Physics Enhanced Data-Driven Models With Variational Gaussian Processes

DANIEL L. MARINO AND MILOS MANIC (Fellow, IEEE)

Department of Computer Science, Virginia Commonwealth University, Richmond, VA 23284 USA

CORRESPONDING AUTHOR: DANIEL L. MARINO (e-mail: daniellml55@gmail.com)

This article has supplementary downloadable material available at https://doi.org/10.1109/OJIES.2021.3064820, provided by the authors.

ABSTRACT

Centuries of development in natural sciences and mathematical modeling provide valuable domain expert knowledge that has yet to be explored for the development of machine learning models. When modeling complex physical systems, both domain knowledge and data provide necessary information about the system. In this paper, we present a data-driven model that takes advantage of partial domain knowledge in order to improve generalization and interpretability. The presented approach, which we call EVGP (Explicit Variational Gaussian Process), has the following advantages: 1) using available domain knowledge to improve the assumptions (inductive bias) of the model, 2) scalability to large datasets, 3) improved interpretability. We show how the EVGP model can be used to learn system dynamics using basic Newtonian mechanics as prior knowledge. We demonstrate how the addition of prior domain-knowledge to data-driven models outperforms purely data-driven models.

INDEX TERMS

Bayesian neural networks, domain knowledge, Gaussian process, uncertainty, variational inference.

I. INTRODUCTION

For centuries, scientists and engineers have worked on creating mathematical abstractions of real world systems. This principled modeling approach provides a powerful toolbox to derive white-box models that we can use to understand and analyze physical systems. However, as the complexity of physical systems grow, deriving detailed principled models becomes an expensive and tedious task that requires highly experienced scientists and engineers. Moreover, incorrect assumptions lead to inaccurate models that are unable to represent the real system.

Data-driven black-box models provide an appealing alternative modeling approach that requires little to none domain knowledge. These models are fit to data extracted from the real system, minimizing the problems derived from incorrect assumptions. However, using data-driven models while completely ignoring domain knowledge may lead to models that do not generalize well and are hard to understand. Completely black-box approaches ignore the structure of the problem, wasting resources [1] and making the model less explainable [2].

Gray-box models combine domain knowledge and data as both provide important and complementary information about the system. Domain knowledge can be used to construct a set of basic assumptions about the system, giving the data-driven model a baseline to build upon. Data can be used to fill the gaps in knowledge and model complex relations that were not considered by the domain expert.

In this paper, we explore an approach for embedding domain knowledge into a data-driven model in order to improve generalization and interpretability. The presented gray-box model, which we called EVGP (Explicit Variational Gaussian Process), is a scalable approximation of a sparse Gaussian Process (GP) that uses domain knowledge to define the prior distribution of the GP. In this paper, domain knowledge is extracted from physics-based knowledge, however the EVGP can be applied to any domain.

The work on this paper has three cornerstones (Fig. 1): 1) Gaussian processes are used for learning complex non-linear behavior from data while providing uncertainty estimations, 2) Partial domain knowledge is used as prior in order to improve inductive bias, 3) Variational Inference provides advantageous scalability to large datasets. Inductive bias refers to...
the assumptions made by the model when doing predictions on novel data. The presented approach provides uncertainty estimations which are fundamental in order to avoid the risk associated with overconfidence in unexplored areas [3] and warns the user of possible incorrect estimations [4].

The work in this paper is highly applicable when: 1) modeling physical systems with uncertainty estimations, 2) partial domain knowledge of the system is available, 3) large quantities of data are available. The aim is to help the engineer and take advantage of available knowledge without requiring the derivation of complex and detailed models. Instead, an engineer only has to provide simple, partially formed, models and the EVGP takes care of filling the gaps in knowledge. We show how the EVGP model can be used to learn system dynamics using basic physics laws as prior knowledge. We specifically demonstrate the approach on multi-body physics systems. We chose the pendulum as the base for our discussion to demonstrate how concepts from a simple system can be used as building blocks for priors of 2 DOF systems up to a 7 DOF robot arm.

The contributions of this paper are:

- We present a stochastic data-driven model enhanced with domain knowledge to improve model accuracy using less amount of data when compared with fully data-driven approaches.
- The presented approach uses very simple priors, demonstrating that accuracy improves even when using heavily simplified physics as domain-knowledge.
- Scalability to handle large datasets is achieved by using the sparse variational framework.
- Interpretability is exemplified using the posterior distribution of trainable parameters, i.e. the value of the model parameters after training.

Our objective is to evaluate the benefits of including partially defined (imperfect) models as domain-knowledge to data-driven models. Hence, our experiments compare the performance of the EVGP approach with respect to fully data-driven (black-box) models to evaluate the benefits of the proposed approach.

The rest of the paper is organized as follows: section II presents the proposed EVGP approach; section III presents a set of priors derived from simplified Newtonian dynamics for the EVGP model; section IV presents the experimental section where we compare the EVGP with fully data-driven models; section V presents the related work; section VI discusses extensions of the presented approach to be applied in other domains, explicitly handle noise in dynamic system modeling, and handle non-continuous systems; section VII concludes the paper.

II. EVGP APPROACH - EXPLICIT VARIATIONAL GP

The novel EVGP approach presented in this paper is designed to solve regression problems under uncertainty. Figure 2 offers a visual representation of the approach.

Given a dataset \( D = (x^{(i)}, y^{(i)}) \) composed of input/output pairs of samples \( (x^{(i)}, y^{(i)}) \), we would like to obtain a predictive distribution \( p(y|x, D) \) that estimates the value of the output \( y \) for a given input \( x \). The EVGP model approximates \( p(y|x, D) \) using variational inference. The EVGP is defined as a distribution \( p(y|x, w) \) where \( w \) is a set of parameters with prior distribution \( p(w) \).

In the following sections we describe in detail: A) the EVGP model approach, B) the variational loss function used to train the model, C) the predictive distribution that approximates \( p(y|x, D) \). Appendix A provides a quick description of acronyms and symbols for a quick reference for the reader.

A. EVGP MODEL DEFINITION

The EVGP model takes the following form:

\[
y = g(x) + \epsilon_y; \quad g(x) = h(x)^\top \beta + f(x)
\]

where \( f(x) \sim \mathcal{GP}(0, k(x, x')) \) is a Gaussian process with kernel \( k \), \( \epsilon_y \sim \mathcal{N}(0, \Sigma_y) \) is the observation noise and \( g(x) \) is the denoised prediction for the input \( x \). Figure 2 offers a visual representation of the model. The following is the description of the main components of the EVGP model:

- Domain knowledge is embedded in the explicit function \( h(x)^\top \beta \), parameterized by \( \beta \). The function \( h(x) \) describes a set of features which are explicitly stated by the domain expert (hence the name of our method). \( \beta \) is modeled using a normal distribution with a prior that is also extracted from domain knowledge. In this paper, \( h(x)^\top \beta \) is derived from partially defined Newtonian mechanics.

![Figure 1: EVGP: Variational Gaussian-Process with explicit features.](image1)

![Figure 2: Illustration of the EVGP model. The models uses a function \( h(x)^\top \beta \) to embed domain knowledge, while the GP is used to increase the model capacity to learn complex non-linear relations from data.](image2)
• The Gaussian Process $f(x)$ adds the ability to learn complex non-linear relations, increasing the model capacity of the entire model by supplementing the function $h(x)^T \beta$.

Given a dataset $\mathcal{D}$, the exact predictive distribution $p(y|x, \mathcal{D})$ for the model in Eq. (1) is described in [5]. For the rest of the paper, we refer to the exact distribution as EGP. Computing the EGP predictive distribution has a large computational cost and does not scale well for large datasets.

To alleviate this problem, sparse approximation methods [6] use a small set of $m$ inducing points represented by a tuple $(f_m, X_m)$, where $f_m$ is a vector of size $m$, and $X_m$ is a matrix of size $m \times m$. The inducing points $(f_m, X_m)$ are treated as trainable parameters, which allows to approximate the predictive distribution instead of requiring the entire dataset.

In order to construct a sparse approximation for the model in Eq. (1), we use a set of $m$ inducing points $(f_m, X_m)$ as parameters that will be learned from data. Given $(f_m, X_m)$ and a set of test points $(g, X)$, the prior distribution of the model can be expressed as follows:

$$\begin{bmatrix} g_m \mid \beta \\ g \mid \beta \end{bmatrix} \sim \mathcal{N} \left( \begin{bmatrix} H_m \beta \\ H_x \beta \end{bmatrix}, \begin{bmatrix} K_{mm} & K_{mx} \\ K_{xm} & K_{xx} \end{bmatrix} \right);$$

$$H_m = h(X_m); \quad H_x = h(X)$$

$$g_m = h(X_m) \beta + f_m$$

where $X$ denotes the data matrix, where each row represents an individual sample. The rows of $H_x$ represent the value of the function $h()$ applied to the real samples $X$. The rows of $H_m$ represent the value of the function $h()$ applied to the inducing (learned) points $X_m$. Using the conditional rule for multivariate Gaussian distributions, we obtain the definition of the denoised sparse EVGP model:

$$p(g \mid X, \omega) \sim \mathcal{N}(H_x \beta + \mu_{f(\omega)}, \Sigma_{f(\omega)})$$

$$\mu_{f(\omega)} = K_{xm} K^{-1}_{mm} f_m$$

$$\Sigma_{f(\omega)} = K_{xx} - K_{xm} K^{-1}_{mm} K_{mx}$$

(2)

where $\omega = \{f_m, \beta\}$ are the parameters of our model. Here, $K_{xm}$, $K_{mm}$, $K_{mx}$, and $K_{xx}$ are matrices that represent the value of the Gaussian Process kernel $k(\cdot, \cdot)$ evaluated between $(X, X_m)$, $(X_m, X_m)$, $(X_m, X)$, and $(X, X)$, respectively. Equation (2) defines our scalable EVGP model. In following sections we give prior distributions to the parameters $\omega$ and perform approximate Bayesian inference. Note that Eq. (2) is also conditioned on $X_m$, however we do not indicate this explicitly in order to improve readability.

B. VARIATIONAL LOSS

In this section we present the loss function that we use to fit our model. In this paper we follow a variational Bayesian approach (see supplemental material for a brief overview). Given a training dataset $\mathcal{D}$ and a prior distribution $p(\omega)$, we wish to approximate the posterior distribution $p(\omega | \mathcal{D})$. The posterior of $\omega$ is approximated using a variational distribution $p_\phi(\omega) \approx p(\omega | \mathcal{D})$ parameterized by $\phi$. For the EVGP parameters $\omega = \{f_m, \beta\}$, we use the following variational posterior distributions:

$$p_\phi(f_m) = \mathcal{N}(f_m | \mu, \Sigma)$$

$$p_\phi(\beta) = \mathcal{N}(\beta | \mu_\beta, \Sigma_\beta)$$

where $(\mu, \Sigma)$ are the mean values of the respective normal distributions while $(\mu_\beta, \Sigma_\beta)$ are the covariance matrices. The prior-distributions for $\omega$ are also defined as multivariate normal distributions:

$$p(f_m) = \mathcal{N}(f_m | 0, K_{MM})$$

$$p(\beta) = \mathcal{N}(\beta | \mu_\beta, \Sigma_\beta)$$

these prior distributions represent our prior knowledge, i.e. our knowledge before looking at the data. The matrix $K_{MM}$ is computed using the GP kernel on the inducing points $X_m$, while $\mu_\beta$ and $\Sigma_\beta$ are provided by the expert.

Given the training dataset $\mathcal{D} = (y, X)$, the parameters $\phi$ of $p_\phi(\omega)$ are learned by minimizing the negative Evidence Lower Bound (ELBO). For the EVGP, the negative ELBO takes the following form:

$$\mathcal{L}(\phi) = -\log \mathcal{N}(y \mid H_x \beta + K_{xm} K^{-1}_{mm} a, \Sigma_y)$$

$$+ \frac{1}{2} \left[ \text{Tr}(M_1 A) + \text{Tr}(M_2 B) + \text{Tr}\left(\Sigma_1^{-1} \Sigma f(\omega)\right) \right]$$

$$\mathcal{L}_{KL}$$

(3)

where $M_1 = (K_{mn} K_{mx}) \Sigma_1^{-1} (K_{mn} K_{mx})$, and $M_2 = H_x^T \Sigma_1^{-1} H_x$. The term $\mathcal{L}_{KL}$ is the KL-divergence between the posterior and prior distributions for the parameters:

$$\mathcal{L}_{KL} = D_{KL} (\mathcal{N}(a, A) \mid \mathcal{N}(0, K_{mm}))$$

$$+ D_{KL} (\mathcal{N}(b, B) \mid \mathcal{N}(\mu_\beta, \Sigma_\beta))$$

A detailed derivation of the variational loss in Eq. (3) is presented in the supplemental material. The negative ELBO (Eq. 3) serves as our loss function to learn the parameters $\phi$ given the training dataset $y, X$. In order to scale to very large datasets, the ELBO is optimized using mini-batches. In our case, the parameters of the variational approximation are: $\phi = a, A, b, B, X_m$.

C. PREDICTIVE DISTRIBUTION

After learning the parameters $\phi$, we would like to provide estimations using the approximated variational distribution $p_\phi(\omega)$. Given a set of test points $\hat{X}$, the estimated denoised predictive distribution is computed as an expectation of Eq. (2) w.r.t. $p_\phi(\omega)$:

$$p_\phi(\hat{g} \mid \hat{X}) = \mathbb{E}_{p_\phi(\omega)} [p(\hat{g} \mid \hat{X}, \omega)] = \mathcal{N}(\hat{g} \mid \mu_{\hat{g} \hat{X}}, \Sigma_{\hat{g} \hat{X}})$$

$$\mu_{\hat{g} \hat{X}} = H_x \hat{b} + K_{xm} K^{-1}_{mm} a$$

$$\Sigma_{\hat{g} \hat{X}} = \Sigma_{f(\omega)} + H_x B H_x^T + K_{xm} K^{-1}_{mm} K_{m \hat{X}}$$

(4)

Note that $\hat{g}$ is just a denoised version of $\hat{y}$. Eq. (4) approximates $p(\hat{g} \mid \hat{X}, D)$, using the learned distribution $p_\phi(\omega)$.
The result is equivalent to [7] with the addition of $H, b$ for the mean and $H_{b} B H_{b}^{T}$ for the covariance. These additional terms include the information provided by the prior function $H_{b}$ with the parameters $b$ and $B$ that were learned from data.

The approximated predictive distribution with observation noise is the following:

$$p_{\phi}(\hat{y} | \hat{X}) = \mathcal{N}(\hat{\mathbb{g}}(\hat{\mathbb{g}}(z_{1} \oplus u_{[t]})), \Sigma_{y})$$

where $p_{\phi}(\hat{y} | \hat{X}) \approx p(\hat{y} | \hat{x}, D)$. In the next section, we show how Eq. (4) can be used to model system dynamics and predict the next state of a physical system given the control input and current state.

II. EMBEDDING PHYSICS-BASED KNOWLEDGE

In this paper, we apply the EVGP model to learn the dynamics of a physical system. The state $z_{[t]}$ of the physical system can be modeled using the following recurrent version of the EVGP:

$$z_{[t+1]} \sim g(z_{[t]} \oplus u_{[t]})$$

$$y_{[t]} \sim z_{[t]} + \epsilon_{y}$$

(5)

where $u_{[t]}$ is the control input and $y_{[t]}$ is the measured output of the system at time $t$. The symbol $\oplus$ denotes concatenation and $x_{[t]} = z_{[t]} \oplus u_{[t]}$ is the input to the EVGP model. For example, in the case of a mechatronic system: $u_{[t]}$ are the forces applied by the actuators (e.g. electric motors); $z_{[t]}$ is the position and velocity of the joints; $y_{[t]}$ is the output from the sensors.

We assume independent EVGP models for each output $y_{[t]}$ in the equation (5). The function $g()$ in Eq. (5) is modeled using the EVGP model from Eq. (1) and Eq. (4). The recurrent EVGP primarily handles model uncertainty thanks to the variational approximation used in the EVGP. We do not consider observation or process noise. 1 In the following sections we present how we can use simple Newtonian mechanics to define useful priors $h(x)^{T} \beta$ for the EVGP model.

A. PRIORS DERIVED FROM SIMPLE NEWTONIAN DYNAMICS

Figure 3(a) shows a simple example of a single rigid-body link. The simplest model that we can use for this system comes from Newton’s second law $u = J \dot{q}_{1}$, where $u$ is the torque applied to the system, $J$ is the moment of inertia, and $q_{1}$ is the angle of the pendulum. Using Euler discretization method, we obtain the following state-space representation that serves as the prior $h(x)^{T} \beta$ for our EVGP model:

$$\begin{bmatrix} q_{1}[t+1] \\ \dot{q}_{1}[t+1] \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} q_{1}[t] \\ \dot{q}_{1}[t] \end{bmatrix} + \Delta t \begin{bmatrix} \hat{q}_{1}[t] \\ \hat{u}[t] \end{bmatrix}$$

$$h(x_{[t]})$$

(6)

$$\begin{bmatrix} q_{1}[t+1] \\ \dot{q}_{1}[t+1] \\ u[t] \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & 0 \\ 0 & 1 & \Delta t/J \end{bmatrix} \begin{bmatrix} q_{1}[t] \\ \dot{q}_{1}[t] \\ u[t] \end{bmatrix}$$

$$h(x_{[t]})$$

(7)

We refer to this prior as IF (inertia+force). $\Delta t$ is the discretization time and $q_{1}[t] \dot{q}_{1}[t]$ is the state $z_{[t]}$ of the system. The IF prior in Eq. (7) does not include gravitational effects. Gravity pulls the link to the downward position with a force proportional to $\sin q_{1}$. Hence, a prior that considers gravitational forces can be constructed by including $\sin(q_{1}[t])$:

$$\text{IFG}_{[t+1]} = \begin{bmatrix} 1 & \Delta t & 0 \\ 0 & 1 & \Delta t/J \end{bmatrix} \begin{bmatrix} q_{1}[t] \\ \dot{q}_{1}[t] \\ u[t] \end{bmatrix}$$

$$h(x_{[t]})$$

(8)

we call this prior IFG (inertia+force+gravity). We purposely did not define $J$ and $\gamma$. One of the advantages of the presented approach is that the algorithm can learn the parameters from data if they are not available. If the user does not know the value of $J$ and $\gamma$, a prior with large standard deviation can be provided for these parameters (large $\Sigma_{\beta}$). Although parameters like $J$ and $\gamma$ are easy to compute for a simple pendulum, for more complex systems they may be hard and tedious to obtain. Our objective is to take advantage of domain knowledge and allow the model to fill in the gaps in knowledge.

For the rest of the paper, priors derived from Eq. (7) are referred as IF priors, while Eq. (8) priors are referred as IFG. In the experiments (section IV) we compare the performance for both priors in order to illustrate how performance can be progressively improved with more detailed priors.
B. SIMPLIFIED PRIORS FOR ACROBOT, CARTPOLE AND IIWA

In addition to the pendulum, we consider the Acrobot, Cartpole, and IIWA systems in our analysis. For these systems, we consider much simpler priors than the exact dynamic models derived from multi-body dynamics. We use the same principles shown in the previous section in order to get simple priors for the systems. These rules are summarized as follows:

- The position should increase proportional to the velocity by a factor of $\Delta t$.
- The position should stay the same if the velocity is zero.
- The velocity should stay the same if no external forces are applied.
- For the IFG prior, gravity pulls the links to the downward position proportional to the sine of the angle w.r.t. the horizontal plane. Gravity has no effect when the links are completely down/up.

The objective with these priors is to demonstrate how extremely simplified priors extracted with simple physics can be used to improve performance of data-driven models. Figure 3(c) shows a diagram of the Acrobot system. A simple prior for this system can be constructed using the prior in Eq. (7) for each one of the links of the Acrobot:

$$\text{IF} = \begin{bmatrix} 1 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & \gamma_1 \\ 0 & 0 & 0 & 1 & \gamma_1 \end{bmatrix} \begin{bmatrix} q_1[r] \\ q_2[r] \\ q_1[r] \\ q_2[r] \end{bmatrix}$$ (9)

$$\text{IFG} = \begin{bmatrix} 1 & 0 & \Delta t & 0 & 0 & 0 \\ 0 & 1 & 0 & \Delta t & 0 & 0 \\ 0 & 0 & 1 & 0 & -\gamma_2 & -\gamma_3 \\ 0 & 0 & 0 & 1 & \gamma_1 & 0 & -\gamma_4 \end{bmatrix} \begin{bmatrix} q_1[r] \\ q_2[r] \\ q_1[r] \\ q_2[r] \end{bmatrix}$$ (10)

where $\sin_1 = \sin(q_1[r])$, and $\sin_{12} = \sin(q_1[r] + q_2[r])$. In this case, the input $u[r]$ only drives the second link. The IF and IFG priors for the Cartpole (Figure 3(b)) are the following:

$$\text{IF} = \begin{bmatrix} 1 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & \gamma_1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} q_1[r] \\ q_2[r] \\ q_1[r] \\ q_2[r] \end{bmatrix}$$

For the IIWA system, we constructed the priors following the same rules as before, with an approach that closely resembles to that used for the Acrobot. The priors were constructed using the prior in Eq. (7) for each one of the IIWA links. This results in a matrix $\beta$ with a similar diagonal structure than the matrices in Eq. (10). To compute the IFG factors for $h()$, we derived an equation that can be used for a general 3D link (see Fig. 4). The equation uses forward kinematics to evaluate the contribution of the torque applied by gravity ($v_z$):

$$h_{IFG} \triangleq (v_x \times R_i^T v_z) \cdot v_z$$

where $\times$ denotes cross product, $\cdot$ denotes dot product, and $v_x$ and $v_z$ are unit vectors. The matrix $R_i$ is the rotation matrix of the frame attached to link $i$. This matrix is computed using classic forward kinematics, therefore its value depends on the value of $q$. This equation computes the torque applied by gravity to the axis of rotation $v_z$. The result is separated by monomials, each added as a feature in $h$. We only use unitary vectors as we assume that information like the position of the center of gravity will be captured by the posterior of $\beta$, i.e. the learned parameters.

These priors are extremely simple as they do not consider friction or coriolis/centripetal forces. However, they provide important information about the mechanics of the system.

IV. EXPERIMENTS

In order to evaluate the performance of the presented model, we performed experiments on a set of simulated systems: Pendulum, Cartpole, Acrobot, and IIWA. We also performed qualitative tests on a toy-dataset to visualize the performance of the EVGP model.

We used Drake [8] to simulate the Pendulum, Cartpole, Acrobot and IIWA systems and obtain the control/sensor data used to train and evaluate the EVGP models. We used the squared exponential function for the covariance kernels. The reason for this choice is that all the experiments involve...
continuous systems. The EVGP model was implemented using Tensorflow and the minimization of the negative ELBO loss was done using the ADAM optimizer [9]. The experiments were run in a computer with a single GPU (Nvidia Quadro P5000) with an Intel(R) Xeon(R) CPU (E3-1505 M at 3.00 GHz).

A. EXPERIMENTS ON TOY DATASET

The toy dataset is intended to serve as an illustration of the behavior of the EVGP model and visualize the qualitative differences between several GP models. We use a modified version of the toy dataset used in [10], [7]. The dataset is modified as follows:

\[(y, x) \leftarrow (6y + 3x, x)\]

The modification is intended to provide a global linear trend to the data (See Figure 5). We use this modification to illustrate how the EVGP is able to model both global and local trends. Global trends describe the overall shape of the function over the entire domain, while local trends describe deviations from the global behavior in specific regions of the domain. Figure 5 shows the distribution learned using different versions of a Gaussian Process. Figures 5(a) and Figure 5(c) show the exact posterior distributions for a GP and EGP [5] model, respectively. Figures 5(b) and 5(d) show the variational approximations obtained with a VGP [11] and EVGP model. The standard deviation (black line) is used to visualize the uncertainty. The figures show how the uncertainty grows as we move away from the training dataset.

The original dataset is composed of 200 samples, Figure 5 shows that the variational approximations are able to successfully approximate their exact counterparts with as few inducing points as \(m = 10\). The position of the inducing points are shown with green crosses.

In this case, the prior-knowledge that we provide to the EVGP is a simple linear function \(h(x, \beta) = x\beta_1 + \beta_2\). Figure 5(d) shows how we can use the prior in order to control the global shape of the function. The figure shows how the EGP and EVGP models use the prior knowledge to fit the global behavior of the data (linear) while using the kernels to model the local non-linear behavior.

B. LEARNING SYSTEM DYNAMICS

We evaluated the performance of the EVGP model in learning system dynamics using data obtained from simulations of the Pendulum, Cartpole, Acrobot and IIW A systems. Concretely, we evaluated the accuracy of the EVGP model with IF and IFG priors in predicting the next state of the system given the current control inputs and state.

We show the advantages of EVGP regarding generalization and interpretability as follows:

- For generalization, we evaluate all metrics on a testing dataset that has not been used during training. We show the EVGP achieves lower errors on testing data when compared to other models while using smaller training datasets. This means that the EVGP generalizes better
to previously unseen data while requiring less data for training.

- For interpretability, we illustrate how to interpret the trained model by visualizing the parameter $b$ from the posterior variational distribution.

Data: to evaluate the difference in generalization, we sampled two different datasets for each system: one for training and one for testing. The experimental procedure is described in Algorithm 1. We used the Pendulum, Cartpole, Acrobot, and IIIA provided in the examples of the Drake library [8]. We did not include process noise or observation noise in the simulated systems. For the Pendulum, Cartpole, and Acrobot, the datasets were sampled by simulating the system using random initial states $q_0 \sim N(-1, 1)$ and random control inputs $u \sim N(0, 1)$ drawn from uniform and normal distributions, respectively. Several simulations were performed to obtain the trajectories of states and control inputs which were sampled with a period of $\Delta t = 0.03s$. Table 1 shows the values of the scales ($\alpha, \eta$) that were used to sample the trajectories. In Table 1, $N$ refers to the number of sampled trajectories, $|D|$ refers to the total number of samples. These values were chosen in order to cover at least the range $(-\pi, \pi)$ for the angles on the systems.

For the IIIA, the datasets were obtained by simulating the robot executing a random reference trajectory. The IIIA was controlled by the inverse dynamics controller provided by the Drake library. The controller allowed us to maintain the state of the robot inside a reasonable operating region. The datasets were obtained by simulating the robot starting from a random position $q(0) \sim a \mathcal{U}(-1, 1)$ and random control inputs $u \sim N(0, 1)$ drawn from uniform and normal distributions, respectively. Several simulations were performed to obtain the trajectories of states and control inputs which were sampled with a period of $\Delta t = 0.03s$. Table 1 shows the values of the scales ($\alpha, \eta$) that were used to sample the trajectories. In Table 1, $N$ refers to the number of sampled trajectories, $|D|$ refers to the total number of samples. These values were chosen in order to cover at least the range $(-\pi, \pi)$ for the angles on the systems.

For the IIIA, the datasets were obtained by simulating the robot executing a random reference trajectory. The IIIA was controlled by the inverse dynamics controller provided by the Drake library. The controller allowed us to maintain the state of the robot inside a reasonable operating region. The datasets were obtained by simulating the robot starting from a random position $q(0) \sim a \mathcal{U}(-1, 1)$ and random control inputs $u \sim N(0, 1)$ drawn from uniform and normal distributions, respectively. Several simulations were performed to obtain the trajectories of states and control inputs which were sampled with a period of $\Delta t = 0.03s$. Table 1 shows the values of the scales ($\alpha, \eta$) that were used to sample the trajectories. In Table 1, $N$ refers to the number of sampled trajectories, $|D|$ refers to the total number of samples. These values were chosen in order to cover at least the range $(-\pi, \pi)$ for the angles on the systems.

Baseline: we compare the EVGP model with a standard VGP model [11], a residual VGP (RES-VGP) [12], [11], and a residual Deep Bayesian Neural Network (RES-DBNN) [13], [14]. The VGP model is based on [11] and uses a zero mean prior. The residual VGP and DBNN models assume the system can be approximated as $z_{t+1} = z_t + g_t(z_t \oplus u_t)$, where $g_t$ represents either the VGP or the DBNN model. Approximating residuals is a common approach used to simplify the work for GP and DBNN models [12], [14], [15]. The RES-DBNN model is trained using the local reparameterization trick presented in [13] with Normal variational distributions for the weights. For these set of experiments, we did not consider the exact GP and EGP models given the large number of samples in the training datasets.

Table 2 shows the number of inducing points (m) used for the VGP and EVGP models. The table also shows the number of hidden units for the DBNN model, where [15, 15] means a two-layer network with 15 units in each layer. We used the LeakyRelu activation function.

Table 3 shows the space and time complexity of the models considered in this paper. Space complexity shows how the memory requirements grow asymptotically as we increase the size of the models. Time complexity shows how the execution time grows asymptotically as we increase the size of the models. In this table, $m$ is the number of inducing points, $o$ is the number of outputs, $L$ is the number of hidden layers, and $n$ is the number of hidden units in each layer. To simplify the analysis, we assume all hidden layers of the DBNN have the same number of hidden units. The table shows that the time complexity of the VGP and EVGP models are governed by the matrix inversion $K_{mm}^{-1}$ which is $O(n^3)$. Because we assume completely independent VGP and EVGP models for each output of the system, their complexity also depends on

### Table 1: Parameters Used to Collect Training and Testing Data

| System      | $\alpha$ | $\eta$ | $N$  | $\Delta t$ (sec) | $|D|$ |
|-------------|----------|--------|------|------------------|------|
| Pendulum    | $[\pi, 0.5]$ | 1.0    | 48   | 0.03             | 4800 |
| Cartpole    | $[1.0, \pi, 0.5, 0.5]$ | 100.0  | 48   | 0.03             | 4800 |
| Acrobot     | $[\pi, 1.0, 0.5, 0.5]$ | 0.5    | 93   | 0.03             | 9300 |
| IIWA        | 0.8      | 0.03   | 1500 | 0.006            | 225000 |

### Algorithm 1 Experimental Procedure

**Input:** SYSTEM to sample data from. MODEL to be tested.

1. sample TRAIN dataset. Sample N trajectories from SYSTEM using parameters in Table 1
2. sample TEST dataset. Sample N trajectories from SYSTEM using parameters in Table 1
3. Instantiate MODEL using parameters in Table 2
4. train the MODEL using TRAIN dataset and ADAM optimizer
   
   $$\text{MODEL} \leftarrow \text{ADAM}($$MODEL, TRAIN$$)$$

5. test the model using the TEST dataset
   
   Error, STD, CR $\leftarrow$ MODEL($$TEST$$)

**Return:** Error, STD, CR

### Table 2 Number of Inducing Points and Hidden Units

|                | Pendulum | Cartpole | Acrobot | IIWA |
|----------------|----------|----------|---------|------|
| VGP            | 40       | 100      | 250     | 350  |
| RES-VGP        | 40       | 100      | 250     | 350  |
| RES-DBNN       | [50, 50] | [60, 60] | [250, 250] |
| EVGP-IF        | 10       | 60       | 150     | 350  |
| EVGP-JFG       | 10       | 60       | 150     | 350  |

### Table 3 Complexity Comparison

|                | Space  | Time  |
|----------------|--------|-------|
| VGP            | $O(m^2)$ | $O(n^3)$ |
| DBNN           | $O(Lm^2)$ | $O(Ln^3)$ |
| EVGP           | $O(m^2)$ | $O(n^3)$ |
the number of outputs $o$. The time complexity of the DBNN model is governed by the matrix-vector product between the weight matrices and the hidden activation vectors, which is $O(n^2)$. For space complexity, the VGP and EVGP require to store $m$ number of inducing points for each output $o$ as we assume independent models for each output. The DBNN requires to store the weight matrices ($n^2$) for each hidden layer $L$. All models have constant space complexity w.r.t. the training dataset size $|D|$. Furthermore, all models have linear time complexity w.r.t. $|D|$ if we assume that training requires to visit each sample in $D$ at least once.

**Metrics:** For comparison, we used three metrics: 1) prediction error (Error), 2) predicted standard-deviation ($\|\text{STD}\|$), 3) containing ratios (CR). Prediction error is computed as the difference between the dataset values ($y$) and the model expected estimated output ($\mathbb{E}[\hat{y}(i)]$). $\|\text{STD}\|$ is computed as the magnitude of the predicted standard deviation:

$$\text{Error} = \frac{1}{|D|} \sum_{i=1}^{|D|} y(i) - \mathbb{E}[\hat{y}(i)]$$

$$\|\text{STD}\| = \frac{1}{|D|} \sum_{i=1}^{|D|} \|\text{STD}(\hat{y}(i))\|$$

where $|D|$ is the number of samples in the respective dataset. The expected output ($\mathbb{E}[\hat{y}(i)]$) for the EVGP model is equal to $\mu_{\hat{g}}$ in Eq. (4). For the DBNN model, the expectation is estimated using Monte-Carlo.

The containing ratios (CR) are the percentage of values covered by the estimated distribution $\hat{y}$. We consider containing ratios for one, two and three standard deviations (CR-1, CR-2, and CR-3 respectively). We expect the best model to have Error and $\|\text{STD}\|$ close to zero, while best containing ratios will be closer to the (68-95-99.7) rule of standard distributions.

**Results:** Table 4 shows the prediction error and CR scores obtained in the testing dataset. EVGP-IF and EVGP-IFG refers to the use of an IF or IFG prior, respectively. We can observe a considerable improvement on the testing error and CR scores when using EVGPs. EVGPs provided the lowest error and best CR scores. We also see a progressive improvement on the testing error when using more detailed priors. The IFG prior provided lower prediction errors when...
compared with the IF prior. The EVGP-IFG model provided the estimations with the lowest prediction error, with low $\|\text{STD}\|$ and best CR scores. Table 4 also shows that the EVGP model achieved the best performance using fewer number of parameters.

Figure 6 shows a comparison of the prediction error on the test dataset as we increase the number of training samples. For this experiment, we kept the testing dataset fixed while samples were progressively aggregated into the training dataset. The figure shows the mean, max, and min values obtained for four independent runs. Figure 6(a) shows a comparison that includes all models. As expected, the prediction error is reduced as we increase the size of our training dataset. The figure shows that the EVGP provides the most accurate predictions while requiring fewer training samples.

Figure 6(a) also shows how the performance of VGP and RES-VGP plateaus, struggling to take advantage of larger datasets. Although the RES-DBNN performs poorly with small training datasets, the high capacity of the RES-DBNN model allows it to take advantage or large datasets and improve accuracy, reducing the performance gap w.r.t. the EVGP models as more data is available. Thanks to the lower computational time-cost of the RES-DBNN (see Table 3), this model can use a larger set of parameters without incurring in excessive training times.

Figure 6(b) shows a scaled version that only considers the EVGP model with different priors. This figure shows that the IFG prior provides more accurate predictions when compared to the IF prior. In the case of the pendulum, the IFG prior provides a highly accurate model of the system, requiring only a small number of training samples. Figure 6(b) also shows how as training data is aggregated, the accuracy gap between IF and IFG priors is reduced. In the case of the IIWA, we observe that the performance of both IF and IFG is very close. The IIWA proved to be the most challenging system. Even when the DBNN provided CR-3 values above 89% for Pendulum, Cartpole, and Acrobot, the DBNN struggled with the IIWA in terms of containing ratios. The advantage of IFG over IF for the IIWA is mostly observed in the containing ratios in Table 4.

Figure 7 shows a visualization of the EVGP state estimations of the IIWA for multiple time-steps ahead. The multi-step estimations are performed using a standard sequential Monte-Carlo method where particles were used to unroll the EVGP model to predict the state trajectory 45 time-steps ahead [14], [15]. The figure shows that the EVGP is able to provide long-term estimations that follow the dynamics of the physical system. We use the standard deviation to visualize the uncertainty. The figure shows how the uncertainty propagates over time.

Table 5 shows single step error and multi-step estimation error on the testing dataset. The multi-step error is an average of the error in Eq. (11) computed across the estimated state trajectory for 45 time steps ahead. The single step error is equivalent to Eq. (11). The table shows a comparison of the errors between the EVGP and different RES-DBNN models with different activation functions. The table shows that the EVGP provides the lowest single and multi-step error for all models. The table also shows how small single-step errors are accentuated in the multi-step Error. The table also shows that LeakyReLU activation function provides the DBNN model with the lowest error, which is the reason why we choose it for all other experiments.

Understanding the learned model: one of the advantages of incorporating domain knowledge is that the learned model is easy to interpret by the domain expert. For example, in the case of the Acrobot, the value of the parameter $\beta$ can
be visualized to understand and debug what the model has learned. Figure 8 shows the value of $b$ learned with the IF (Fig. 8(a)) and IFG priors (Fig. 8(b)).

We observe in Figure 8 that the learned parameters follow a similar structure given by the prior (see 10). In our experiments, we did not enforce the sparse structure from the priors, i.e. zero parameters in the prior are allowed to have non-zero values in the posterior.

Figure 8(a) shows that when using the IF prior, the EVGP model compensates for the missing gravity information by assigning negative values to $(\dot{q}_1, q_1)$ and $(\dot{q}_2, q_2)$. The reason for this behavior is that $q_1 \approx \sin q_1$ for small $q_1$, however this approximation is no longer valid for large $q_1$. When using IFG priors (Fig. 8(b)), we observe that the model no longer assigns negative values to $(\dot{q}_1, q_1)$ and $(\dot{q}_2, q_2)$. The reason is that IFG provides the values of $\sin(q_1)$ and $\sin(q_1 + q_2)$ which help to model the effect of gravity more accurately.

Figure 9(a) shows the visualization of the mean variational posterior $b$ for the IIWA. Similar to the Acrobot, the IIWA posterior has a familiar diagonal structure that resembles the priors in 10. This observation matches our expectations as we expect the posterior to resemble the prior, which was constructed by placing the pendulum prior on each of the IIWA links. The figure also shows that the lower rows corresponding to $\dot{q}$ have larger values than those corresponding to $q$. This is also expected as most of the inputs should only directly affect the velocity (according to Eq. (7)). Meanwhile, the position can be estimated by performing discrete integration of the velocities, an observation that is embedded in the priors and evidenced by the diagonal structure and low values of the top rows in Figure 9(a).

Figure 9(b) shows in detail the visualization of the posterior for $q \times u$, scaled by the standard deviation of $u$. The figure shows that elements close to the diagonal have larger values, which makes sense as the input torque has the largest influence on its corresponding link and the adjacent links. Figure 9(c) shows the same posterior but constraining the elements off-diagonal to be zero. Figure 9(c) is an example of how an engineer can manipulate the model by visually inspecting the learned posterior. In our methodology, constraining elements of $b$ is straightforward as $b$ is a trainable parameter whose individual elements can be set to predefined values in each training step. In practice we did not see any improvement of accuracy by constraining zero values in the posterior, but this exemplifies the options available to the engineer thanks to the interpretability of the model provided by relationship between the prior $h(x)^T \beta$ and posterior $b$.

V. RELATED WORK

Incorporating prior scientific knowledge in machine-learning algorithms is an ongoing effort that has recently gained increased attention. In [16] LEAP nets are introduced for system identification applied to power systems. The architecture is based on ResNets and uses a set of discrete structural parameters that encode the network topology. Hamiltonian neural networks [17] embeds Hamiltonian mechanics in the architecture of the neural network to ensure that the model follows energy conservation laws. DGNets [18] use a more
general formulation for energy-based modeling than Hamltonian networks and extend the methodology to discrete time. Convolutional neural networks (CNNs) have been used in modeling and simulation applications such as forecasting of sea surface temperature [19] and efficient simulation of the Navier-Stokes equations [20]. Hybrid modeling approaches that combine data and physics based methods are also starting to gain attention in modeling cyber-physical systems [21]. Hybrid models and physics-based loss functions have been introduced in physics-informed neural networks [22], [23] in order to ensure that the trained neural network satisfies energy conservation. In [24] first principles models are combined with machine learning to create deterministic models of process engineering systems. The machine learning model is trained separately to predict the mismatch between a first principles model and the target solution. No training is performed on the first principles model. The solutions of the first principles model and the machine learning model are added to provide the final prediction. In contrast, for the EVGP we model domain knowledge (extracted from simplified first-principles) using a prior probability distribution. This allows us to encode uncertainty in the provided domain-knowledge which in our case is important as we consider domain-knowledge to be imperfect. When we train the EVGP, we use variational inference to obtain a posterior over the parameters given in the prior distribution. The mean of this posterior is represented by \( b \). Data is used to simultaneously tune the parameters in the domain-knowlege function \( h(x) \beta \) and train the data-driven GP.

Gaussian Processes (GP) have been used as a general purpose non-parametric model for system identification and control under uncertainty [12], [25]. Previous work has explored using GPs to include partial model information [26]. In [27] the EGP model described in [5] is used in combination with a simplified physics model in a thermal building modelling application. Our work is based on the GP model with explicit features (EGP) presented in [5]. Variations of this model are commonly used in calibration of computer models [28], [29]. To the best of our knowledge, we are the first to apply variational inference to the EGP model in order to embed simplified physics models and improve scalability.

Despite the advantages of GP models for modeling complex non-linear relations with uncertainty, GPs are computationally expensive. A large bulk of research has focused on improving the computational requirements of GPs. Sparse GP approximation methods are some of the most common approaches for reducing GP computation cost [6]. Bayesian approximation techniques such as Variational Inference provide a rich toolset for dealing with large quantities of data and highly complex models [30], [7]. Variational approximations of a sparse GP have been explored in [7], [11]. In [3] a variational GP model is presented for nonlinear state-space models. In [15], [14], Deep Bayesian Neural Networks (DBNNs) are proposed as an alternative to GPs in order to improve scalability in reinforcement learning problems. Given the popularity of GP models and Variational Inference, there is an increased interest on developing automated variational techniques for these type of models [31], [32].

VI. DISCUSSION

In this paper we wanted to demonstrate how we can use simple concepts from physics that can be scaled and embedded into machine learning models. For this reason we chose the pendulum as representative domain problem, demonstrating how it can be used as building block for 2 DOF systems and expanded to a full 7-DOF manipulator. In a similar manner, the EVGP can be used in other areas such as thermal, electro-magnetic, fluids, among others. These systems are usually modeled using differential equations. The EVGP as used in 5 can be used to approximate discretized versions of these systems. The case study that we showed in section III-B uses an Euler discretization of the differential equation to build the function \( h(x) \beta \), as shown in Eq.7. The presented approach can be extended to other areas by extracting priors from discretized differential equations from the respective area of interest. The extracted priors can then be introduced in the function \( h(x) \beta \) of the EVGP. If detailed domain knowledge is available, this can be directly incorporated in the function \( h(x) \beta \). Furthermore, if some parameters values in \( \beta \) are known with high precision (for example, some of them are known to be zero), it is possible to constrain these values in the posterior mean \( b \) to ensure that the value is not changed after training.

For our analysis of system dynamics, we did not consider measurement or process noise. The presented approach focuses primarily on handling model uncertainty, i.e. uncertainty that derives because of the approximation nature of the learning algorithm [12], [14], [15]. We achieve this by using Gaussian processes and the variational inference framework. Although the EVGP is able to handle observation noise when used in regression problems such as in Fig. 5, the recurrent structure used for system dynamics (Eq. 5) makes the approach sensible to observation noise. This limitation of recurrent GP formulations has been studied in existing literature [33]–[35], but an in-depth analysis is out the scope of this paper. These approaches can be incorporated into the recurrent formulation of the EVGP to explicitly handle observation noise. For example, the direct method from [35] can be incorporated into the EVGP by using moment-matching to estimate denoised states given a sequence of previous noisy measurements. The denoised states can be introduced as inputs to the EVGP and the entire model can be trained using back-propagation.

In this paper, all discussed examples were continuous systems. Hence, we used the squared exponential kernel in all experiments as it encodes smoothness assumptions. The presented approach can be extended to non-continuous systems in several ways: 1) discontinuous functions can be added to the domain-knowledge function \( h(x) \); 2) a kernel that is adapted to model discontinuities can be introduced instead of the squared exponential [36]; 3) a transformation of the input
space can be learned to find a representation that encodes the discontinuity [37].

VII. CONCLUSION
In this paper, we presented the EVGP model, a stochastic data-driven model that is enhanced with domain knowledge extracted from simplified physics. The EVGP is a variational approximation of a Gaussian Process which uses domain knowledge to define the mean function of the prior. The priors provided a rough but simple approximation of the mechanics, informing the EVGP of important structure of the real system. We compared the performance of the EVGP model against purely data-driven approaches and demonstrated improved accuracy and interpretability after incorporating priors derived from simplified Newtonian mechanics. We showed that as we include more detailed priors, the algorithm performance increases. We also showed how the difference between physics enhanced and purely data drive becomes smaller as we increase the availability of training data. We demonstrated that in case of smaller training data sets, the EVGP proved its superiority over the purely data driven approaches. Finally, we illustrate how visualizing the learned parameters can be useful to gain insights into the learned model and hence improve understandability.

APPENDIX A DEFINITION OF MATHEMATICAL ABBREVIATIONS AND PARAMETERS

**Acronyms**

ML Machine Learning.
GP Gaussian Process.
ECP Explicit Gaussian Process.
EVGP Explicit Variational Gaussian Process (our proposed method).
KL Kullback-Leibler (divergence).
ELBO Evidence lower bound.
IF inertia-force prior.
IFG inertia-force-gravity prior.
DBNN Deep Bayesian Neural Network.
RES-DBNN Residual Deep Bayesian Neural Network.
STD Standard Deviation.
DOF Degrees of Freedom.

**Symbols**

⊕ Concatenation operation.
Tr(·) Trace operation.
\(D\) Dataset composed of input/output pair of samples \((x, y)\).
\(\|\cdot\|\) Euclidean norm.

**EVGP functions and variables**

\[ x \in \mathbb{R}^I \]
\[ X \in \mathbb{R}^{D \times I} \]
\[ \hat{x} \]
\[ f(x) \sim GP(0, k(x, x')) \]
\[ K_{xm}, K_{mm}, K_{mx}, K_{xx} \]
\[ g(x) \]
\[ y \]
\[ \hat{y} \]
\[ h(\cdot) \]
\[ H_x, H_m, \hat{H}_x \]
\[ L(\phi) \]
\[ L_{KL} \]
\[ \epsilon_x \sim N(0, \Sigma_x) \]
\[ f_m, X_m \]
\[ \beta \]

**Parameters**

\[ \omega \]
\[ p(\omega) \]
\[ p(\omega|D) \]
\[ p(\epsilon_x) = N(0, \Sigma_{\epsilon_x}) \]
\[ f_m, X_m \]
\[ \beta \]
\[ p(f_m) = N(0, K_{mm}) \]
\[ p(\beta) = N(\mu_{\beta}, \Sigma_{\beta}) \]
\[ p(\phi) \]

\(X_m\) are the inducing points in the GM.
Input of the EVGP (vector of size \(I\)).
Inputs for the EVGP model in Matrix format. Each row of the matrix corresponds to a sample \(x^{(i)}\) extracted from the training dataset.
Testing inputs for the EVGP model in Matrix format.
Gaussian Process with zero mean and kernel \(k\).
Matrices that represent the value of the kernel \(k(\cdot, \cdot)\) evaluated between: \((X, X_m), (X_m, X_m), (X_m, X), (X, X)\), respectively.
Denoised output of the EVGP.
Output of the EVGP.
Explicit function where domain-knowledge is embedded in the EVGP.
Matrices where each row represent the value of the function \(h(\cdot)\) applied to samples in \(X, X_m, \hat{x}\), respectively.
Loss function used to train the EVGP. \(\phi\) are all the trainable parameters in the EVGP.
KL divergence between the posterior and prior distributions for the parameters \(f_m\) and \(\beta\).

Set of model parameters in the EVGP model. These parameters are modeled using probability distributions and include \(f_m, \beta\).
Prior distribution of the EVGP parameters.
Posterior distribution of EVGP parameters given a training dataset.
Inducing points in the GP. Parameters that together with \(h(x)\) encode the Domain-Knowledge provided to the EVGP.
Prior distribution of the parameter \(f_m\) defined as a normal distribution with mean \(\mu_{f_m}\) and covariance matrix \(K_{mm}\).
Prior distribution of the parameter \(\beta\) defined as a normal distribution with mean \(\mu_{\beta}\) and covariance matrix \(\Sigma_{\beta}\). Variational distribution that approximates the posterior \(p(\omega|D)\).
\[ \phi \quad \text{Set of parameters of the variational approximation which include (a, A, b, B, X_m).} \]
\[ p_\phi(f_m) = \mathcal{N}(a, A) \]

Variational distribution of the parameter \( f_m \) defined as a normal distribution with mean \( a \) and covariance matrix \( A \).
\[ p_\phi(\beta) = \mathcal{N}(b, B) \]

Variational distribution of the parameter \( \beta \) defined as a normal distribution with mean \( b \) and covariance matrix \( B \).

**EVGP for physical system modeling**

- \( t \): Time (discrete).
- \( \Delta t \): Sampling interval in seconds.
- \( y_{[t]} \in \mathbb{R}^d \): Output (vector) of the EVGP at time \( t \). The vector is composed by the output of independent EVGP models.
- \( z_{[t]} \): State at time \( t \).
- \( u_{[t]} \): Control input at time \( t \).
- \( x_{[t]} = z_{[t]} \oplus u_{[t]} \): Input of the EVGP at time \( t \).
- \( J, \gamma \): Parameters used to define the structure and values of \( \beta \). They are used only to illustrate how to embed domain-knowledge in \( h(x) \beta \), but they ultimately become part of \( \beta \).
- \( q_i \): Position of the link \( i \).
- \( R_i \): Rotation matrix of the frame attacked to a link \( i \).
- \( v_x, v_y, v_z \): Unitary vectors in axes \( x, y, \) and \( z \), respectively.
- \( v_g \): Unitary vector in the direction of gravity.

**REFERENCES**

1. C. Szegedy et al., “Going deeper with convolutions,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2015, pp. 1–9.
2. A. Adadi and M. Berrada, “Peeking inside the black-box: A survey on explainable artificial intelligence (xai),” IEEE Access, vol. 6, pp. 52138–52160, 2018.
3. R. Frigola, Y. Chen, and C. E. Rasmussen, “Variational gaussian process state-space models,” in Proc. Adv. Neural Inf. Process. Syst., 2014, pp. 3680–3688.
4. D. Marino et al., “Data driven decision support for reliable biomass feedstock preprocessing,” in Proc. Resilience Week, 2017, pp. 97–102.
5. C. E. Rasmussen, “Gaussian processes in machine learning,” in Adv. Lectures Mach. Learn., 2004, pp. 27–29.
6. J. Quilliero-Candel and C. E. Rasmussen, “A unifying view of sparse approximate Gaussian process regression,” J. Mach. Learn. Res., vol. 6, Dec. pp. 1939–1959, 2005.
7. M. Titsias, “Variational learning of inducing variables in sparse Gaussian processes,” in Proc. Artif. Intell. Statist., 2009, pp. 567–574.
8. R. Tedrake and the Drake Development Team, “Drake: A planning, control, and analysis toolbox for nonlinear dynamical systems,” 2016. [Online]. Available: https://drake.mit.edu
9. D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014, arXiv:1412.6980.
10. E. Snelson and Z. Ghahramani, “Sparse gaussian processes using pseudo-inputs,” in Proc. Adv. Neural Inf. Process. Syst., 2006, pp. 1257–1264.
11. J. Hensman, N. Fusi, and N. D. Lawrence, “Gaussian processes for big data,” in Proc. 29th Conf. Uncertainty Artif. Intell., pp. 282–290, 2013.
12. M. Deisenroth and C. E. Rasmussen, “Pilco: A model-based and data-efficient approach to policy search,” in Proc. 28th Int. Conf. Mach. Learn., 2011, pp. 465–472.
13. D. P. Kingma, T. Salimans, and M. Welling, “Variational dropout and the local reparameterization trick,” in Proc. Adv. Neural Inf. Process. Syst., 2015, pp. 2575–2583.
14. D. Marino and M. Manic, “Modeling and planning under uncertainty using deep neural networks,” IEEE Trans. Ind. Informat., vol. 15, no. 8, pp. 4442–4454, Aug. 2019.
15. Y. Gal, R. T. McAllister, and C. E. Rasmussen, “Improving PILCO with Bayesian neural network dynamics models,” in Proc. Data-Efficient Mach. Learn. Workshop, ICML, 2016, pp. 1–7.
16. B. Donon et al., “Leap nets for system identification and application to nonlinear systems,” Neurocomputing, vol. 416, pp. 316–327, 2020.
17. S. Greydanus, M. Dzamba, and J. Yosinski, “Hamiltonian neural networks,” in Advances in Neural Information Processing Systems, vol. 32, 2019. [Online]. Available: https://proceedings.neurips.cc/paper/2019/file/26cd8ecadce0d4efd6cc725cbd1f8-Paper.pdf
H. Mohammadi, P. Challenor, M. Goodfellow, and D. Williamson, “Emulating computer models with step-discontinuous outputs using gaussian processes,” arXiv preprint arXiv:1903.02071, 2019.

R. Calandra, J. Peters, C. E. Rasmussen, and M. P. Deisenroth, “Manifold gaussian processes for regression,” in Proc. Int. Joint Conf. Neural Netw., 2016, pp. 3338–3345.

DANIEL L. MARINO received the B.Eng. in automation engineering from La Salle University, Colombia, in 2015. He is currently a Research Assistant and a Doctoral Student with Virginia Commonwealth University, Richmond, VA, USA. His research interests include stochastic modeling, deep learning, and optimal control.

MILOS MANIC (Fellow, IEEE) received the Dipl. Ing. and M.S. degrees in electrical engineering and computer science from the University of Niš, Niš, Serbia, in 1991 and 1997 respectively, and the Ph.D. degree in computer science from the University of Idaho, in 2003. He is a Professor with the Computer Science Department and the Director of VCU Cybersecurity Center, Virginia Commonwealth University, Richmond, VA, USA. He completed more than 40 research grants in data mining and machine learning applied to cyber security, critical infrastructure protection, energy security, and resilient intelligent control. He has given more than 40 invited talks around the world, authored more than 200 refereed articles in international journals, books, and conferences, holds several U.S. patents.

He was the recipient of the 2018 R&D 100 Award for Autonomic Intelligent Cyber Sensor, one of the top 100 science and technology worldwide innovations in 2018, the IEEE IES 2019 Anthony J. Hornfeck Service Award, the 2012 J. David Irwin Early Career Award, and the 2017 IEM Best Paper Award. He is an inductee of the US National Academy of Inventors (class of 2019) and a Fellow of Commonwealth Cyber Initiative specialty in AI & Cybersecurity. He was an Associate Editor for the TRANSACTIONS ON INDUSTRIAL INFORMATICS, the OPEN JOURNAL OF INDUSTRIAL ELECTRONICS SOCIETY, and is an IES Officer and a Senior AdCom Member. He was an Associate Editor for the TRANSACTION ON INDUSTRIAL ELECTRONICS, was the Founding Chair of IEEE IES Technical Committee on Resilience and Security in Industry, and the General Chair of the IEEE IECON 2018, the IEEE HSI 2019.