q}_i$ gives

$$\boldsymbol{\Phi}_k(\boldsymbol{q}_i,\delta\boldsymbol{x}_i,\boldsymbol{q}) := \left[\delta\boldsymbol{x}_i^{\top}\nabla_{\boldsymbol{q}_i}\theta_{knm}(\boldsymbol{q}_i,\boldsymbol{q})\right] \tag{24}$$

The matrix $\boldsymbol{\Phi}_k$ is the directional matrix-derivative of $\boldsymbol{\Theta}_k$ from (14) moving through position $\boldsymbol{q}_i$ with velocity $\delta\boldsymbol{x}_i$. In particular, note that

$$\boldsymbol{\Phi}_k(\boldsymbol{q}_i(\tau),\dot{\boldsymbol{q}}_i(\tau),\boldsymbol{q}) = \dot{\boldsymbol{\Theta}}_k(\boldsymbol{q}_i(\tau),\boldsymbol{q})$$

holds for fixed $\boldsymbol{q}$ by deriving w.r.t. the virtual time τ.

Lemma 1: The mean estimates $\hat{f}_k(\boldsymbol{q}) \equiv \mathrm{E}[f_k(\boldsymbol{q})|\boldsymbol{y},\gamma_{\boldsymbol{0}}]$ of kinetic and elastic GPs (10a) and (11b) in the joint model (19) are analytically guaranteed to be of quadratic form:

$$\hat{f}_k(\boldsymbol{q}) = \frac{1}{2}\overset{(k)}{\boldsymbol{q}}{}^{\top}\hat{\boldsymbol{H}}_k(\boldsymbol{q})\overset{(k)}{\boldsymbol{q}} \,. \tag{25}$$

The posterior matrix estimates $\hat{\boldsymbol{H}}_k$ are decomposed as

$$\hat{\boldsymbol{H}}_k(\boldsymbol{q}) = \boldsymbol{H}_{k0}(\boldsymbol{q}) + \frac{1}{2}\sum_{i=1}^{D}\boldsymbol{N}_{ki}(\boldsymbol{q}) + \boldsymbol{N}_{ki}^{\top}(\boldsymbol{q}) \,, \tag{26}$$

where the basis matrices $\boldsymbol{N}_{ki}$ are given by

$$\begin{aligned} \boldsymbol{N}_{ki}(\boldsymbol{q}) = &\frac{\partial^k}{\partial\tau^k}\left[\delta\boldsymbol{x}_i\overset{(k)}{\boldsymbol{q}}{}_i^{\top}(\tau)\circ\boldsymbol{\Theta}_k(\boldsymbol{q}_i(\tau),\boldsymbol{q})\right] \\ &+ \frac{1}{2}(-1)^k\overset{(k)}{\boldsymbol{q}}_i\overset{(k)}{\boldsymbol{q}}{}_i^{\top}\circ\boldsymbol{\Phi}_k(\boldsymbol{q}_i,\delta\boldsymbol{x}_i,\boldsymbol{q}) \end{aligned} \tag{27}$$

with the subvectors $\delta\boldsymbol{x}_i = [\Delta x_n]_{n=1+(i-1)N}^{iN}$ of the transformed innovation $\Delta\boldsymbol{x} = \boldsymbol{K}_D^{-1}\Delta\boldsymbol{y}$, the Kalman-like kernel gain matrix $\boldsymbol{K}_D = \boldsymbol{K}_{\boldsymbol{y}} - \boldsymbol{K}_{\boldsymbol{y}\boldsymbol{0}}\boldsymbol{K}_{\boldsymbol{0}}^{-1}\boldsymbol{K}_{\boldsymbol{y}\boldsymbol{0}}^{\top}$, and the innovation difference $\Delta\boldsymbol{y} = \boldsymbol{y} - \boldsymbol{m}_{\boldsymbol{y}}$.

Proof: We set $G \equiv 0$ w.l.o.g., leading to $V = U \equiv f_0$ and $\boldsymbol{k}_{\boldsymbol{0}V} = \boldsymbol{0}$ due to (16a) and (13). Marginalizing (19) over the complementary $f_{\bar{k}}$ for $\{\bar{k}\} = \{0,1\} \setminus \{k\}$ and conditioning on $\boldsymbol{y}$ and $\gamma_{\boldsymbol{0}}$ yields another Gaussian distribution with mean

$$\hat{f}_k(\boldsymbol{q}) = m_{f_k}(\boldsymbol{q}) + \begin{bmatrix}\boldsymbol{0}^{\top} & \boldsymbol{k}_{\boldsymbol{y}f_k}^{\top}(\boldsymbol{q})\end{bmatrix}\begin{bmatrix}\boldsymbol{K}_{\boldsymbol{0}} & \boldsymbol{K}_{\boldsymbol{y}\boldsymbol{0}}^{\top} \\ \boldsymbol{K}_{\boldsymbol{y}\boldsymbol{0}} & \boldsymbol{K}_{\boldsymbol{y}}\end{bmatrix}^{-1}\begin{bmatrix}\gamma_{\boldsymbol{0}} \\ \Delta\boldsymbol{y}\end{bmatrix} .$$

Then, applying standard inversion formulas for partitioned matrices together with $\boldsymbol{\gamma}_{\boldsymbol{0}} = \boldsymbol{0}$, we can write

$$\hat{f}_k(\boldsymbol{q}) = m_{f_k}(\boldsymbol{q}) + \boldsymbol{k}_{\boldsymbol{y}f_k}^{\top}(\boldsymbol{q})(\boldsymbol{K}_{\boldsymbol{y}} - \boldsymbol{K}_{\boldsymbol{y}\boldsymbol{0}}\boldsymbol{K}_{\boldsymbol{0}}^{-1}\boldsymbol{K}_{\boldsymbol{y}\boldsymbol{0}}^{\top})^{-1}\Delta\boldsymbol{y} \,. \tag{28}$$

Defining $\boldsymbol{l}_{ki}(\boldsymbol{q}) \equiv \boldsymbol{\mathcal{L}}_{\boldsymbol{q}_i}\kappa_k(\boldsymbol{q}_i,\boldsymbol{q})$ as the application of (17) to (13) such that $\boldsymbol{k}_{\boldsymbol{y}f_k} = [\boldsymbol{l}_{ki}]$, it can be verified that

$$\begin{aligned} \boldsymbol{l}_{ki}(\boldsymbol{q}) = &\frac{1}{2}\frac{\partial^k}{\partial\tau^k}\left[\overset{(k)}{\boldsymbol{q}}(t)\overset{(k)}{\boldsymbol{q}}{}_i^{\top}(\tau)\circ\boldsymbol{\Theta}_k(\boldsymbol{q}_i(\tau),\boldsymbol{q}(t))\right]\overset{(k)}{\boldsymbol{q}}(t) \\ &+ \frac{1}{4}(-1)^k\sum_n\sum_m\overset{(k)}{q}_{in}\overset{(k)}{q}_{im}\overset{(k)}{q}_n\overset{(k)}{q}_m\nabla_{\boldsymbol{q}_i}\theta_{knm}(\boldsymbol{q}_i,\boldsymbol{q}) \end{aligned}$$

holds after some differential vector-algebraic computations, where we utilize among other intermediate steps that

$$\nabla_{\dot{\boldsymbol{q}}_i}\nabla_{\boldsymbol{q}_i}^\top \kappa_k(\boldsymbol{q}_i,\boldsymbol{q}) = \begin{cases} 0\,, & k=0 \\ \frac{1}{2}\dot{\boldsymbol{q}}\dot{\boldsymbol{q}}^\top \circ \boldsymbol{\Phi}_k(\boldsymbol{q}_i,\dot{\boldsymbol{q}}_i,\boldsymbol{q})\,, & k=1 \end{cases}.$$

The full derivation is omitted due to space limitations. Further exploiting the commutativity of the Hadamard product and the definition of the projected gradient matrix $\boldsymbol{\Phi}_k$ in (24), we follow that $\delta\boldsymbol{x}_i^\top \boldsymbol{l}_{ki}(\boldsymbol{q}) = \frac{1}{2}\boldsymbol{q}^{(k)\top}\boldsymbol{N}_{ki}(\boldsymbol{q})\boldsymbol{q}^{(k)}$ with $\boldsymbol{N}_{ki}$ as in (27). Then, eliminating the skew-symmetric part of $\boldsymbol{N}_{ki}$ combined with the quadratic priors m_{f_k} from Section IV-B, we reformulate (28) using $\boldsymbol{k}_{\boldsymbol{y}f_k}^\top \boldsymbol{K}_D^{-1}\Delta\boldsymbol{y} = \sum_i \delta\boldsymbol{x}_i^\top \boldsymbol{l}_{ki}$ and obtain (25)–(26). ■

Essentially, Lemma 1 follows from the quadratic kernel structure (13). The summation (26) represents an extension of matrix decompositions from vector to matrix spaces which are aligned with the Lagrangian structure (1)–(2).

B. Positive Definiteness

We now move on to investigate positive definiteness of the matrix estimates $\hat{\boldsymbol{H}}_k$. Before that, however, we impose a minor structural restriction on the used kernel functions.

Assumption 5: The Cholesky matrix kernel components r_{knm} from (15) and the gravitational covariance k_G from (11a) belong to the class of metric kernels $\mathcal{M}$ given by

$$\mathcal{M} = \left\{\boldsymbol{\Lambda} \succ \boldsymbol{0} \,\middle|\, k_{\mathcal{M}}(|\boldsymbol{d}|) = \sigma^2 \exp\left(-1/2\boldsymbol{d}^\top\boldsymbol{\Lambda}\boldsymbol{d}\right)\right\} \quad (29)$$

parametrized by $\sigma_{knm}, \boldsymbol{\Lambda}_{knm}$ and $\sigma_G, \boldsymbol{\Lambda}_G$, respectively.

Note that class $\mathcal{M}$ in Assumption 5 includes a wide variety of covariance functions including the commonly used squared exponential kernel [15]. Therein, kernels $k(\boldsymbol{q},\boldsymbol{q}') \equiv k_{\mathcal{M}}(|\boldsymbol{d}|)$ are Gaussian radial basis functions. They depend only on the squared length of the difference

$$\boldsymbol{d} := \boldsymbol{q} - \boldsymbol{q}' \quad (30)$$

in the Riemann space [20, p. 243] defined by metric $\boldsymbol{\Lambda} \succ \boldsymbol{0}$. The special case of variance $\sigma^2 = (2\pi)^{-N/2}|\det\boldsymbol{\Lambda}^{1/2}|$ leads to i.i.d. $\boldsymbol{d} \sim \mathcal{N}(\boldsymbol{0},\boldsymbol{\Lambda}^{-1})$.

We can now explicitly state the components r_{knm} of the Cholesky factor $\boldsymbol{R}_k$ from (15). Introducing for $0 \leq n \leq m \leq N$ the functions $\tilde{r}_{knm} \in \mathcal{M}$ from Assumption 5 with identical Riemannian hypermetrics $\boldsymbol{\Lambda}_{knm} = \boldsymbol{\Lambda}_k$, we set

$$r_{knm} \equiv \tilde{r}_{knm}\,, \quad \boldsymbol{\Sigma}_{f_k} := [\sigma_{knm}^2]\,, \quad \boldsymbol{\Sigma}_{\boldsymbol{d}_k} := \boldsymbol{\Lambda}_k^{-1}\,. \quad (31)$$

Note that the hypervariance $\boldsymbol{\Sigma}_{f_k}$ is upper-triangular.

We now make a stochastic analysis of the data set $\mathcal{D}$ gathered via (3a)–(3b), enabling the probabilistic investigation of positive definiteness preservation. Based on the product $\overset{(n)}{\boldsymbol{p}}_i$ and difference $\boldsymbol{d}_j$ vectors

$$\overset{(n)}{\boldsymbol{p}}_i := \delta\boldsymbol{x}_i \circ \overset{(n)}{\boldsymbol{q}}_i\,, \quad \forall n \in \{k, 2k\}\,, \quad (32a)$$

$$\boldsymbol{d}_j := \boldsymbol{q} - \boldsymbol{q}_j\,, \quad \forall j = 1,\ldots,D\,, \quad (32b)$$

we define for $\boldsymbol{\mu}_k, \boldsymbol{\nu}_k \in \mathbb{R}^D$ the composite random variable

$$\boldsymbol{\beta}_k := \boldsymbol{\mu}_k + \boldsymbol{\nu}_k \quad (33)$$

for those $i \in \{1,\ldots,D\}$ which satisfy component-wise $\forall n \in \{k, 2k\}$, $k \in \{0,1\}$, the condition $\boldsymbol{p}_i^{(n)} > \boldsymbol{0}$, by

$$\begin{aligned} \mu_{ki} &= \Upsilon_k\Big(\overset{(2k)}{\boldsymbol{p}}_i\Big)\det\operatorname{diag}\Big(\boldsymbol{\varrho}_k^{\downarrow} \circ \overset{(2k)}{\boldsymbol{p}}_i^{\uparrow}\Big) \\ &\quad + 2k\dot{\vartheta}_{ki}\begin{cases} \Upsilon_k\Big(\overset{(k)}{\boldsymbol{p}}_i\Big)\det\operatorname{diag}\Big(\boldsymbol{\varrho}_k^{\downarrow} \circ \overset{(k)}{\boldsymbol{p}}_i^{\uparrow}\Big) & \dot{\vartheta}_{ki} \geq 0 \\ \varrho_{k1}\max_n \overset{(k)}{p}_{in} & \dot{\vartheta}_{ki} < 0 \end{cases} \\ \nu_{ki} &= \varphi_{ki}\begin{cases} \Upsilon_k\Big(\overset{(k)}{\boldsymbol{q}}_i^{\circ 2}\Big)\det\operatorname{diag}\Big(\boldsymbol{\varrho}_k^{\downarrow} \circ \overset{(k)}{\boldsymbol{q}}_i^{\circ 2\uparrow}\Big) & \varphi_{ki} \geq 0 \\ \varrho_{k1}\max_n \overset{(k)}{q}_{in}^2 & \varphi_{ki} < 0 \end{cases} \end{aligned}$$

with constant, decreasingly ordered, radial vector $\boldsymbol{\varrho}_k^{\downarrow} \in \mathbb{R}^N$, $\boldsymbol{\varrho}_k^{\downarrow} > \boldsymbol{0}$. Here, the projection φ_{ki}, the differential angle $\dot{\vartheta}_{ki}$ and the normalizing function $\Upsilon_k(\boldsymbol{p}_i)$ are given by

$$\varphi_{ki}(\delta\boldsymbol{x}_i) = (-1)^k \boldsymbol{d}_i^\top \boldsymbol{\Lambda}_k \delta\boldsymbol{x}_i\,, \quad (34a)$$

$$\dot{\vartheta}_{ki}(\boldsymbol{q}_i(\tau)) = \varphi_{ki}(\dot{\boldsymbol{q}}_i(\tau))\,, \quad (34b)$$

$$\Upsilon_k(\boldsymbol{p}_i) = 1/\Pi_{j=1}^{N-1}\lambda_j(\boldsymbol{\Sigma}_{f_k}^2\operatorname{diag}(\boldsymbol{p}_i))\,. \quad (34c)$$

If $\boldsymbol{p}_i^{(n)} > \boldsymbol{0}$ does not hold $\forall n \in \{k, 2k\}$, we define

$$\beta_{ki} := \exp(\|\boldsymbol{d}_i\|_{\boldsymbol{\Lambda}_k}^2)\lambda_N(\hat{\boldsymbol{H}}_{ki})\,, \quad \overset{(n)}{\boldsymbol{p}}_i \not> \boldsymbol{0}\,, \quad (35)$$

where $\hat{\boldsymbol{H}}_{ki} = \frac{1}{2}(\boldsymbol{N}_{ki} + \boldsymbol{N}_{ki}^\top)$ from (26). The random variable $\boldsymbol{\beta}_k$ from (33)–(35) is the result of nonlinearly transformed GRVs stemming from the data set $\mathcal{D}$. We now formulate the main result.

Theorem 1: Consider the posterior, generalized, matrix-valued L-GP estimate $\hat{\boldsymbol{H}}_k(\boldsymbol{q})$ from (26) with metric covariances $r_{knm} \in \mathcal{M}$ forming the Cholesky kernel $\boldsymbol{\Theta}_k(\boldsymbol{q}_i,\boldsymbol{q})$ from (14) identically specified by Riemannian hypermetric $\boldsymbol{\Sigma}_{\boldsymbol{d}_k}^{-1}$ via Assumption 5 and (31). Positive definiteness of the generalized matrix estimate is guaranteed with probability

$$\Pr\{\hat{\boldsymbol{H}}_k(\boldsymbol{q}) \succ \boldsymbol{0}\} = 1 - \delta_k(\boldsymbol{q})\,, \quad (36)$$

where the physical inconsistency measure $\delta_k(\boldsymbol{q})$ is upper bounded according to

$$\delta_k(\boldsymbol{q}) \leq \Pr\left\{\lambda_N(\boldsymbol{H}_{k0}) + \exp\left[-\operatorname{vec}(\boldsymbol{\Sigma}_{\boldsymbol{d}_k}^{-1})^\top \boldsymbol{D}(\boldsymbol{q})\right]\boldsymbol{\beta}_k \leq 0\right\}$$

based on the column-wise Kronecker-constructed squared distance matrix

$$\boldsymbol{D}(\boldsymbol{q}) = [\boldsymbol{d}_j(\boldsymbol{q}) \otimes \boldsymbol{d}_j(\boldsymbol{q})]$$

which depends only on the absolute values of the positional joint distances $|\boldsymbol{d}_j(\boldsymbol{q})| = |\boldsymbol{q} - \boldsymbol{q}_j|$.

Proof: Firstly, we express the basis matrices (27) from Lemma 1 writing $\boldsymbol{\Theta}_k \equiv \tilde{\boldsymbol{\Theta}}_k(\boldsymbol{d})$ based on Assumption 5 and (31). Proceeding analogously via $\boldsymbol{\Phi}_k \equiv \tilde{\boldsymbol{\Phi}}_k(\boldsymbol{d},\delta\boldsymbol{x}_i)$, we follow that

$$\tilde{\boldsymbol{\Phi}}_k(\boldsymbol{d}_i,\delta\boldsymbol{x}_i) = (-1)^k 2\varphi_{ki}(\delta\boldsymbol{x}_i)\tilde{\boldsymbol{\Theta}}_k(\boldsymbol{d}_i)$$

with the projection φ_{ki} and the difference vector $\boldsymbol{d}_i$ as in (34a) and (32b), respectively. Then, we can write for each matrix summand $\boldsymbol{N}_{ki}$ in (27) that

$$\boldsymbol{N}_{ki} = \bar{\boldsymbol{N}}_{ki} \circ \tilde{\boldsymbol{\Theta}}_k(|\boldsymbol{d}_i|)\,,$$

where

$$\bar{\boldsymbol{N}}_{ki} = \sum_{n=0}^{k} \big(2\dot{\vartheta}_{ki}(\boldsymbol{q}_i)\big)^{k-n} \delta\boldsymbol{x}_i \overset{(n+k)}{\boldsymbol{q}}{}_i^{\top} + \varphi_{ki}(\delta\boldsymbol{x}_i) \overset{(k)}{\boldsymbol{q}}_i \overset{(k)}{\boldsymbol{q}}{}_i^{\top} \quad (37)$$

dropping dependencies on $\boldsymbol{q}$ for the sake of notational brevity. Consider now the i-th summand $\hat{\boldsymbol{H}}_{ki} = \frac{1}{2}(\boldsymbol{N}_{ki} + \boldsymbol{N}_{ki}^{\top})$ from (26). Exploiting

$$\tilde{\boldsymbol{\Theta}}_k(|\boldsymbol{d}_i|) = \exp(-\|\boldsymbol{d}_i\|^2_{\boldsymbol{\Lambda}_k})\boldsymbol{\Sigma}^2_{f_k} \,,$$

where $\boldsymbol{\Sigma}^2_{f_k} = \boldsymbol{\Sigma}^{\top}_{f_k}\boldsymbol{\Sigma}_{f_k}$, and its eigenvalue decomposition

$$\boldsymbol{\Sigma}^2_{f_k} = \boldsymbol{W}_k \mathrm{diag}(\boldsymbol{\varrho}_k)\boldsymbol{W}_k^{\top}$$

with decreasingly ordered $\boldsymbol{\varrho}_k = \boldsymbol{\lambda}^{\downarrow}(\boldsymbol{\Sigma}^2_{f_k}) > \boldsymbol{0}$, we now leverage the eigenvalue behavior of the sum of Hermitian matrices [21] in combination with Weyl's inequality [22, Theorem III.2.1]. Thus, with the Hermitian summation $\hat{\boldsymbol{H}}_k = \sum_i \hat{\boldsymbol{H}}_{ki}$ as in (26), we derive

$$\lambda_N(\hat{\boldsymbol{H}}_k) \geq \sum_{i=1}^{D} \lambda_N(\hat{\boldsymbol{H}}_{ki}) \,.$$

Then, utilizing the product constructions (37), we obtain

$$\lambda_N(\hat{\boldsymbol{H}}_k) \geq \lambda_N(\boldsymbol{H}_{k0}) + \exp\left[-\mathrm{vec}(\boldsymbol{\Lambda}_k)^{\top}\boldsymbol{D}\right]\boldsymbol{\beta}_k \quad (38)$$

by successively exploiting the invariance of eigenvalues w.r.t. similarity transforms and by using $\|\boldsymbol{d}_i\|^2_{\boldsymbol{\Lambda}_k} = (\boldsymbol{d}_i \otimes \boldsymbol{d}_i)^{\top}\mathrm{vec}(\boldsymbol{\Lambda}_k)$ as well as the definition of the random vector $\boldsymbol{\beta}_k \in \mathbb{R}^D$ as given component-wise by (33)–(35). Taking the complement of the set in which the lower bound (38) is positive, we arrive at (36) and finish the proof. ■

Note that, for a given data set $\mathcal{D}$, the upper bound on δ_k in Theorem 1 can be used to deterministically estimate the domain in which positive definiteness is fulfilled. A probabilistic computation can be made by means of the distribution of products and quotients of continuous, nonstandardized, independent Normal random variables [23].

C. Equilibria

Despite the point-wise inclusion of $\boldsymbol{\gamma_0} = \boldsymbol{0}$ in the distribution (19) being stochastic, we can provide an analytical guarantee for the preservation of the equilibrium at the origin.

Theorem 2: The L-GP-based potential mean estimate $\hat{V}(\boldsymbol{q}) \equiv \mathrm{E}[V(\boldsymbol{q})|\boldsymbol{y}, \boldsymbol{\gamma_0}]$ is guaranteed to have an equilibrium:

$$\hat{V}(\boldsymbol{0}) = 0 \,, \quad \nabla_{\boldsymbol{q}}\hat{V}(\boldsymbol{0}) = \boldsymbol{0} \,. \quad (39)$$

Proof: Analogously to the derivation of (28), we now marginalize (19) over T instead of V, and condition on $\boldsymbol{y}$ as well as $\boldsymbol{\gamma_0}$. This leads to $\hat{V}(\boldsymbol{q}) \equiv \mathrm{E}[V(\boldsymbol{q})|\boldsymbol{y}, \boldsymbol{\gamma_0}]$ with

$$\hat{V}(\boldsymbol{q}) = m_V(\boldsymbol{q}) + \left(\boldsymbol{k}^{\top}_{\boldsymbol{y}V} - \boldsymbol{k}^{\top}_{\boldsymbol{0}V}\boldsymbol{K}_{\boldsymbol{0}}^{-1}\boldsymbol{K}^{\top}_{\boldsymbol{y0}}\right)\boldsymbol{K}_D^{-1}\Delta\boldsymbol{y} \,. \quad (40)$$

Based on the quadratic form of $\hat{U}$ proven in Lemma 1, we set $U \equiv 0$ w.l.o.g. and obtain $V = G$. Exploiting the structure (29) of the gravitational kernel k_G according to Assumption 5 then leads to $\boldsymbol{k}^{\top}_{\boldsymbol{0}V}(\boldsymbol{0}) = [\sigma^2_G, \boldsymbol{0}^{\top}]$, block-diagonal $\boldsymbol{K}_{\boldsymbol{0}} = \sigma^2_G\mathrm{diag}(1, \boldsymbol{\Lambda}_G)$ and thus $\boldsymbol{k}^{\top}_{\boldsymbol{0}V}(\boldsymbol{0})\boldsymbol{K}_{\boldsymbol{0}}^{-1} = [1, \boldsymbol{0}^{\top}]$. Therefore,

$$\boldsymbol{k}^{\top}_{\boldsymbol{y}V}(\boldsymbol{0}) = \boldsymbol{k}^{\top}_{\boldsymbol{0}V}(\boldsymbol{0})\boldsymbol{K}_{\boldsymbol{0}}^{-1}\boldsymbol{K}^{\top}_{\boldsymbol{y0}} \,. \quad (41)$$

Plugging (41) into (40) and using $m_G(\boldsymbol{0}) = 0$ due to Assumption 3, we follow $\hat{V}(\boldsymbol{0}) = 0$ holds. Similarly applying the ∇ operator to (40), we compute $\nabla_{\boldsymbol{q}}\boldsymbol{k}^{\top}_{\boldsymbol{y}V}(\boldsymbol{0}) = \nabla_{\boldsymbol{q}}\boldsymbol{k}^{\top}_{\boldsymbol{0}V}(\boldsymbol{0})\boldsymbol{K}_{\boldsymbol{0}}^{-1}\boldsymbol{K}^{\top}_{\boldsymbol{y0}}$. Thus we can conclude that $\nabla_{\boldsymbol{q}}\hat{V}(\boldsymbol{0}) = \boldsymbol{0}$ holds, again for $\nabla_{\boldsymbol{q}}m_G(\boldsymbol{0}) = \boldsymbol{0}$ as in Assumption 3. ■

Corollary 1: The estimative Lagrangian $\hat{L} \equiv \mathrm{E}[L|\boldsymbol{y}, \boldsymbol{\gamma_0}]$ based on (19) is analytically guaranteed to have a stationary point at $\boldsymbol{q} = \dot{\boldsymbol{q}} = \boldsymbol{0}$, i.e., $\nabla_{\boldsymbol{q},\dot{\boldsymbol{q}}}\hat{L}(\boldsymbol{0}, \boldsymbol{0}) = \boldsymbol{0}$, with $\hat{L}(\boldsymbol{0}, \boldsymbol{0}) = 0$.

Proof: Since $\hat{L} = \hat{T} - \hat{G} - \hat{U}$, this directly follows from the preservation of quadratic kinetic and elastic forms proven in Lemma 1 combined with the guaranteed gravitational equilibrium from Theorem 2, i.e., $\hat{G}(\boldsymbol{0}) = 0$. ■

D. Energy Conservation

Lastly, the L-GP is also equivalent to a dynamic system. We investigate this intuition in the following.

Proposition 2: Upon explicit inclusion of a test torque $\boldsymbol{\tau}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$ in the form of (18) in (19), the conditional expectation $\hat{\boldsymbol{\tau}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) \equiv \mathrm{E}[\boldsymbol{\tau}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})|\boldsymbol{y}, \boldsymbol{\gamma_0}]$ is given by

$$\hat{\boldsymbol{\tau}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) = \hat{\boldsymbol{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \hat{\boldsymbol{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \hat{\boldsymbol{g}}(\boldsymbol{q}) \quad (42)$$

with $\hat{\boldsymbol{C}} = \frac{1}{2}[\nabla_{\dot{\boldsymbol{q}}}\nabla^{\top}_{\boldsymbol{q}}\hat{T} + \dot{\hat{\boldsymbol{M}}} - \nabla_{\boldsymbol{q}}\nabla^{\top}_{\dot{\boldsymbol{q}}}\hat{T}]$, $\hat{\boldsymbol{M}}$ from (26) for $k = 1$, and $\hat{\boldsymbol{g}} = \nabla_{\boldsymbol{q}}\hat{V}$ from (40) and (26) for $k = 0$.

Proof: Follows as a direct consequence of the kernel construction via the Lagrangian-differential operator (17) and the equivalent representation of (1) by (2). Thus, we include (18) in (19) and get

$$\mathrm{E}[\boldsymbol{\tau}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})|\boldsymbol{y}, \boldsymbol{\gamma_0}] \equiv \boldsymbol{\mathcal{L}}_{\boldsymbol{q},\dot{\boldsymbol{q}}}\hat{L}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \hat{\boldsymbol{\tau}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$$

after marginalization and conditioning, proving the result. ■

We now show that the equivalent L-GP dynamics (42) preserve the physical property of energy conservation as a last result. For notational consistency, we use the (estimative) state vector $\hat{\boldsymbol{x}}^{\top} = [\boldsymbol{q}^{\top}, \dot{\boldsymbol{q}}^{\top}]$ in the following.

Corollary 2: Consider the system $\dot{\hat{\boldsymbol{x}}} = \hat{\boldsymbol{\phi}}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}})$ defined $\forall\hat{\boldsymbol{x}} \in \boldsymbol{\Xi}$ in the compact domain $\boldsymbol{\Xi} \subseteq \boldsymbol{\mathcal{X}} \subseteq \mathbb{R}^{2N}$ with

$$\boldsymbol{\Xi} = \{\boldsymbol{0}\} \cup \{\hat{\boldsymbol{M}}(\hat{\boldsymbol{x}}_1) \succ \boldsymbol{0} \wedge \hat{V}(\hat{\boldsymbol{x}}_1) \geq -1/2\lambda_N(\hat{\boldsymbol{M}})\|\hat{\boldsymbol{x}}_2\|^2\}$$

spanned $\forall\hat{\boldsymbol{u}} \in \mathbb{R}^N$ by the GP (18) with estimate (42), where

$$\hat{\boldsymbol{\phi}}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}}) = \begin{bmatrix} \hat{\boldsymbol{x}}_2 \\ \hat{\boldsymbol{M}}^{-1}(\hat{\boldsymbol{x}}_1)\big(\hat{\boldsymbol{u}} - \hat{C}(\hat{\boldsymbol{x}}_1, \hat{\boldsymbol{x}}_2)\hat{\boldsymbol{x}}_2 - \hat{\boldsymbol{g}}(\hat{\boldsymbol{x}}_1)\big)\end{bmatrix} . \quad (43)$$

The dynamic system (43) with state vector $\hat{\boldsymbol{x}}^{\top} = [\hat{\boldsymbol{x}}_1^{\top}, \hat{\boldsymbol{x}}_2^{\top}]$ and input $\hat{\boldsymbol{u}} = \hat{\boldsymbol{\tau}}$ is passive and, moreover, lossless, with respect to the energy storage function $\hat{E} = \hat{T} + \hat{V}$, where

$$\hat{E}(\hat{\boldsymbol{x}}) = 1/2\hat{\boldsymbol{x}}_2^{\top}\hat{\boldsymbol{M}}(\hat{\boldsymbol{x}}_1)\hat{\boldsymbol{x}}_2 + \hat{V}(\hat{\boldsymbol{x}}_1) \,, \quad (44)$$

based on (25), (40), and the system output mapping $\hat{\boldsymbol{y}} = \hat{\boldsymbol{x}}_2$.

Proof: From the definition of $\boldsymbol{\Xi}$, we can follow that (43) is locally Lipschitz in the domain $\boldsymbol{\Xi} \times \mathbb{R}^N$. Also, the storage function (44) is positive semidefinite, since $\hat{E}(\boldsymbol{0}) = 0$ due to Corollary 1 and $\lambda_N(\hat{\boldsymbol{M}})\|\hat{\boldsymbol{x}}_2\|^2 \leq \hat{\boldsymbol{x}}_2^{\top}\hat{\boldsymbol{M}}\hat{\boldsymbol{x}}_2$ for $\hat{\boldsymbol{M}} \succ \boldsymbol{0}$ ensures $\hat{E}(\hat{\boldsymbol{x}}) \geq 0$, $\forall\boldsymbol{x} \in \boldsymbol{\Xi}$. The output function $\hat{\boldsymbol{y}} = \hat{\boldsymbol{x}}_2$ is continuous and for the dynamics $\hat{\boldsymbol{\phi}}(\boldsymbol{0}, \boldsymbol{0}) = \boldsymbol{0}$ holds. Taking the time derivative of (44) along the trajectories (43), we obtain the power fed into the system by input $\hat{\boldsymbol{u}}$ computing

$$\begin{aligned}\dot{\hat{E}} &= \hat{\boldsymbol{x}}_2^\top \left(\hat{\boldsymbol{M}}(\hat{\boldsymbol{x}}_1)\dot{\hat{\boldsymbol{x}}}_2 + \frac{1}{2}\dot{\hat{\boldsymbol{M}}}(\hat{\boldsymbol{x}}_1)\hat{\boldsymbol{x}}_2 + \nabla_{\hat{\boldsymbol{x}}_1}\hat{V}(\hat{\boldsymbol{x}}_1) \right) \\ &= \hat{\boldsymbol{x}}_2^\top \left(\hat{\boldsymbol{M}}(\hat{\boldsymbol{x}}_1)\dot{\hat{\boldsymbol{x}}}_2 + \hat{C}(\hat{\boldsymbol{x}}_1, \hat{\boldsymbol{x}}_2)\hat{\boldsymbol{x}}_2 + \hat{\boldsymbol{g}}(\hat{\boldsymbol{x}}_1) \right) \\ &= \hat{\boldsymbol{y}}^\top \hat{\boldsymbol{u}} \,,\end{aligned}$$

where we have exploited the symmetry of $\hat{\boldsymbol{M}} = \hat{\boldsymbol{M}}^\top$, the identity $\dot{\hat{\boldsymbol{M}}} = \hat{\boldsymbol{C}} + \hat{\boldsymbol{C}}^\top$ and the skew-symmetry of $\dot{\hat{\boldsymbol{M}}} - 2\hat{\boldsymbol{C}}^\top$. Thus, according to [24], the system (43) is lossless. ■

Note that from the passivity of system (43) we can also directly follow stability of the origin of the unforced system $\dot{\hat{\boldsymbol{x}}} = \hat{\boldsymbol{\phi}}(\hat{\boldsymbol{x}}, \boldsymbol{0})$ by taking (44) as a Lyapunov function [24].

VI. Numerical Illustrations

In this section, we now demonstrate the efficacy of our approach and validate the derived theoretical results, choosing a simple example for the sake of easy interpretation.

A. Setup

We benchmark our method on the two-link robotic manipulator from [12, p. 164]. Gravity $g = 10$ ms^{-2} acts along the positive x-axis, leading to an equilibrium at the origin $\boldsymbol{q} = \boldsymbol{0}$. The links have unit masses $m_n = 1$ kg and unit lengths $l_n = 1$ m for $n \in \{1, 2\}$. Estimates are available but erroneous: $\hat{m}_n = (1 + \Delta_n)m_n$, $\hat{l}_n = (1 + \Delta_n)l_n$, where $\Delta_n = (-1)^n/2$.

For all of the numerical simulations, we use only 25 training pairs equally distanced on the domain $\boldsymbol{q} \in [-1, 1]^2$ with fixed $\dot{q}_n = (-1)^n$ and $\ddot{\boldsymbol{q}} = \boldsymbol{1}$. The torque measurement noise has covariance $\boldsymbol{\Sigma}_{\boldsymbol{\varepsilon}_i} = \sigma_{\boldsymbol{\varepsilon}}^2 \boldsymbol{I}$, $\sigma_{\boldsymbol{\varepsilon}} = 0.1$ Nm, while the differential process noise corrupting the accelerations has covariance $\boldsymbol{\Sigma}_{\boldsymbol{\alpha}_i} = \sigma_\alpha^2 \boldsymbol{I}$, $\sigma_\alpha = \pi/180$ rad/s^2. Positions and velocities are kept noisefree, enabling explicit numerical integration with ode45. We reduce the kinetic mass inertia hypermetrics $\boldsymbol{\Lambda}_{knm} = \boldsymbol{\Lambda}_k$, $\forall n, m \in \{1, 2\}$, to the constant Euclidian form $\boldsymbol{\Lambda}_k^{-1} = \sigma_{\boldsymbol{d}_T}^2 \boldsymbol{I}$, where $k = 1$, and fix $\sigma_{\boldsymbol{d}_T} = 10^2$. The same is done for the gravitational distance covariance keeping $\boldsymbol{\Sigma}_{\boldsymbol{d}_G}^{1/2} = \mathrm{diag}([1.6, 2.7])$. The remaining radial gravitational hypervariance σ_G^2 along with the Cholesky-kinetic upper-triangular hypercovariance $\boldsymbol{\Sigma}_{f_k}$, where again $k = 1$, are optimized via the log-likelihood.

B. Results

As a first experiment, we demonstrate the closed-loop applicability of the L-GP model. For this, we compare a standard PD tracking controller [12, p. 194] with its L-GP-based version as shown in Fig. 1. Starting from the equilibrium, the controllers with gains $\boldsymbol{K}_p = \boldsymbol{K}_d = 10\boldsymbol{I}$ have the task of following a sinusoidal reference trajectory $\boldsymbol{q}_d(t) = a_d \sin t\boldsymbol{1}$ with amplitude $a_d = \pi/2$. The simulation shows that the L-GP-based version considerably improves the accuracy, demonstrating reliable performance despite the suboptimal distribution of the data points, independent from the reference trajectory, requiring substantial extrapolation.

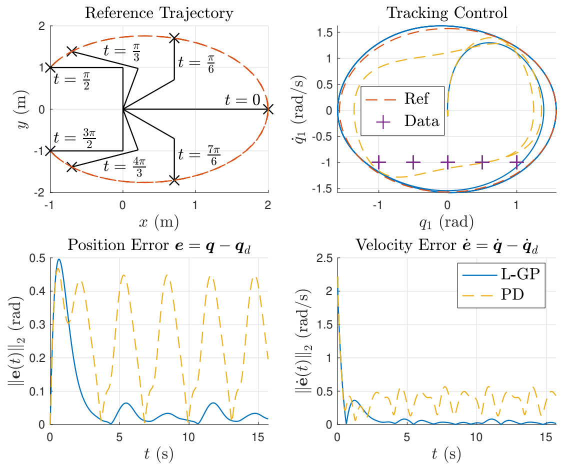

Fig. 1. Tracking performance of the standard PD controller (dashed yellow line) and its L-GP augmented version (solid blue line). The top left plot shows the reference trajectory (dashed red line) w.r.t. the end-effector in its cartesian work space, the top right the closed-loop behavior in the state space of the first joint along with the training data points (purple crosses). The bottom two subfigures indicate the Euclidian norms of position and velocity tracking errors over time.

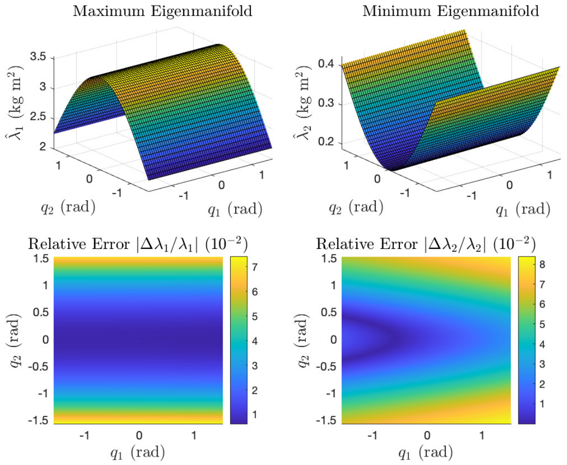

Fig. 2. Eigenmanifolds of the generalized inertia matrix of the two-link robot over the domain $\boldsymbol{q} \in [-\pi/2, \pi/2]^2$. The top left and right surface plots show the maximum and minimum estimative eigenvalue manifolds of the L-GP-based estimate $\hat{\boldsymbol{M}}(\boldsymbol{q})$, respectively, while the bottom two subfigures are heat maps indicating the relative approximation errors in percent.

Next, we validate the physical consistency of the L-GP. Therefore, using the same parametrization as in the previous experiment, we evaluate the mass inertia estimate over the joint domain $\boldsymbol{q} \in [-a_d, a_d]^2$ and investigate its eigenmanifolds, as shown in Fig. 2. Clearly, the L-GP accurately approximates the positive definite function space despite only being a subcomponent of the input-output relation. The consistency of the gravitational potential estimate with equilibrium is validated in Fig. 3, along with the conservatism of the equivalent L-GP-based dynamics simulated for different initial conditions $\hat{\boldsymbol{x}}(0) = [a_0\boldsymbol{1}^\top, \boldsymbol{0}^\top]^\top$ with displacement amplitudes $a_0 = 0.1$ (red line), $a_0 = 0.5$ (yellow line)

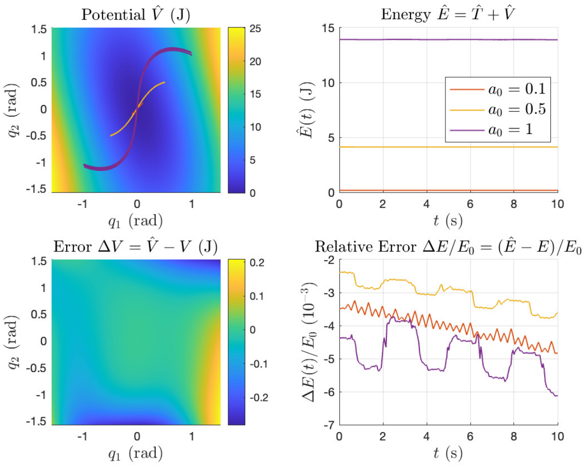

Fig. 3. Potential energy estimate and free trajectories of the L-GP equivalent dynamic system (top left plot) with initial conditions $\boldsymbol{q}(0) = a_0\mathbf{1}$ and $\dot{\boldsymbol{q}}(0) = \mathbf{0}$ for $a_0 = 0.1$ (red line), $a_0 = 0.5$ (yellow line) and $a_0 = 1$ (purple line). The constant time evolution of the energy estimates are shown in the top right plot for the same trajectories. The bottom left plot visualizes the signed potential approximation error, the bottom right one the signed relative energy error in permille.

and $a_0 = 1$ (purple line). The signed potential and energy approximation errors show their properties of local positive definiteness and passivity, respectively.

VII. Conclusion

This paper presented an approach for the identification of uncertain Lagrangian systems exploiting prior physical knowledge by kernel construction. Physical consistency of the data-driven method in terms of guarantuees for the fulfillment of certain properties was proven rigorously and validated in numerical simulation, along with the effective applicability of the model to an exemplary tracking control problem. Future research will focus on the extension of the approach to handle dissipative and time-variant systems. Another promising direction is the application of the Lagrangian variance estimate for uncertainty quantification.

VIII. Acknowledgments

This work was supported by the Consolidator Grant ”Safe data-driven control for human-centric systems” (CO-MAN) of the European Research Council (ERC) under grant agreement ID 864686, and by the Horizon 2020 Research and Innovation Action project ”Rehabilitation based on Hybrid neuroprosthesis” (ReHyb) of the European Union (EU) under grant agreement number 871767. The authors gratefully acknowledge the thoughtful comments from L. Evangelisti, Institute of System Dynamics and Control, German Aerospace Center (DLR), and the fruitful discussions with A. Lederer, Chair of Information-oriented Control, Technical University of Munich.

References

[1] A. Gahlawat, P. Zhao, A. Patterson, N. Hovakimyan, and E. Theodorou, “L1-gp: L1 adaptive control with bayesian learning,” in *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, vol. 120 of *Proceedings of Machine Learning Research*, pp. 826–837, PMLR, 10–11 Jun 2020.

[2] F. Berkenkamp, R. Moriconi, A. P. Schoellig, and A. Krause, “Safe learning of regions of attraction for uncertain, nonlinear systems with gaussian processes,” in *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 4661–4666, 2016.

[3] M. P. Deisenroth, D. Fox, and C. E. Rasmussen, “Gaussian processes for data-efficient learning in robotics and control,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 2, pp. 408–423, 2015.

[4] L. Ljung, *System Identification: Theory for the User*. PTR Prentice Hall Information and System Sciences, Pearson, 2nd ed., 1998.

[5] Q. Leboutet, J. Roux, A. Janot, J. R. Guadarrama-Olvera, and G. Cheng, “Inertial parameter identification in robotics: A survey,” *Applied Sciences*, vol. 11, no. 9, 2021.

[6] A. A. Amiri Moghadam, K. Torabi, A. Kaynak, M. N. H. Zainal Alam, A. Kouzani, and B. Mosadegh, “Control-oriented modeling of a polymeric soft robot,” *Soft Robotics*, vol. 3, no. 2, pp. 82–97, 2016.

[7] R. E. Day, “Coupling dynamics in aircraft: A historical perspective,” special publication, NASA Dryden Flight Research Center, Edwards, CA, 1997.

[8] T. I. Fossen and O.-E. Fjellstad, “Nonlinear modelling of marine vehicles in 6 degrees of freedom,” *Mathematical Modelling of Systems*, vol. 1, no. 1, pp. 17–27, 1995.

[9] K. Rath, C. G. Albert, B. Bischl, and U. von Toussaint, “Symplectic gaussian process regression of maps in hamiltonian systems,” *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 31, no. 5, p. 053121, 2021.

[10] 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, 2016.

[11] T. Beckers, D. Kulić, and S. Hirche, “Stable gaussian process based tracking control of euler–lagrange systems,” *Automatica*, vol. 103, pp. 390–397, 2019.

[12] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

[13] A. Mchutchon and C. Rasmussen, “Gaussian process training with input noise,” in *Advances in Neural Information Processing Systems*, vol. 24, Curran Associates, Inc., 2011.

[14] J. E. Johnson, V. Laparra, and G. Camps-Valls, “Accounting for input noise in gaussian process parameter retrieval,” *IEEE Geoscience and Remote Sensing Letters*, vol. 17, no. 3, pp. 391–395, 2020.

[15] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[16] M. Raissi, P. Perdikaris, and G. E. Karniadakis, “Inferring solutions of differential equations using noisy multi-fidelity data,” *Journal of Computational Physics*, vol. 335, pp. 736–746, 2017.

[17] E. Solak, R. Murray-smith, W. Leithead, D. Leith, and C. Rasmussen, “Derivative observations in gaussian process models of dynamic systems,” in *Advances in Neural Information Processing Systems*, vol. 15, MIT Press, 2002.

[18] M. A. Álvarez, L. Rosasco, and N. D. Lawrence, “Kernels for vector-valued functions: A review,” *Foundations and Trends® in Machine Learning*, vol. 4, no. 3, pp. 195–266, 2012.

[19] S. Reece and S. Roberts, “An introduction to gaussian processes for the kalman filter expert,” in *2010 13th International Conference on Information Fusion*, pp. 1–9, 2010.

[20] D. Lovelock and H. Rund, *Tensors, Differential Forms, and Variational Principles*. Dover Publications, 1989.

[21] A. Knutson and T. Tao, “Honeycombs and sums of hermitian matrices,” *Notices Of The American Mathematical Society*, vol. 48, no. 2, pp. 175–186, 2001.

[22] R. Bhatia, *Matrix Analysis*. Graduate Texts in Mathematics, Springer, New York, NY, 1 ed., 1997.

[23] M. D. Springer, *The Algebra of Random Variables*. John Wiley & Sons, Inc., 1979.

[24] H. K. Khalil, *Nonlinear Systems*. Prentice Hall, third ed., 2002.