Controlled Gaussian Process Dynamical Models with Application to Robotic Cloth Manipulation

Fabio Amadio[1], Juan Antonio Delgado-Guerrero[2], Adrià Colomé[2], and Carme Torras[2]

***Abstract*—Over the last years, robotic cloth manipulation has gained relevance within the research community. While significant advances have been made in robotic manipulation of rigid objects, the manipulation of non-rigid objects such as cloth garments is still a challenging problem. The uncertainty on how cloth behaves often requires the use of model-based approaches. However, cloth models have a very high dimensionality. Therefore, it is difficult to find a middle point between providing a manipulator with a dynamics model of cloth and working with a state space of tractable dimensionality. For this reason, most cloth manipulation approaches in literature perform static or quasi-static manipulation. In this paper, we propose a variation of Gaussian Process Dynamical Models (GPDMs) to model cloth dynamics in a low-dimensional manifold. GPDMs project a high-dimensional state space into a smaller dimension latent space which is capable of keeping the dynamic properties. Using such approach, we add control variables to the original formulation. In this way, it is possible to take into account the robot commands exerted on the cloth dynamics. We call this new version Controlled Gaussian Process Dynamical Model (CGPDM). Moreover, we propose an alternative parametric structure for the model, that is richer than the one employed in previous GPDM realizations. The modeling capacity of our proposal has been tested in both a simulated and a real scenario, where CGPDM proved to be capable of generalizing over a wide range of movements and correctly predicting the cloth motions obtained by previously unseen sequences of control actions.**

I. Introduction

Robotic cloth manipulation has a wide range of applications, from textile industry to assistive robotics [1], [2], [3], [4], [5], [6]. However, the complexity of cloth behaviour results in a high uncertainty in the state transition given a certain action. This uncertainty is what makes cloth manipulation much harder than manipulating rigid objects. Intuitively, learning the cloth's dynamics is the solution to reduce such uncertainty. In literature, we can find several cloth models that simulate the internal cloth state [7], [8], [9]. They represent cloth as a mesh of material points, and simulate their behaviour taking into account physical constraints. However, fitting those models to real data can be a complex task. Moreover, such models need not only to behave similarly enough to the cloth garment, but to have a tractable dimension, for computational reasons. As an example, an 8×8 mesh representing a square towel results in a 192-dimensional manifold. Such dimensionality is unmanageable, not only in terms of computational costs, but also for building a tractable state-action space policy. Such is the case of [10], where simulated results are obtained after hours of computations. Hence, Dimensionality Reduction (DR) methods are necessary.

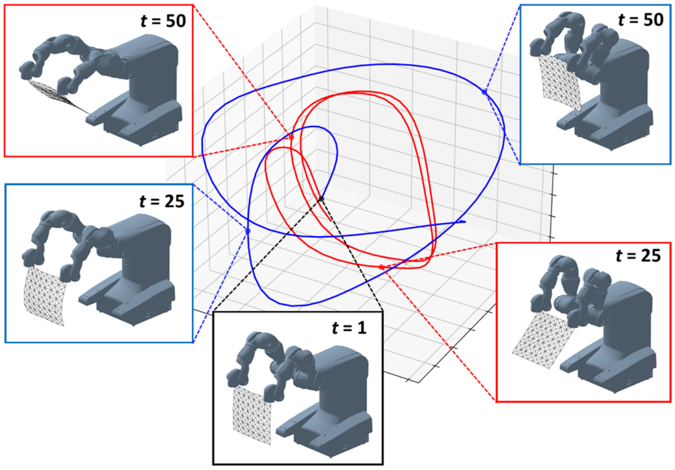

Fig. 1: *The same CGPDM predicts two distinct latent state trajectories in response to two different sequences of unseen control actions. Each point in the latent space has associated a particular configuration of the modeled cloth.*

In [11], linear DR techniques were used for learning cloth manipulation by biasing the latent space projection with each execution's performance. Nonlinear methods, such as Gaussian Process Latent Variable Models (GPLVM) [12] have also been applied for this purpose. In [13], GPLVM was employed to project task-specific motor-skills of the robot onto a much smaller state representation, whereas in [14] GPLVM was also used to represent a robot manipulation policy in a latent space, taking contextual features into account. However, these approaches focus the dimensionality reduction in the robot action characterization, rather than in the manipulated object's dynamics. Instead, in [15] the same DR technique was applied to learn a latent representation of the cloth state from point clouds taken by depth sensors. However, such approach did not consider the dynamics of the cloth handling task, and its applicability is limited to a quasi-static manipulation.

In this paper, we assume to have recorded data from several cloth motions, as a time-varying mesh of points. This data will often come from the processed RGB-D data of a camera. To fit such data into a tractable dynamical model, we consider Gaussian Process Dynamical Models (GPDM), first introduced in [16], which are an extension of the GPLVM structure

This work was partially developed in the context of the project CLOTHILDE ("CLOTH manIpulation Learning from DEmonstrations"), which has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Advanced Grant agreement No 741930).

[1] Department of Information Engineering, Università di Padova, Italy amadiofa@dei.unipd.it

[2] Institut de Robòtica i Informàtica Industrial (IRI), CSIC-UPC, Spain [jdelgado, acolome, torras]@iri.upc.edu

explicitly oriented to the analysis of high-dimensional time series. GPDMs have been applied in several different fields, from human motion tracking [17], [18] to dynamic texture modeling [19]. In the context of cloth manipulation, GPDMs were adopted in [20] to learn a latent model of the dynamics of a cloth handling task. However, this framework, as it stands, lacks in its structure a fundamental component to correctly describe the dynamics of a system, namely control actions, limiting generalization capacity.

Therefore, we propose here an extension of the GPDM structure, that takes into account the influence of external control actions on the modeled dynamics. We call it *Controlled Gaussian Process Dynamical Models* (CGPDMs). In this new version, control actions directly affect the dynamics in the latent space. Thus, a CGPDM, trained on a sufficiently diverse set of interactions, is able to predict the effects of control actions never experienced before inside a space of reduced dimensions, and then reconstruct high-dimensional motions by projecting the latent state trajectories into the observation space. We tested the applicability of such solution in both a simulated and a real cloth manipulation scenario. CGPDM has proved capable of fitting different types of cloth movement, and predict the results of control actions never seen during training. As an example, in Fig. 1, we show the trajectories predicted by the same CGPDM subject to two different sequences of control actions. Note how the two trajectories, starting from the same latent state, are driven into different regions of the latent space, where different cloth poses are reconstructed.

Previous GPDM applications [16], [17], [18] were based on Gaussian Processes (GPs) equipped with isotrophic squared exponential (SE) and homogeneous linear kernel functions. The limited number of hyper-parameters of such GPs could limit the generalization properties of the overall model. In this work, we study the effects that a richer parameterization could possibly have on prediction accuracy and generalization capabilities. In particular, we compare two distinct CGPDM implementations: (i) the *lowly-parameterized* CGPDM is a direct extension of the standard GPDM's maps to the case of a controlled dynamical system; (ii) the *highly-parameterized* CGPDM, instead, is characterized by the presence of a higher number of hyper-parameters. In the last case, we employed SE kernels with automatic relevance determination (ARD) [21] and inhomogeneous linear kernels, together with tunable scaling factors in the dynamical map. Despite these kernels are commonly used in GP regression, they have not yet been applied in the context of GPDM. In fact, to the best of our knowledge, only [19] proposed to use kernels different from the naive ones adopted in the original implementation [16], but it introduced a multi-kernel structure only for the GPs modeling the dynamics, without modifying the latent map.

The remainder of the paper is structured as follows. Preliminaries are given in Sec. II. CGPDM framework is presented in Sec. III. Sec. IV treats its training and explains how to make predictions, in both latent and observation spaces. Experimental results on a simulated robotic cloth manipulation scenario are reported in Sec. V, while the application of CGPDM on a real setup is described in Sec. VI. Sec. VII draws the conclusions.

II. Preliminaries: From GPs to GPDMs

GPs [22] are the infinite-dimensional generalization of multivariate Gaussian distributions. They are defined as infinite-dimension stochastic processes such that, for any finite set of input locations $\mathbf{x}_1, ..., \mathbf{x}_n$, the random variables $f(\mathbf{x}_1), ..., f(\mathbf{x}_n)$ have joint Gaussian distributions. A GP is defined by its mean function $m(\mathbf{x})$ and kernel $k(\mathbf{x}, \mathbf{x}')$, that must be symmetric and positive semi-definite. Usually GPs are indicated as

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')).$$

GPs can be used for regression models of the form $y = f(\mathbf{x}) + \varepsilon$, with ε an i.i.d. Gaussian noise, as they provide closed formulae to predict new target y^*, given new input $\mathbf{x}^*$. GP regression has been widely applied as a data-driven tool for dynamical system identification [23], usually describing each state by its own GP. Nevertheless, such approach is unfeasible when dealing with high-dimensional systems, due to the high computational demands. Thus, DR strategies must be included.

GPLVMs [12], [24] emerged as feature extraction methods that can be used as multiple-output GP regression models. These models, under a DR perspective, associate and learn low-dimensional representations of higher-dimensional observed data, assuming that observed variables are determined by the latent ones. Finally, GPLVMs provide, as a result of an optimization, a mapping from the latent space to the observation space, together with a set of latent variables representing the observed values. However, GPLVMs are not explicitly thought to deal with data from time series, where observations at different time steps are connected by some form of dynamics.

Thus, [16] first introduced Gaussian Process Dynamical Models (GPDM), an extension of the GPLVM structure explicitly oriented to the analysis of high-dimensional time series. A GPDM entails essentially two stages: (i) a latent mapping that projects high-dimensional observations to a low-dimensional latent space (1); (ii) a discrete-time Markovian dynamics that captures the evolution of the time series inside the reduced latent space (2). GPs are used to model both the latent map, as in GPLVMs, and the undergoing dynamics transition function. GPDMs are then defined by

$$\boldsymbol{y}_t = g(\boldsymbol{x}_t) + \boldsymbol{n}_{y,t}, \tag{1}$$

$$\boldsymbol{x}_{t+1} = h(\boldsymbol{x}_t) + \boldsymbol{n}_{x,t}, \tag{2}$$

where $\boldsymbol{y}_t$ is the high-dimensional observation vector and $\boldsymbol{x}_t$ represents the latent state, at time step t. Here, $\boldsymbol{n}_{y,t}$ and $\boldsymbol{n}_{x,t}$ are two zero-mean isotropic white Gaussian noise processes, while g and h are two unknown functions.

III. Controlled GPDM

Let us consider a system governed by an unknown dynamics. At each time step t, it is possible to influence it by applying control actions $\boldsymbol{u}_t \in \mathbb{R}^E$ and getting an observation $\boldsymbol{y}_t \in \mathbb{R}^D$. For high-dimensional observation spaces, it could be unfeasible to directly model the evolution of a sequence of observations in response to a series of inputs. For instance, in the case of a robot moving a piece of cloth, we can consider as control actions $\boldsymbol{u}_t$ the instantaneous movement of the end-effector, while the observations $\boldsymbol{y}_t$ could be the coordinates of a mesh of material points, representing the cloth configuration. In this context, it could be convenient to capture the dynamics of the system in a low-dimensional latent space $\mathbb{R}^d$, with $d << D$. Let $\boldsymbol{x}_t \in \mathbb{R}^d$ be the latent state associated with $\boldsymbol{y}_t$. We propose to use a variation of the GPDM that keeps into account the influence of control actions, while maintaining the dimensionality reduction properties of the original model. We call it *Controlled Gaussian Process Dynamical Model* (CGPDM).

A CGPDM consists of a latent map (3) projecting observations $\boldsymbol{y}_t$ into latent states $\boldsymbol{x}_t$, and a dynamics map (4) that describes the evolution of the latent state $\boldsymbol{x}_t$, subject to $\boldsymbol{u}_t$.

$$\boldsymbol{y}_t = g(\boldsymbol{x}_t) + \boldsymbol{n}_{y,t}, \tag{3}$$

$$\boldsymbol{x}_{t+1} - \boldsymbol{x}_t = h(\boldsymbol{x}_t, \boldsymbol{u}_t) + \boldsymbol{n}_{x,t}. \tag{4}$$

Differently from original GPDM (2), control actions have influence on the CGPDM transition function (4). On the other hand, latent map (3) is identical to (1) because control actions should not affect the dimensionality reduction process. Note that we consider $\boldsymbol{x}_{t+1} - \boldsymbol{x}_t$ to be the output of the CGPDM dynamic map, [17] suggested that this choice can help to improve the smoothness of latent trajectories. In the following, we report how to model (3) and (4) by means of GPs.

A. Latent variable mapping

Each component of the observation vector $\boldsymbol{y}_t = [y_t^{(1)}, \ldots, y_t^{(D)}]^T$ can be modeled a priori as a zero-mean GP that takes as input $\boldsymbol{x}_t$, for $t = 1, \ldots, N$.

Let $Y = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_N]^T \in \mathbb{R}^{N \times D}$ be the matrix that collects the set of N observations, and $X = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]^T \in \mathbb{R}^{N \times d}$ be the matrix of associated latent states. We denote with $Y_{:,j}$ the vector containing the j-th components of all the N observations.

Then, if we assume that the D observation components are independent variables, the probability over the whole set of observations can be expressed by the product of the D GPs. In addition, if we choose the same kernel function $k_y(\cdot, \cdot)$ for each GP, differentiated only through a variable scaling factor $w_{y,j}^{-2}$, with $j = 1, \ldots, D$, the joint likelihood over the whole set of observations is given by

$$p(Y|X) = \frac{|W_y|^N \exp\left(-\frac{1}{2}\mathrm{tr}\left((K_y(X))^{-1} Y W_y^2 Y^T\right)\right)}{\sqrt{(2\pi)^{ND}|K_y(X)|^D}}, \tag{5}$$

where $W_y = \mathrm{diag}(w_{y,1}, \ldots, w_{y,D})$, $K_y(X)$ is the covariance matrix defined element-wise by $k_y(\cdot, \cdot)$. Independence assumption may be relaxed by applying, for instance, coregionalization models [25], at the cost of greater computational demands.

In previous works on GPDMs [16], [17], [18], the GPs of the latent map were equipped with an isotrophic SE kernel,

$$k'_y(\boldsymbol{x}_r, \boldsymbol{x}_s) = \exp\left(-\frac{\beta_1}{2}||\boldsymbol{x}_r - \boldsymbol{x}_s||^2\right) + \beta_2^{-1}\delta(\boldsymbol{x}_r, \boldsymbol{x}_s), \tag{6}$$

with parameters β_1 and β_2 ($\delta(\boldsymbol{x}_r, \boldsymbol{x}_s)$ is the Kronecker delta).

Instead here, we adopt the richer ARD structure for the SE kernel, characterized by the presence of length-scales that can weight differently each component of the latent state:

$$k_y(\boldsymbol{x}_r, \boldsymbol{x}_s) = \exp\left(-||\boldsymbol{x}_r - \boldsymbol{x}_s||_{\Lambda_y^{-1}}\right) + \sigma_y^2\delta(\boldsymbol{x}_r, \boldsymbol{x}_s). \tag{7}$$

$\Lambda_y^{-1} = \mathrm{diag}(\lambda_{y,1}^{-2}, \ldots, \lambda_{y,D}^{-2})$ is a positive definite diagonal matrix, which weights the norm used in the SE function, and σ_y^2 is the variance of the isotropic noise in (3). The trainable hyper-parameters of the latent map model are then $\theta_y = \{w_{y,1}, \ldots, w_{y,D}, \lambda_{y,1}, \ldots, \lambda_{y,D}, \sigma_y\}$.

B. Dynamics mapping

Similarly to Sec. III-A, we can model a priori each component of the latent state difference $\boldsymbol{x}_{t+1} - \boldsymbol{x}_t = [x_{t+1}^{(1)} - x_t^{(1)}, \ldots, x_{t+1}^{(d)} - x_t^{(d)}]^T$ as a zero-mean GP that takes as input the pair $(\boldsymbol{x}_t, \boldsymbol{u}_t)$, for $t = 1, \ldots, N-1$.

Let $X = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]^T \in \mathbb{R}^{N \times d}$ be the matrix collecting the set of N latent states, we can denote by $X_{r:s,i}$ the vector of the i-th components from time step r to time step s, with $r, s = 1, \ldots, N$. We indicate the vector of differences between consecutive latent states along their i-th component with $\Delta_{:,i} = (X_{2:N,i} - X_{1:N-1,i}) \in \mathbb{R}^{N-1}$. $\Delta = [\Delta_{:,1}, \ldots, \Delta_{:,d}] \in \mathbb{R}^{(N-1) \times d}$ is the matrix that collects differences along all the components. Finally, we compactly represent the GP input of the dynamic model as $\tilde{\boldsymbol{x}}_t = [\boldsymbol{x}_t^T, \boldsymbol{u}_t^T]^T \in \mathbb{R}^{d+E}$, and refer to the the matrix collecting $\tilde{\boldsymbol{x}}_t$ for $t = 1, \ldots, N-1$ with $\tilde{X} = [\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_{N-1}]^T \in \mathbb{R}^{(N-1) \times (d+E)}$.

With similar assumptions to the ones made for the latent map, and denoting the common kernel function for all the GPs with k_x, and the different scaling factors with $w_{x,1}$, for $i = 1, \ldots, d$, the joint likelihood is given by

$$p(\Delta|\tilde{X}) = \frac{|W_x|^{N-1} \exp\left(-\frac{1}{2}\mathrm{tr}\left(\left(K_x(\tilde{X})\right)^{-1} \Delta W_x^2 \Delta^T\right)\right)}{\sqrt{(2\pi)^{(N-1)d}|K_x(\tilde{X})|^d}}, \tag{8}$$

where $W_x = \mathrm{diag}(w_{x,1}, \ldots, w_{x,d})$ and $K_x(\tilde{X})$ is the covariance matrix defined element-wise by $k_x(\cdot, \cdot)$.

In standard GPDM [16], dynamic mapping GPs have been proposed with constant scaling factors $w_{x,i} = 1$ for $i = 1, \ldots, d$, and equipped with the simple kernel resulting from the sum of an isotrophic SE and an homogeneous linear function, characterized only by four trainable parameters:

$$k'_x(\tilde{\boldsymbol{x}}_r, \tilde{\boldsymbol{x}}_s) = \alpha_1 \exp\left(-\frac{\alpha_2}{2}||\tilde{\boldsymbol{x}}_r - \tilde{\boldsymbol{x}}_s||^2\right) + \ldots \tag{9}$$
$$\cdots + \alpha_3 \tilde{\boldsymbol{x}}_r^T \tilde{\boldsymbol{x}}_s + \alpha_4^{-1}\delta(\tilde{\boldsymbol{x}}_r, \tilde{\boldsymbol{x}}_s).$$

Analogously to the latent mapping, we decided to adopt a more complex kernel function, detailed in the following,

$$k_x(\tilde{\boldsymbol{x}}_r, \tilde{\boldsymbol{x}}_s) = \exp\left(-||\tilde{\boldsymbol{x}}_r - \tilde{\boldsymbol{x}}_s||_{\Lambda_x^{-1}}\right) + \ldots \tag{10}$$
$$\cdots + [\tilde{\boldsymbol{x}}_r^T, 1]\Phi[\tilde{\boldsymbol{x}}_s^T 1]^T + \sigma_x^2\delta(\tilde{\boldsymbol{x}}_r, \tilde{\boldsymbol{x}}_s).$$

$\Lambda_x^{-1} = \mathrm{diag}(\lambda_{x,1}^{-2}, \ldots, \lambda_{x,d+E}^{-2})$ is a positive definite diagonal matrix, which weights the norm used in the SE component of the kernel. Also $\Phi = \mathrm{diag}(\phi_1^2, \ldots, \phi_{d+E+1}^2)$ is a positive definite diagonal matrix that describes the linear component. σ_x^2 is the variance of the isotropic noise in (4). In comparison to (9), the adopted kernel weights differently the various components of the input in both SE and linear part, where the GP input is also extended as $[\tilde{\boldsymbol{x}}_s^T, 1]^T$. The trainable hyper-parameters of the dynamic map model are then $\theta_x = \{w_{x,1}, \dots, w_{x,d}, \lambda_{x,1}, \dots, \lambda_{x,d}, \phi_1, \dots, \phi_{d+E+1}, \sigma_x\}$.

In the following we will refer to the proposed CGPDM structure with kernels (7) and (10), and trainable scaling factors in the dynamical map, as *highly-parameterized* CGPDM. On the contrary, *lowly-parameterized* CGPDM will indicate the version that straightforwardly extends the standard GPDM structure from [16], using its same kernels, (6) and (9), and constant scaling factors. Although kernels such (7) and (10) are commonly adopted in GP regression literature [22], to the best of our knowledge, they have not been tested before in the context of GPDM. Furthermore, also the adoption of trainable scaling factors constitutes a novelty for this kind of models.

IV. CGPDM Training and Predictions

Training the CGPDM entails using numerical optimization techniques to estimate the unknowns in the model, i.e., latent states X and the hyper-parameters θ_x, θ_y. Latent coordinates X are initialized by means of PCA [26], selecting the first d principal components of Y. A natural approach for training CGPDMs is to maximize the joint log-likelihood $\ln p(Y|X) + \ln p(\Delta|\tilde{X})$ w.r.t. $\{X, \theta_x, \theta_y\}$. As regards numerical optimization, we used the L-BFGS algorithm [27].

Hence, the overall loss will be given by

$$\mathcal{L} = \mathcal{L}_y + \mathcal{L}_x, \tag{11}$$

where

$$\mathcal{L}_y = \frac{D}{2}\ln|K_y(X)| + \frac{1}{2}\mathrm{tr}(K_y(X)^{-1} Y W_y^2 Y^T) - N\ln|W_y|,$$

$$\mathcal{L}_x = \frac{d}{2}\ln|K_x(\tilde{X})| + \frac{1}{2}\mathrm{tr}(K_x(\tilde{X})^{-1} \Delta W_x^2 \Delta^T) - (N-1)\ln|W_x|.$$

A trained CGPDM can be used to fulfill two different purposes: (i) map a given new latent state $\boldsymbol{x}_t^*$ to the corresponding $\boldsymbol{y}_t^*$ in observation space, (ii) predict the evolution of the latent state at the next time step $\boldsymbol{x}_{t+1}^*$, given $\boldsymbol{x}_t^*$ and a certain control $\boldsymbol{u}_t^*$. The two processes, together, can predict the observations produced by a given series of control actions.

A. Latent prediction

Given $\boldsymbol{x}_t^*$, its corresponding observation $\boldsymbol{y}_t^*$ is distributed as $p(\boldsymbol{y}_t^*|\boldsymbol{x}_t^*, X, \theta_y) = \mathcal{N}(\mu_y(\boldsymbol{x}_t^*), v_y(\boldsymbol{x}_t^*) W_y^{-2})$, with

$$\mu_y(\boldsymbol{x}_t^*) = Y^T K_y(X)^{-1} \boldsymbol{k}_y(\boldsymbol{x}_t^*, X)$$

$$v_y(\boldsymbol{x}_t^*) = k_y(\boldsymbol{x}_t^*, \boldsymbol{x}_t^*) - \boldsymbol{k}_y(\boldsymbol{x}_t^*, X)^T K_y(X)^{-1} \boldsymbol{k}_y(\boldsymbol{x}_t^*, X),$$

where $\boldsymbol{k}_y(\boldsymbol{x}_t^*, X) = [k_y(\boldsymbol{x}_t^*, \boldsymbol{x}_1), \dots, k_y(\boldsymbol{x}_t^*, \boldsymbol{x}_N)]^T$.

B. Dynamics prediction

Given $\boldsymbol{x}_t^*$ and $\boldsymbol{u}_t^*$, let's define $\tilde{\boldsymbol{x}}_t^* = [\boldsymbol{x}_t^{*T}, \boldsymbol{u}_t^{*T}]^T$. The probability density of the latent state at the next time step $\boldsymbol{x}_{t+1}^*$ is $p(\boldsymbol{x}_{t+1}^*|\tilde{\boldsymbol{x}}_t, X, \theta_x) = \mathcal{N}(\mu_x(\boldsymbol{x}_t^*), v_x(\boldsymbol{x}_t^*) W_x^{-2})$, with

$$\mu_x(\boldsymbol{x}_t^*) = \boldsymbol{x}_t^* + \Delta^T K_x(\tilde{X})^{-1} \boldsymbol{k}_x(\tilde{\boldsymbol{x}}_t^*, \tilde{X}),$$

$$v_x(\boldsymbol{x}_t^*) = k_x(\tilde{\boldsymbol{x}}_t^*, \tilde{\boldsymbol{x}}_t^*) - \boldsymbol{k}_x(\tilde{\boldsymbol{x}}_t^*, \tilde{X})^T K_x(\tilde{X})^{-1} \boldsymbol{k}_x(\tilde{\boldsymbol{x}}_t^*, \tilde{X}),$$

where $\boldsymbol{k}_x(\tilde{\boldsymbol{x}}_t^*, \tilde{X}) = \left[k_x(\tilde{\boldsymbol{x}}_t^*, \tilde{\boldsymbol{x}}_1), \dots, k_x(\tilde{\boldsymbol{x}}_t^*, \tilde{\boldsymbol{x}}_{N-1})\right]^T$.

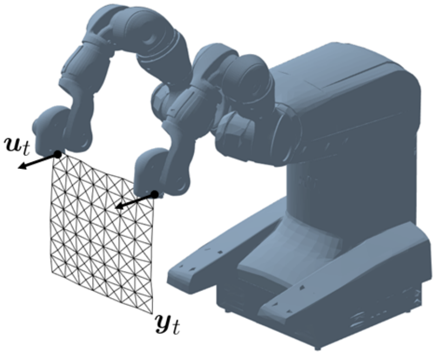

Fig. 2: *Simulated setup for cloth manipulation with bimanual robot. The cloth is positioned in its starting configuration.*

C. Trajectory prediction

Starting from an initial latent state $\boldsymbol{x}_1^*$, one can predict the evolution of the system over a desired horizon of length N_d, when subject to a given sequence of control actions $\boldsymbol{u}_1^*, \dots, \boldsymbol{u}_{N_d-1}^*$. At each time step $t = 1, \dots, N_d - 1$, $\boldsymbol{x}_{t+1}^*$ can be sampled from the normal distribution $p(\boldsymbol{x}_{t+1}^*|\tilde{\boldsymbol{x}}_t, X, \theta_x)$ defined in Sec. IV-B. Hence, the generated trajectory in the latent space $\boldsymbol{x}_1^*, \dots, \boldsymbol{x}_{N_d}^*$ can be mapped into the associated predicted sequences of observations $\boldsymbol{y}_1^*, \dots, \boldsymbol{y}_{N_d}^*$ by mean of $p(\boldsymbol{y}_t^*|\boldsymbol{x}_t^*, X, \theta_y)$, defined in Sec. IV-A.

V. Experiment with Simulated Cloth

Initially, we validated CGPDM in a simulated scenario, consisting of a bimanual robot that moves a piece of cloth by holding its two upper corners. The cloth is modeled as an 8×8 mesh of material points. The two points in the upper corners are assumed to be attached to the two robot's end-effectors, while the other points move following the dynamical model proposed in [28]. In this context, the observation vector is given by the Cartesian coordinates of all the points in the mesh (measured in meters); hence $\boldsymbol{y}_t \in \mathbb{R}^D$ with $D = 192$. We assume that the two end-effectors can be controlled exactly in the operational space. In this system, the controls acting at time step t are the differences between position commands at instant $t+1$ and t (measured in meters); so $\boldsymbol{u}_t \in \mathbb{R}^E$ with $E = 6$. The overall setup is shown in Fig. 2. The objective of the experiment is to learn the high-dimensional cloth dynamics using CGPDM, in order to make predictions about cloth movements in response to sequences of actions that were not seen during training. We adopted a latent space of dimension $d = 3$, resulting in a dimensionality reduction factor of $D/d = 64$. Such high-dimensional task would be unfeasible to model by standard GP regression without DR. CGPDMs was implemented in Python*, employing PyTorch library [29]. We aim to evaluate how prediction accuracy is affected by

- the number of observation sequences used for training,
- the range of the cloth movements,
- the use of *lowly* or *highly* CGPDM (defined in Sec. III).

A. Data collection

Data were obtained by recording mesh trajectories associated with several types of cloth oscillation, obtained by applying

*Code publicly available at https://bitbucket.org/fabio_ama/gpdm_lib.

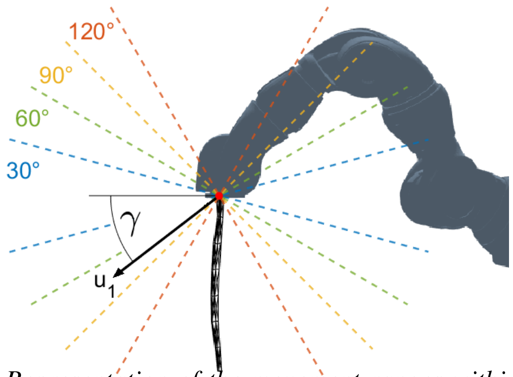

Fig. 3: *Representation of the movement ranges within which control parameter γ was sampled during data collection.*

different sequences of control actions. All the considered trajectories start from the same cloth configuration and last 5 seconds. Observations were recorded at 20 Hz, hence $N = 100$ total number of steps for each sequence.

Let $\boldsymbol{u}_t = \left[{}^r\delta_t^X, {}^r\delta_t^Y, {}^r\delta_t^Z, {}^l\delta_t^X, {}^l\delta_t^Y, {}^l\delta_t^Z\right]^T$, where ${}^r\delta_t^X$, ${}^r\delta_t^Y$ and ${}^r\delta_t^Z$ (${}^l\delta_t^X$, ${}^l\delta_t^Y$ and ${}^l\delta_t^Z$) indicate the displacement of the right (left) end-effector position along the three Cartesian axes, between step t and $t+1$. Specifically, we used the same commands for both end-effectors. Denoting with $\left(\delta_t^X, \delta_t^Y, \delta_t^Z\right)$ the common displacements, the applied $\boldsymbol{u}_t$ were given by,

$$\delta_t^X = 0,\ \delta_t^{[Y,Z]} = A \cdot \cos(2\pi f_Y t)\,[-\cos(\gamma), \sin(\gamma)] \quad (12)$$

Such controls make the end-effectors oscillate on the Y-Z plane of the operational space. The maximum displacement is regulated by A, that we set to 0.01 meters. Parameter γ can be interpreted as the inclination of $\boldsymbol{u}_1$ w.r.t. the horizontal, and it loosely defines a direction of the oscillation. f_Y and f_Z define the frequencies of the oscillations along Y and Z axes. If they are similar, the end-effectors move mostly along the direction defined by γ, if not, they swipe in a broader space.

In order to obtain a heterogeneous set of trajectories for the composition of training and test sets, we collected several movements obtained by choosing in a random fashion the control parameters γ, f_Y and f_Z. Angles γ were uniformly sampled inside a variable range $[-\frac{R}{2}, \frac{R}{2}]$ (deg); in the following, we indicate this range with the amplitude of its angular area, R (deg). Instead, frequencies f_Y and f_Z were uniformly sampled inside the fixed interval [0.3, 0.6] (Hz).

We considered four movement ranges of increasing size, namely $R \in \{30°, 60°, 90°, 120°\}$ (see Figure 3), and collected a specific data-set $\mathcal{D}_R$ associated with each range. Every set contains 50 cloth trajectories obtained by applying control actions of the form (12) with 50 different random choices for parameters γ, f_Y and f_Z. From each $\mathcal{D}_R$, 10 trajectories were extracted and used as test sets $\mathcal{D}_R^{test}$ for the corresponding movement range, while several training sets $\mathcal{D}_R^{train}$ were built by randomly picking from the remaining sequences.

B. Model training

The objective of the experiment is to evaluate CGPDM prediction accuracy in different movement ranges, and for different amounts of training data. Also, we want to observe if the use of *lowly-parameterized* or *highly-parameterized* CGPDMs yield a substantial difference in terms of accuracy.

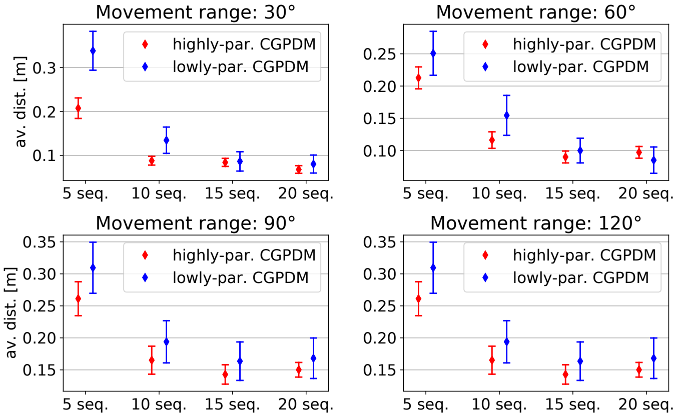

Fig. 4: *Average distances (with 95% C.I.) between true and predicted mesh points obtained by different CGPDM setups.*

Consequently, for each considered movement range R, we trained different *highly-parameterized* CGPDMs employing an increasing number of sequences randomly picked from $\mathcal{D}_R^{train}$. Specifically, we used 10 different combinations of 5, 10, 15 and 20 sequences for each range. In this way, we were able to reduce the dependencies on the specific training trajectories employed, and to average prediction accuracy over different possible sets of training data. For each such model, a *lowly-parameterized* CGPDM was trained on exactly the same data.

C. Results

We used each learned CGPDM to predict the cloth movements when subject to the test control actions relative to the associated $\mathcal{D}_R^{test}$, for $R \in \{30°, 60°, 90°, 120°\}$. Let $\boldsymbol{y}_t^{(R,k)}$ and $\boldsymbol{u}_t^{(R,k)}$ denote the observation and control action, at time step t, of the k-th test trajectory in $\mathcal{D}_R^{test}$, with $k = 1, \dots, 10$.

For every considered range R, one can follow the procedure described in Sec. IV-C and employ the trained CGPDMs to predict the trajectories resulting from the application of control action sequences $\{\boldsymbol{u}_t^{(R,k)}\}_{t=1}^{N-1}$, for $k = 1, \dots, 10$. Let $\boldsymbol{x}_t^{*(R,k)}$ be the predicted latent state at time t, and $\boldsymbol{y}_t^{*(R,k)}$ the corresponding predicted observation.

By visualization of the predicted movements[†], we can state that CGPDMs, trained with a sufficient amount of data (10, 15 and 20 sequences in this example), are able to capture the cloth dynamics of oscillations along axes Y and Z. Such models obtained satisfying results in a variety of movement ranges. In fact, for smaller movement ranges ($R = 30°$ or $R = 60°$), the reconstructed trajectories of the mesh of points appear very similar to the true ones. On the other hand, for wider ranges ($R = 90°$ or $R = 120°$), discrepancies between true and predicted points begin to be more evident, but the CGPDMs are still able to describe the overall movement of the cloth. In Figure 4, we report, for all the movement ranges, the mean distances between true and predicted mesh points obtained in the test sets by the different CGPDM setups. Results

[†] Video of the experiments available at https://youtu.be/vUO_3nYgMeg.

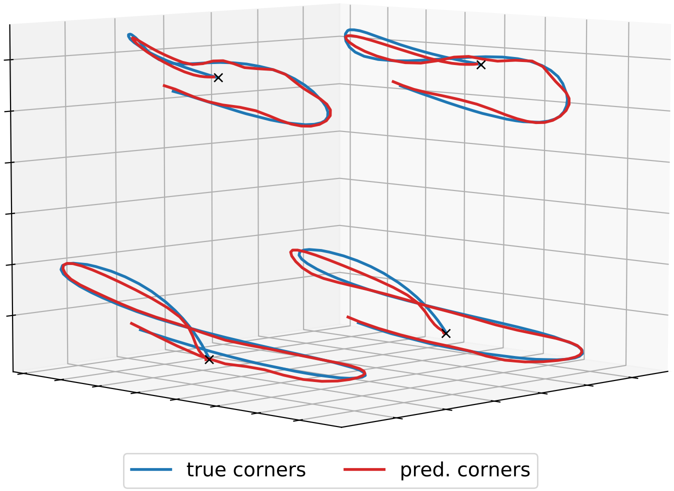

Fig. 5: *True and predicted corners for a real cloth movement.*

are expressed in terms of mean and 95% confidence intervals obtained by averaging over the different training sets adopted (as indicated in Section V-B, we repeated all the experiments 10 times, using a $\mathcal{D}_R^{train}$ of random composition every turn).

As it was natural to expect, CGPDMs trained with only 5 sequences show higher errors than the models trained employing more data. But the resulting errors do not always diminish with the increase in the amount of training trajectories. In fact, in all the considered movement ranges, accuracy does not considerably change passing from 15 training sequences to 20. Moreover, the proposed *highly-parameterized* CGPDM structure seems able to improve accuracy and consistency of the results in the majority of cases. This effect is clearer in a low-data regime, with models trained on 5 or 10 sequences.

VI. Experiment with Real Cloth

Finally, we tested CGPDM on data collected in a real cloth manipulation experiment. For this purpose, we used a Barrett WAM Arm, whose end-effector consists of a coat rack that can firmly grip a piece of cloth from its corners. The overall setup is depicted in Fig. 6. The task considered is totally analogous to the one simulated in Sec. V, despite having here only a single robot involved. We controlled robot's end-effector in position, recording the resulting movement of the cloth through a motion capture system. We combined object detection, image and point cloud processing for segmenting cloth-like objects. The implementation[‡] is based on [30], [31] and [32].

A. Data Collection

As before, we captured the cloth as a 8×8 mesh of points, whose coordinates constitute the observation vector $\boldsymbol{y}_t \in \mathbb{R}^D$ with $D = 192$. While the mesh size can be changed, its sensitivity analysis is left out of the scope of this paper. As control actions, we considered the difference between consecutive position commands (with fixed orientation); hence $\boldsymbol{u}_t \in \mathbb{R}^E$ with $E = 3$. Robot was controlled at 100 Hz, with commands chosen following (12). In this case, f_Y and f_Z have been uniformly sampled in [0.2, 0.5] (Hz) and A was reduced to 0.004 meters, due to the higher control frequency. On the other hand, the motion capture system could work only at lower rates, with no guarantees on the sampling interval. Thus, it was necessary to post-process the data to make them ready for modeling. Firstly, motion capture data were smoothed by a moving average filter. Then we interpolated the positions of both the end-effector and the cloth mesh, to obtain two synchronized sequences of observations and control actions, sampled at 20 Hz. We collected 20 different 3 second long trajectories for each of the two ranges, $R = 30°$ and $R = 60°$ (40 sequences in total).

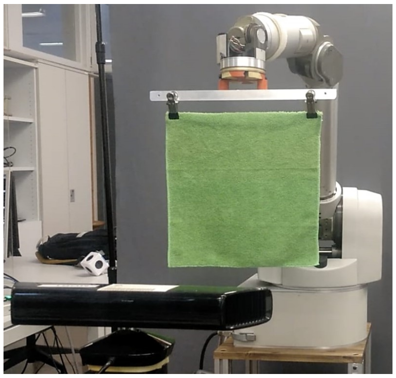

Fig. 6: *Real experimental setup.*

B. Model training & Results

We trained two CGPDMs model, one for $R = 30°$ and the other for $R = 60°$. We adopted the *highly-parameterized* structure, as it performed better in the simulated experiment. For each R, the CGPDM was trained using ten trajectories, while the others were set aside for testing the trained model. As before, we set the dimension of the latent space to $d = 3$.

We tested the two trained CGPDM following the same procedure that was adopted in Sec. V-C. The two models were used to predict the cloth movements obtained in response to the control actions of each test trajectory[†]. The average distance between the real and the predicted mesh points in range $R = 30°$, was 0.012 ± 0.005, while, in range $R = 60°$, it was 0.015 ± 0.006. These results are slightly inferior to those obtained in the simulated experiment (Fig. 4). Nevertheless, CGPDMs seem able to cope with the high noise that afflicts the real experimental setup and still capture the dynamics of the cloth. In Fig. 5, we provide a visual representation of the cloth movements, by representing the true and predicted trajectories of the four corners, for one of the considered test cases.

VII. Conclusions

We presented CGPDM, a modeling framework for high-dimensional dynamics governed by control actions. Essentially, this model projects observations into a latent space of low dimension, where dynamical relations are easier to infer. CGPDMs were applied to a robotic cloth manipulation task, where the observations are the coordinates of the cloth mesh. We tested CGPDMs on both simulated and real experiments. The first was used to compare two possible parameterizations of the model, while in the second we challenged CGPDM with noisy real data. In the future, CGPDM formulation could be extended through the introduction of back constraints [33] to preserve local distances and obtain an explicit formulation of the mapping from the observation to latent space. Moreover, the models can be developed further in order to include contacts.

[‡] Code: https://github.com/MiguelARD/cloth_point_cloud_segmentation

References

[1] C. Bersch, B. Pitzer, and S. Kammel, "Bimanual robotic cloth manipulation for laundry folding," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 1413–1419.

[2] S. Miller, J. van den Berg, M. Fritz, T. Darrell, K. Goldberg, and P. Abbeel, "A geometric approach to robotic laundry folding," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 249–267, 2012.

[3] K. Lakshmanan, A. Sachdev, Z. Xie, D. Berenson, K. Goldberg, and P. Abbeel, "A constraint-aware motion planning algorithm for robotic folding of clothes," in *Experimental Robotics*. Springer, 2013, pp. 547–562.

[4] J. Sanchez, J. A. Corrales Ramon, B. C. BOUZGARROU, and Y. Mezouar, "Robotic manipulation and sensing of deformable objects in domestic and industrial applications: A survey," *The International Journal of Robotics Research*, vol. 37, pp. 688 – 716, 06 2018.

[5] I. Garcia-Camacho, M. Lippi, M. C. Welle, H. Yin, R. Antonova, A. Varava, J. Borras, C. Torras, A. Marino, G. Alenyà, and D. Kragic, "Benchmarking bimanual cloth manipulation," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1111–1118, 2020.

[6] J. Borràs, G. Alenyà, and C. Torras, "A grasping-centered analysis for cloth manipulation," *IEEE Transactions on Robotics*, vol. PP, pp. 1–13, 05 2020.

[7] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer, "Elastically deformable models," *SIGGRAPH Comput. Graph*, vol. 21, no. 4, pp. 205–214, 1987.

[8] D. Baraff and A. Witkin, "Large steps in cloth simulation," *Association for Computing Machinery*, pp. 43–54, 1998.

[9] A. Nealen, M. M¨uller, R. Keiser, E. Boxerman, and M. Carlson, "Physically based deformable models in computer graphics," *Computer Graphics Forum*, vol. 25, pp. 809–836, 2006.

[10] D. Baraff and A. Witkin, "Dexterous manipulation of cloth," *Computer Graphics Forum*, vol. 35, no. 2, pp. 523–532, 2016.

[11] A. Colomé and C. Torras, "Dimensionality reduction for dynamic movement primitives and application to bimanual manipulation of clothes," *IEEE Transactions on Robotics*, vol. 34, no. 3, pp. 602–615, 2018.

[12] N. Lawrence and A. Hyvärinen, "Probabilistic non-linear principal component analysis with gaussian process latent variable models." *Journal of machine learning research*, vol. 6, no. 11, 2005.

[13] N. Koganti, T. Shibata, T. Tamei, and K. Ikeda, "Data-efficient learning of robotic clothing assistance using bayesian gaussian process latent variable model," *Advanced Robotics*, vol. 33, pp. 1–15, 04 2019.

[14] J. A. Delgado-Guerrero, A. Colomé, and C. Torras, "Contextual policy search for micro-data robot motion learning through covariate gaussian process latent variable models," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020, pp. 5511–5517.

[15] N. Koganti, T. Tamei, K. Ikeda, and T. Shibata, "Bayesian nonparametric learning of cloth models for real-time state estimation," *IEEE Transactions on Robotics*, vol. 33, no. 4, pp. 916–931, 2017.

[16] J. M. Wang, A. Hertzmann, and D. J. Fleet, "Gaussian process dynamical models," *Advances in neural information processing systems*, vol. 18, pp. 1441–1448, 2005.

[17] J. M. Wang, D. J. Fleet, and A. Hertzmann, "Gaussian process dynamical models for human motion," *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 2, pp. 283–298, 2007.

[18] R. Urtasun, D. J. Fleet, and P. Fua, "3d people tracking with gaussian process dynamical models," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 1. IEEE, 2006, pp. 238–245.

[19] Z. Zhu, X. You, S. Yu, J. Zou, and H. Zhao, "Dynamic texture modeling and synthesis using multi-kernel gaussian process dynamic model," *Signal Processing*, vol. 124, pp. 63–71, 2016.

[20] N. Koganti, J. G. Ngeo, T. Tomoya, K. Ikeda, and T. Shibata, "Cloth dynamics modeling in latent spaces and its application to robotic clothing assistance," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 3464–3469.

[21] R. M. Neal, *Bayesian learning for neural networks*. Springer Science & Business Media, 2012, vol. 118.

[22] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer School on Machine Learning*. Springer, 2003, pp. 63–71.

[23] J. Kocijan, A. Girard, B. Banko, and R. Murray-Smith, "Dynamic systems identification with gaussian processes," *Mathematical and Computer Modelling of Dynamical Systems*, vol. 11, no. 4, pp. 411–424, 2005. [Online]. Available: https://doi.org/10.1080/13873950500068567

[24] P. Li and S. Chen, "A review on gaussian process latent variable models," in *CAAI Transactions on Intelligence Technology,*, vol. 1, 2016, pp. 366–376.

[25] M. Álvarez, L. Rosasco, and N. Lawrence, *Kernels for Vector-Valued Functions: A Review*, 01 2012.

[26] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

[27] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on scientific computing*, vol. 16, no. 5, pp. 1190–1208, 1995.

[28] F. Coltraro, J. Amorós, M. Alberich-Carramiñana, and C. Torras, "An inextensible model for robotic simulations of textiles," 2021.

[29] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, 2019, pp. 8024–8035.

[30] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," 2020.

[31] C. Rother, V. Kolmogorov, and A. Blake, ""grabcut": Interactive foreground extraction using iterated graph cuts," *ACM Trans. Graph.*, vol. 23, no. 3, p. 309–314, Aug. 2004. [Online]. Available: https://doi.org/10.1145/1015706.1015720

[32] Q. Zhan, Y. Liang, and Y. Xiao, "Color-based segmentation of point clouds," *ISPRS Laser Scanning Workshop*, vol. 38, 07 2009.

[33] N. D. Lawrence and J. Quinonero-Candela, "Local distance preservation in the gp-lvm through back constraints," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 513–520.