Mixtures of Controlled Gaussian Processes for Dynamical Modeling of Deformable Objects

Ce Xu CE.XU@ESTUDIANTAT.UPC.EDU
Institut de Robòtica i Informàtica Industrial, CSIC-UPC

Adrià Colomé ACOLOME@IRI.UPC.EDU
Institut de Robòtica i Informàtica Industrial, CSIC-UPC

Luis Sentis LSENTIS@AUSTIN.UTEXAS.EDU
University of Texas at Austin

Carme Torras TORRAS@IRI.UPC.EDU
Institut de Robòtica i Informàtica Industrial, CSIC-UPC

Editors: R. Firoozi, N. Mehr, E. Yel, R. Antonova, J. Bohg, M. Schwager, M. Kochenderfer

Abstract

Control and manipulation of objects is a highly relevant topic in Robotics research. Although significant advances have been made over the manipulation of rigid bodies, the manipulation of non-rigid objects is still challenging and an open problem. Due to the uncertainty of the outcome when applying physical actions to non-rigid objects, using prior knowledge on objects' dynamics can greatly improve the control performance. However, fitting such models is a challenging task for materials such as clothing, where the state is represented by points in a mesh, resulting in very large dimensionality that makes models difficult to learn, process and predict based on measured data. In this paper, we expand previous work on Controlled Gaussian Process Dynamical Models (CGPDM), a method that uses a non-linear projection of the state space onto a much smaller dimensional latent space, and learns the object dynamics in the latent space. We take advantage of the variability in training data by employing Mixture of Experts (MoE), and we devise theory and experimental validations that demonstrate significant improvements in training and prediction times, plus robustness and error stability when predicting deformable objects exposed to disparate movement ranges.

Keywords: Gaussian Processes, Learning from Demonstration, Dimensionality Reduction, Mixture of Gaussian Experts, Data-driven Dynamic Model Learning

1. Introduction

Manipulation of deformable objects such as cloth is a major challenge in the field of robotics (Sanchez et al. (2018); Garcia-Camacho et al. (2020)). Some notable approaches (Bersch et al. (2011); Miller et al. (2012); Sun et al. (2013)) are based on planning a sequence of predefined actions, executing them and using vision after each action to assess the result and plan the next step. However, for getting closer to human performance in cloth manipulation, the decision of which action to take needs to be performed at the control level in order to speed up the manipulation process. In controlling cloth garments, open-loop policies Colomé and Torras (2018) have achieved a certain degree of success, but the uncertainty coming from the cloth's dynamics makes model-free approaches too vulnerable to any slight deviation from the expected initial condition, such as wind, grasping point, initial wrinkling of cloth, etc. Therefore, prior knowledge of cloth's dynamics can be of great help when implementing a controller. Cloth dynamics has been largely studied in literature (Terzopoulos et al. (1987); Baraff and Witkin (1998); Nealen et al. (2006)), but in controlling a cloth garment with a model-based method, the large dimensionality of a mesh results in very large computational costs for fitting the model to reality. Also, it makes it intractable to build control policies in such high-dimensional spaces. Baraff and Witkin (2016) proposed a Model Predictive Control (MPC) method for cloth manipulation, but the complexity of the cloth model resulted in hours of computations for motions of less than a minute.

The computational costs of the dynamics of cloth, as well as other systems, require newer solutions in order to have a tractable model that is not too complex computationally, and can be fit with real data. Colomé and Torras (2018) used linear dimensionality reduction on the action space of a robot manipulator in order to reduce the action space's dimension, and recently, Amadio et al. (2021) extended a previous work on Gaussian Process Dynamical Models (GPDM, Wang et al. (2005)), which relied on a non-linear dimensionality reduction method to project the system's state to a much smaller dimensional space, and learn the system dynamics there. GPDM used Gaussian Process Latent Variable Models (GPLVM) Lawrence (2005) as a dimensionality reduction method. However, the work by Wang et al. (2005) did not consider control inputs, which were added later in Amadio et al. (2021). The results obtained by Amadio et al. (2021) were satisfactory in learning the dynamics – including control actions – of a cloth garment by using data from certain motions and generalizing in a certain neighbourhood of the data, but showed vulnerability when action's variability was too large, or if different actions were being considered in the data. Therefore, MoE approaches allow better computational complexity scaling due to the splitting of the data and more flexibility to learn wider ranges of movements applied to deformable objects.

In this paper, we expand the theory of CGPDM Amadio et al. (2021) by devising theoretical underpinnings to combine MoE formalism with CGPDM to take advantage of the variability in training data, and significantly improve training and prediction times, plus robustness and error stability when predicting deformable objects exposed to disparate movement ranges. This advancement in prediction and error stability is geared toward the real time manipulation of deformable objects where better computational complexity than the state-of-the-art is increasingly demanded.

2. Preliminaries

2.1. From Gaussian Processes to Control Gaussian Process Dynamical Models

Gaussian processes (GPs) Rasmussen and Williams (2006) are a powerful tool in machine learning allowing designers to make predictions based on data by incorporating prior knowledge. Statistically, a GP is a stochastic process (a collection of indexed random variables), such that every finite collection of those random variables has a multivariate normal distribution. With Gaussian processes, we can estimate the output of a non-linear function if we consider it to be a random variable, where the correlation between different points will be given by a kernel relating their inputs, i.e.:

$$f(X) \sim \mathcal{N}\left(\mu(X), k(X, X')\right),$$

where $X \in \mathcal{R}^{N\times d}$ is the matrix of input variables, N is the number of points, d is the dimension of the input space, $\mu(X)$ is the expected mean values for each output point which can be computed from a prior knowledge given system inputs, and $k(X, X')$ is the kernel matrix that relates the statistics of the input variables. Assuming that we know the function values at various points, $f = f(X)$, we can estimate the function at new points $f^{new} = f(X^{new})$ using a normal distribution where the covariance and the mean are given by the prior knowledge of the mean and the conditioning property:

$$f^{new}|f \sim \mathcal{N}(\mu_{f^{new}} + \Sigma_{X^{new},X}\Sigma_{X,X}^{-1}(f - \mu_f), \Sigma_{X^{new},X^{new}} - \Sigma_{X^{new},X}\Sigma_{X,X}^{-1}\Sigma_{X,X^{new}}), \quad (1)$$

where the prior mean is usually assumed to be the mean of the known data, and the values Σ correspond to covariance matrices.

Employing GPs to estimate the dynamics of deformable objects such as clothing is unfeasible due to the high dimensionality of the input space. The input space corresponds to the states of the deformable object, normally a vector containing the positions of the points defining the deformable mesh which in turn yield high computational costs when computing the correlation matrix. This limitation points us to introduce a dimensionality reduction step to facilitate the learning of the object dynamics. To do so, in Wang et al. (2008) the authors presented a process called Gaussian Process Dynamical Models (GPDM) which extended previous work discussed in Lawrence (2005) addressing the temporal series dependencies. Further on, GPDM was further extended into Control Gaussian Process Dynamical Models (CGPDM) Amadio et al. (2021) by incorporating the control signal of the deformable system as an input variable. The models above involve two steps: a latent mapping that projects high dimensional observations to a low-dimensional latent space, Eq. (2), and a discrete-time Markovian dynamic process that captures the evolution of the time series inside the reduced latent space Eq. (3)

$$y_t = g(x_t; \boldsymbol{\theta}) + n_{y,t} \quad (2)$$

$$x_{t+1} = f(x_t, u_t; \boldsymbol{\phi}) + n_{x,t} \quad (3)$$

Where y_t stands for the high dimensional observation, x_t is its dimensionally reduced representation and u_t is the input control that we are exerting on the deformable object at time t. The g and f labels correspond to non-linear functions that represent the dynamic mapping that we are trying to learn, with $\boldsymbol{\theta}, \boldsymbol{\phi}$ being their hyper-parameters, and $n_{y,t}$, $n_{x,t}$ being zero-mean isotropic white Gaussian noise processes.

2.2. Infinite Mixture of Gaussian Experts

In traditional MoE methods, the parametric model is built without accounting for model data correlations, but in the case of Gaussian processes, correlations are directly embedded into the model. This property was used in Rasmussen and Ghahramani (2001) where they considered the likelihood of the output given the input as the marginalization of all the possible configurations of experts, i.e.:

$$P(O|I, \lambda, \gamma) = \sum_{\mathbf{z}} P(O|I, \mathbf{z}, \lambda)P(\mathbf{z}|I, \gamma) = \sum_{\mathbf{z}} \left[\prod_{j \in E} P(O_{\{i|z_i=j\},:}|I_{\{i|z_i=j\},:}, \lambda_j) \right] P(\mathbf{z}|I, \gamma), \quad (4)$$

where I and O define the input and output variables of the mixture, z defines an indicator variable that assigns each pair of input-output data to one specific expert. Notice that we use the subscript ":" as a symbol indicating all the values in the dimension corresponding to its position – this is similar to Matlab's use of the ":". Thus, $z_i = j$ means that the i-th pair of data is assigned to the j-th expert in the set of experts E. In addition, λ_j are the parameters of the j-th expert. This approach suffers from the curse of dimensionality due to the combinatorics of input-output pairs, so the previous authors proposed an algorithm to estimate the likelihood using a Markov chain, where they employ Gibbs sampling Maklin (2020) of the indicators z given previous expert assignments:

$$P(z_i = j|\mathbf{z}_{(..,\hat{i},..)}, I, O, \lambda, \gamma) \propto P(O|I, z_i = j, \mathbf{z}_{(..,\hat{i},..)}, \lambda)P(z_i = j|\mathbf{z}_{(..,\hat{i},..)}, I, \gamma). \tag{5}$$

where $\hat{i}$ refers to all elements except the i-th element. Because the first term of the right hand side in the above equation can be computed via multiplication of the experts' likelihood as shown in Eq. (4) – assuming independency–, we only need to find the second term in the right hand side. It can be shown Rasmussen (2000), that generalizing a symmetrical Dirichlet process with an α hyper-parameter, the conditional probability of a single indicator when the number of experts tends to infinity is equal to:

$$\begin{aligned} P(z_i = j|\mathbf{z}_{(..,\hat{i},..)}, \alpha) &= \frac{\hat{n}_{(..,\hat{i},..),j}}{n-1+\alpha}, \forall \text{ experts with } \hat{n}_{(..,\hat{i},..),j} > 0 \\ P(z_i \neq \mathbf{z}_{i'} \forall i' \neq i|\mathbf{z}_{(..,\hat{i},..)}, \alpha) &= \frac{\alpha}{n-1+\alpha}, \forall \text{ other (non assigned) experts,} \end{aligned} \tag{6}$$

where $\hat{n}_{(..,\hat{i},..),j}$ is called the occupancy number and represents the number of data points associated with a particular expert j excluding the i-th observation, and n is the total number of data points. We have now determined the probability of assigning a data point to each of the existing experts (which in turn is proportional to the occupation number), and it is now necessary to calculate the probability to create new experts. Therefore, we introduce the dependency of this distribution on the input by adapting the occupancy number employing a local estimator, like a kernel classifier:

$$\hat{n}_{(..,\hat{i},..),j} = (n-1)\frac{\sum_{i' \neq i} k_\gamma(I_{i,:}, I_{i',:})\delta(z_{i'}, j)}{\sum_{i' \neq i} k_\gamma(I_{i,:}, I_{i',:})}, \tag{7}$$

where $\delta(.,.)$ is the Kronecker's Delta function to sum up only the points corresponding to the j-th expert. The kernel function k_γ will be parametrized by γ and will be defined later. Once we have sampled the indicators using Gibbs sampling, we can apply maximization over the likelihood using a fixed experts assignment. Iterating over this process, we approximate the posterior likelihood of Eq. (4) without the need to directly compute the summations.

3. Learning Mixture of CGPDMs

3.1. Combining Mixture of Experts and CGPDMs

Like in GPDMs, we will maximize the likelihood of the latent variables and the parameters of the Gaussian model. Y and U are matrices that contain the observations of the mesh and the control inputs in their rows, X represents the latent variables that characterize, in their lower dimensional space, the Y observations row-wise, and α, β stand for all the parameters used in a *Mapping Model* and a *Latent Model*, respectively, that will be described further below. As the optimization does not depend on the variables Y, U, we can develop the following optimization problem:

$$\operatorname*{argmax}_{X,\alpha,\beta} P(X, \alpha, \beta|Y, U) = \operatorname*{argmax}_{X,\alpha,\beta} P(Y, X, \alpha, \beta, U) = \operatorname*{argmax}_{X,\alpha,\beta} P(Y|X, \beta)P(X|\alpha, U), \tag{8}$$

where $P(Y|X,\beta)$ and $P(X|\alpha,U)$ can be considered as two separated mixtures of GP models. For easier identification, we will name the first of these terms as the **Mapping Model** – because it is related to the ability of the model to infer the observations from the latent points – and the second term as the **Latent Dynamics Model** – because it is related to the ability of the model to predict the next point in the latent space given the previous positions and the current control. We also define D as the total number of dimensions of the observable space and d the number of dimensions of the latent space. The Mapping Model likelihood can be expressed by marginalizing all the possible configurations of the experts ($\mathbf{z}^{map}$):

$$P(Y|X,\beta) = \sum_{\mathbf{z}^{map}} \left[\prod_{j\in E^{map}} P(Y_*|X_*,\theta_j^{map}) \right] P(\mathbf{z}^{map}|X,\phi^{map}), \tag{9}$$

where we define the subscript $* := \{i|z_i^{map} = j\},:$, $\mathbf{z}^{map}$ is the expert identifier used for assigning experts to Mapping Model. In addition we split β into two additional parameters dubbed, θ_j^{map} and ϕ^{map}, which correspond to specific parameters for the j-th expert, and to gating parameters, respectively.

For the Latent Dynamics Model, we can proceed similarly, but with some new subtleties. The likelihood of the latent variables given the control input and the model parameters should be written in terms of the input and output variables of the dynamical function that we want to estimate. To estimate the dynamics of clothing, we use as inputs to the describing function of the model the last two latent states in time and the control input. The output of the function is the difference between the previous and the current latent states. Using the previously described mixture of Gaussian experts we define the following model:

$$P(X|\alpha,U) = P(\Delta|\hat{X},\alpha) = \sum_{\mathbf{z}^{lat}} \left[\prod_{j\in E} P(\Delta_\star|\hat{X}_\star,\theta_j^{lat}) \right] P(\mathbf{z}^{lat}|\hat{X},\phi^{lat}), \tag{10}$$

where we define the subscript $\star := \{i|z_i^{lat} = j\},:$, Δ is the matrix of latent state increments $(X_{2:n,:} - X_{1:n-1,:})$, and $\hat{X}$ represents various latent variables and input states concatenated horizontal-wise as follows: $(X_{0:n-2,:}, X_{1:n-1,:}, U_{1:n-1,:})$. The equality $P(X|\alpha,U) = P(\Delta|\hat{X},\alpha)$ can be demonstrated thanks to the properties of the conditioned probability. We have, once more, split the model parameters α into experts parameters θ_j^{lat} and gating parameters ϕ^{lat}. Finally, we reformulate Eq. (8) as follows:

$$\begin{aligned} &\operatorname*{argmax}_{X,\alpha,\beta} P(Y|X,\beta)P(X|\alpha,U) = \\ &\operatorname*{argmax}_{X,\theta_j^{map},\phi^{map},\theta_j^{lat},\phi^{lat}} \left(\sum_{\mathbf{z}^{map}} \left[\prod_{j\in E} P(Y_*|X_*,\theta_j^{map}) \right] P(\mathbf{z}^{map}|X,\phi^{map}) \right) \\ &\qquad \cdot \left(\sum_{\mathbf{z}^{lat}} \left[\prod_{j\in E} P(\Delta_\star|\hat{X}_\star,\theta_j^{lat}) \right] P(\mathbf{z}^{lat}|\hat{X},\phi^{lat}) \right), \end{aligned} \tag{11}$$

which allows us to compute an estimate of the total likelihood of the latent model and maximize it without computing expensive summations of likelihoods for all possible configurations.

Note that in case that we provide more than one trajectory as a training reference, we only need to join all the observable Y matrices and their corresponding latent states X. As for the Latent Dynamics Model we need to join the increment matrices for all the trajectories, i.e. $\Delta = (X^1_{2:n,:} - X^1_{1:n-1,:}; X^2_{2:n,:} - X^2_{1:n-1,:}; \ldots)$ for the output, and $\hat{X} = (X^1_{0:n-2,:}, X^1_{1:n-1,:}, U^1_{1:n-1,:}; X^2_{0:n-2,:}, X^2_{1:n-1,:}, U^2_{1:n-1,:}; \ldots)$ for the input, where the superscripts indicate the trajectory index.

3.2. Design Choices

To characterize the Gaussian experts used in the Mapping Model, we consider that each of the D dimensions of the output space is independent of each other and has a base kernel $k_y(.,.)$ scaled with a factor $(\omega^{map})^{-2}$. The kernel chosen is a Radial Basis Function (RBF) with a Gaussian noise:

$$k_y(X_{i,:}, X_{j,:}) = \exp\left(-\|X_{i,:} - X_{j,:}\|_{\Lambda^{map}}\right) + (\sigma^{map})^2\, \delta\left(X_{i,:}, X_{j,:}\right), \tag{12}$$

where $\|X_{i,:} - X_{j,:}\|_{\Lambda^{map}} = \left[\sum_{k=1}^{d} (\lambda_k^{map})^2 (X_{i,k} - X_{j,k})^2\right]^{1/2}$, and $\delta(.,.)$ is the Kronecker delta function. The likelihood for each expert is the multiplication of the likelihood of each column of Y_*, yielding the following expression:

$$P(Y_*|X_*, \theta_j^{map}) = \frac{|W_j^{map}|^N\, exp\left[-\frac{1}{2}\text{tr}\left(K_y(X_*)^{-1} Y_*\, \left(W_j^{map}\right)^2\, Y_*^T\right)\right]}{\sqrt{(2\pi)^{(N\cdot D)}\, |K_y(X_*)|^D}}, \tag{13}$$

where $\theta_j^{map} = (\omega_{j,1}^{map}, ..., \omega_{j,D}^{map}, \lambda_{j,1}^{map}, ..., \lambda_{j,d}^{map}, \sigma_j^{map})$, N is the number of data points assigned to the j-th expert, $W_j^{map} := diag(\omega_{j,1}^{map}, ..., \omega_{j,D}^{map})$, and $K_y(X_*)$ is the covariance matrix built with the previously defined kernel function.

Similarly, we assume independence in all the d dimensions of the Latent Dynamics Model and employ the same base kernel function $k_x(.,.)$ across them, scaled with a factor $(\omega^{lat})^{-2}$. As such, we design the following RBF kernel with a quadratic component and a noise term:

$$k_x(\hat{X}_{i,:}, \hat{X}_{j,:}) = exp\left(-\|\hat{X}_{i,:} - \hat{X}_{j,:}\|_{\Lambda^{lat}}\right) + [\hat{X}_{i,:}, 1]\Psi^{lat}[\hat{X}_{j,:}, 1]^T + \left(\sigma^{lat}\right)^2 \delta(\hat{X}_{i,:}, \hat{X}_{j,:}), \tag{14}$$

where $\|\hat{X}_{i,:} - \hat{X}_{j,:}\|_{\Lambda^{lat}} = \left[\sum_{k=1}^{2d+dim(u)} (\lambda_k^{map})^2 (\hat{X}_{i,k} - \hat{X}_{j,k})^2\right]^{1/2}$, $dim(u)$ is the number of horizontal dimensions of the control variables, and $\Psi^{lat} = diag(\psi_1^2, \ldots, \psi_{2d+dim(u)+1}^2)$. The likelihood for each expert is the multiplication of the likelihood of each column of Δ_*, yielding the following expression:

$$P(\Delta_\star|\hat{X}_\star, \theta_j^{lat}) = \frac{|W_j^{lat}|^N exp\left[-\frac{1}{2}\text{tr}\left(K_x(\hat{X}_\star)^{-1}\Delta_\star\left(W_j^{lat}\right)^2 \Delta_\star^T\right)\right]}{\sqrt{(2\pi)^{(N\cdot d)}|K_x(\hat{X}_\star)|^D}}, \tag{15}$$

where $\theta_j^{lat} := (\omega_{j,1}^{lat}, \ldots, \omega_{j,d}^{lat}, \lambda_{j,1}^{lat}, \ldots, \lambda_{j,d}^{lat}, \psi_{j,1}, ..., \psi_{j,2d+dim(u)+1}, \sigma_j^{lat})$, $W_j^{lat} := diag(\omega_{j,1}^{lat}, ..., \omega_{j,d}^{lat})$, and $K_x(\hat{X})$ is the covariance matrix built with the kernel function that we described further above.

The reason to choose these types of likelihoods is due to their good performance previously shown in single expert systems as described in Amadio et al. (2021). For the Mapping and Latent

Dynamics gating models – $P(\mathbf{z}^{map}|X,\phi^{map})$ and $P(\mathbf{z}^{lat}|X,\phi^{lat})$ respetively –, we have chosen a simple exponential distance of the scaled square distance kernel as follows:

$$k_{\phi}^{map}(X_{i,:},X_{j,:}) = exp\left(-\frac{1}{2}\sum_{k=1}^{d}(X_{i,k}-X_{j,k})^2/(\phi_k^{map})^2\right), \tag{16}$$

$$k_{\phi}^{lat}(\hat{X}_{i,:},\hat{X}_{j,:}) = exp\left(-\frac{1}{2}\sum_{k=1}^{dim(u)+2d}(\hat{X}_{i,k}-\hat{X}_{j,k})^2/(\phi_k^{lat})^2\right). \tag{17}$$

To decide the number of latent dimensions that we are going to employ, we have to find a balance between the precision that we want to get in the reconstruction of the observed data, and the computational cost for having a large number of latent dimensions. In our case, we have chosen $d = 5$ as a result of computing the variance ratio of the Principal Component Analysis (PCA) applied to the observation space and selecting the most significant dimensions.

3.3. Training Procedure

Our training procedure consists of several steps where we optimize and sample different parts of the models sequentially, always trying to maximize the final likelihood of the complete model:

Parameter initialization: As mentioned before, we obtain the latent space using PCA. To initialize the configuration of the Mapping Model, we initialize its experts using a k-means clustering algorithm over the observation space. Regarding the Latent Dynamics Model, we assign all points belonging to a given trajectory to the same expert. First, we initialize $\hat{X}$ as defined in Eq.(10) using all the training trajectories combined. We then assign the portions of $\hat{X}$ corresponding to a single trajectory, $\{X^k_{0:n-2,:}, X^k_{1:n-1,:}, U^k_{1:n-1,:}\}$, to a random expert for each of them, where k is the index of the trajectory. We initialize the expert parameters θ_j^{map}, θ_j^{lat} using a UNIFORM$(0.5, 1)$, as such distribution has been empirically shown to yield well-conditioned correlation matrices for optimization. Finally, to initialize the gating parameters we pre-train them as explained further below under the label: **Maximize the log-likelihood of the predictions**.

Expert likelihood optimization: We then perform likelihood optimization of the Mapping and Latent Dynamics models for the experts as follows:

$$\begin{aligned}
&\underset{X,\theta^{map},\theta^{lat}}{\text{argmax}}\ P(Y|X,\theta^{map},\mathbf{z}^{map})P(X|\theta^{lat},U,\mathbf{z}^{lat}) = \\
&\underset{X,\theta^{map},\theta^{lat}}{\text{argmax}} \prod_{j\in E^{map}} P(Y_*|X_*,\theta_j^{map}) \cdot \prod_{j\in E^{lat}} P(\Delta_\star|\hat{X}_\star,\theta_j^{lat}) = \\
&\underset{X,\theta^{map},\theta^{lat}}{\text{argmin}} \sum_{j\in E^{map}} \mathcal{L}_j^{map} + \sum_{j\in E^{lat}} \mathcal{L}_j^{lat}
\end{aligned} \tag{18}$$

where the losses are described using negative log likelihood, i.e. $\mathcal{L}_j^{map} = -\log(P(Y_*|X_*,\theta_j^{map}))$, and $\mathcal{L}_j^{lat} = -\log(P(\Delta_\star|\hat{X}_\star,\theta_j^{lat}))$.

Gibbs sampling: We perform Gibbs sampling over the expert configuration variables using Eq.(5) for both the Mapping and Latent Dynamics Models. For the mapping model we first remove the input/output data points from their expert. We then compute the probabilities for each expert –fixing the maximum number of points per expert to N_{max}– and assign the probability of creating a new expert given by $\frac{\alpha}{n-1+\alpha}$. For the Latent Dynamics model we prioritized assigning the input/output data points pertaining to the same trajectory to the same expert to avoid fragmentation and sampling whole trajectories avoiding the gating term in Eq.(5).

Gating optimization: We employ the error between gating predictions and current expert assignments. The gating predictions are obtained by removing each i-th data point from its expert and computing the gating model prediction:

$$\underset{\phi^{map},\phi lat}{\operatorname{argmax}} \sum_i P(z_i^{map}|X, \mathbf{z}^{map}_{(..,\hat{i},..)}\phi^{map}) + \sum_i P(z_i^{lat}|\hat{X}, \mathbf{z}^{alt}_{(..,\hat{i},..)}\phi^{lat}). \tag{19}$$

We repeat these steps – parameter initialization, expert optimization, Gibbs sampling and gating optimization – until model convergence.

Once we have performed model training, we end up with a particular configuration of the experts, $(\mathbf{z^{map}}, \mathbf{z^{lat}})$. We consider the combination of the Gaussian Models described in Eqs. (13) and (15) as a form of MoE. Instead of mixing the probability distribution of all experts weighted with the gating model, we use the predictions of each expert as separate, uncorrelated random variables with normal distributions. As such, the MoE behaves as the weighted sum of all the random variables. Since all the mapping and latent dynamics model predictions are uncorrelated and their covariance matrices are diagonal, our MoE output prediction becomes:

$$Y_{model} = \sum_{j\in E} P(j)Y_j \sim \sum_{j\in E} P(j)\mathcal{N}(\mu_j, \sigma_j^2) \sim \mathcal{N}\left(\sum_{j\in E} P(j)\mu_j, \sum_{j\in E} P(j)^2\sigma_j^2\right), \tag{20}$$

where $P(j)$ is the probability of assigning the input to the j-th expert and Y_j is the prediction given by the j-th expert.

The above choice for our model provides robustness to predictions far from the known training points and near turning points of the trajectories, where big changes of position and direction in the latent space happen. To map new observations into latent variables we average the latent representation of the nearest training points to the new observations [1].

4. Experiments

For evaluation we employ a physics simulation of inextensible textiles Coltraro et al. (2022) for which the authors gave us access to the dataset described in Amadio et al. (2021). The dataset contains data from swinging a square cloth garment grabbed and manipulated from the two upper corners. The simulation is performed by applying forces to this two points in a pattern that resembles a butterfly shape, performing a combination of diagonal and vertical movements, with diagonal movements at 30, 60, 90 and 120 degrees of inclination of the applied forces with respect to the horizontal plane. The greater the degree of inclination of the forces, the greater the variability of the dataset, and therefore the greater need for more training for the models. We followed the procedure in Sec.3.3 and solved Eq. (18) using *ADAM* optimizer Kingma and Ba (2017).

1. A further and more detailed development can be found in Xu Zheng (2021) in section 3.2.

To evaluate our model performance, we measure the accuracy of the observable space prediction with the relative error along one complete trajectory (or some subsequence of the trajectory):

$$\epsilon^k = 100 \cdot \sum_{t=1}^{N} \frac{\|y_t^k - y_t^{*k}\|}{\|y_t^k\|}, \tag{21}$$

where the k superscript indicates the k-th test trajectory, y_t represents the observation at time t and y_t^* is our model prediction. This prediction is obtained by mapping from the latent trajectory x^* using our mapping model, and the latent trajectory is obtained by compounding the predictions of the latent dynamical model.

To avoid the bias of the model to just a few experts and improve the computation time, we set a maximum size of each expert to N_{max}. As the training time and the evaluation time for every single Gaussian expert grows cubically with the number of points assigned, our Mixture of CGPDM (MoCGPDM) will have better computational complexity than CGPDM Amadio et al. (2021) as the complexity of one expert only depends on a constant N_{max} cubically and linearly with the number of experts – that contains N_{max} points each at most. Nevertheless, the training time will still be cubical due to the complexity that the Gibbs sampling carries with it, involving the computation of a new likelihood for each one of the possible designated experts for each training points[2].

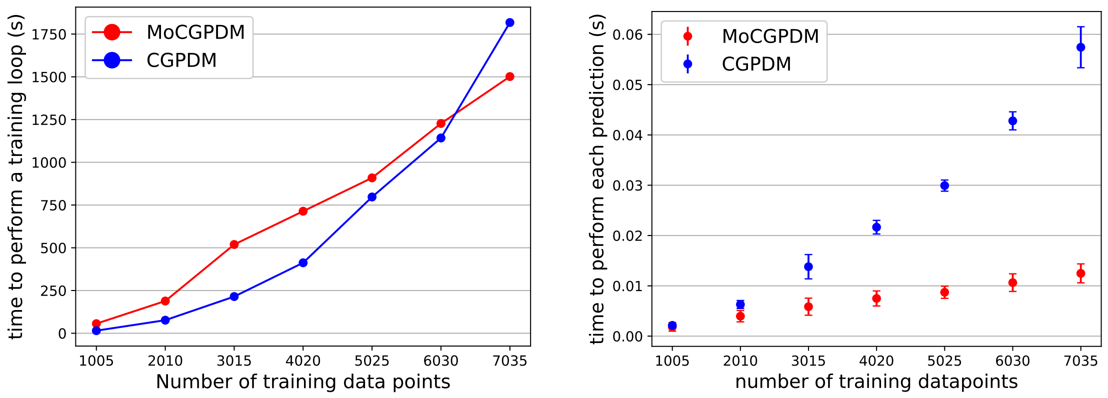

Figure 1: On the left side, we show the time needed to perform one optimization training loop as a function of the number of data points used for training. We compare the time needed to make one training step with CGPDM against the time that it takes with our MoCGPDM method. On the right side, we show the average time needed to perform one prediction step by the MoCGPDM compared to CGPDM as a function of the training dataset size.

As we can see in Fig.1 the training of MoCGPDM grows at a slower rate than CGPDM. In addition, the prediction time for MoCGPDM grows linearly – as an increase of the number of points will only affect to the number of experts involved– compared to cubical growth for CGPDM. Also CGPDM requires more training loops than MoCGPDM, meaning that the needed time to training CGPDM will surpass the total time needed to train MoCGPDM. In Fig.2 we compare the results

2. A more detailed development of the computational complexity can be found in Xu Zheng (2021), Sections 4.1.3 and 4.1.4. In this case, training and executing smaller experts results in a better computational complexity.

of MoCGPDM and CGPDM for sequences of movements exerted at various degrees and repeated multiple times each. We can observe that MoCGPDM significantly reduces the error variance thus showing higher stability. MoCGPDM also outperforms CGPDM in situations with high density of trajectories (e.g. 20 similar trajectories) showing robustness to redundancy compared to CGPDM.

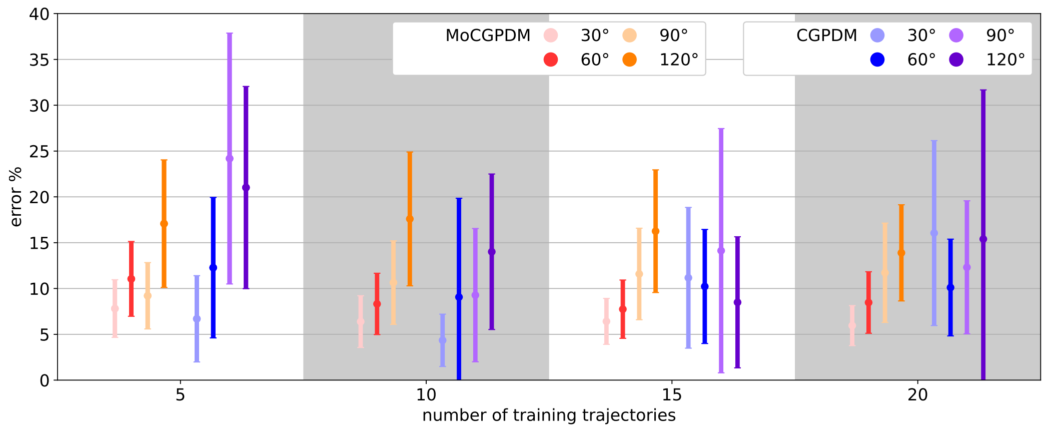

Figure 2: Mean relative errors (with 95% confidence internal) obtained in different trajectory setups with various disparate movement ranges. Warm colors correspond to the performance of our MoCGPDM method while cold colors correspond to CGPDM.

5. Conclusions

We have proposed a data-driven model that improves significantly the prediction time of deformable objects as shown in Fig. 1 while improving robustness and stability as shown in Fig. 2. One interesting remark is that it relies heavily in the choosing of hyper-parameters. The maximum size of the experts (in terms of number of data points) will affect the sensibility of the model to sparse or dense data points, the greater the sparsity, the more data points we will need to perform better using each of the experts.

One of the parameters that we have found meaningful to describe the performance of the trained model is the smoothness of the optimized latent trajectories. Because they are initialized with PCA, they tend to yield smooth trajectories at the beginning. If we later observe saw tooth trajectories they can be easily inspected visually to change parameters or stop optimizing thus avoiding overfitting.

We have employed the proposed model to learn the dynamics of cloth garments allowing a very good reduction of dimensions of the used dataset, from 192 to 5. The high non-linearities of the movement of deformable objects makes the use of Gaussian processes a good choice to perform predictions. In the future we would like to use this predictions for online control such as employing model predictive control for robustness during manipulation of clothes and other deformable objects.

The code developed for the MoCGPDM is publicly available in a Github repository along with the employed dataset [3].

3. `https://github.com/cexuzheng/MoCGPDM`.

Acknowledgments

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).

We would like to thank the members of the HCRL Lab and the Department of Aerospace Engineering and Engineering Mechanics at The University of Texas at Austin for their feedback during the development of this work.

References

Fabio Amadio, Juan Antonio Delgado-Guerrero, Adrià Colomé, and Carme Torras. Controlled gaussian process dynamical models with application to robotic cloth manipulation. In *arXiv:2103.06615*. SIAM, 2021.

David Baraff and Andrew Witkin. Large steps in cloth simulation. In *Association for Computing Machinery*, pages 43–54, 1998.

David Baraff and Andrew Witkin. Dexterous manipulation of cloth. In *Computer Graphics Forum*, volume 35, pages 523–532, 2016.

Christian Bersch, Benjamin Pitzer, and Sören Kammel. Bimanual robotic cloth manipulation for laundry folding. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1413–1419, 2011.

Adrià Colomé and Carme Torras. Dimensionality reduction for dynamic movement primitives and application to bimanual manipulation of clothes. In *IEEE Transactions on Robotics*, volume 34, pages 602–615, 2018.

Franco Coltraro, Jaume Amorós, Maria Alberich-Carramiñana, and Carme Torras. An inextensible model for the robotic manipulation of textiles. In *Applied Mathematical Modelling*, volume 101, pages 832–858, 2022. doi: https://doi.org/10.1016/j.apm.2021.09.013.

Irene Garcia-Camacho, Martina Lippi, Michael C. Welle, Hang Yin, Rika Antonova, Anastasiia Varava, Julia Borras, Carme Torras, Alessandro Marino, Guillem Alenyà, and Danica Kragic. Benchmarking bimanual cloth manipulation. In *IEEE Robotics and Automation Letters*, volume 5, pages 1111–1118, 2020.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

Neil Lawrence. Probabilistic non-linear principal component analysis with gaussian process latent variable models. In *Journal of Machine Learning Research*, volume 6, pages 1783–1816, 2005.

Cory Maklin. Gibbs sampling, 2020.

Stephen Miller, Jur van den Berg, Mario Fritz, Trevor Darrell, Ken Goldberg, and Pieter Abbeel. A geometric approach to robotic laundry folding. In *The International Journal of Robotics Research*, volume 31, pages 249–267, 2012.

Andrew Nealen, Matthias Müller, Richard Keiser, Eddy Boxerman, and Mark Carlson. Physically based deformable models in computer graphics. In *Computer Graphics Forum*, volume 25, pages 809–836, 2006.

Carl E. Rasmussen. The infinite gaussian mixture model. In *Adv Neural Inf Process Syst*, volume 12, pages 554–560, 2000.

Carl E. Rasmussen and Zoubin Ghahramani. Infinite mixtures of gaussian process experts. In *International Conference on Neural information Processing Systems: Natural and Synthetic*, pages 881–888. MIT Press, 2001.

Carl E. Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, USA, 2006.

Jose Sanchez, Juan Antonio Corrales Ramon, B. Chedli BOUZGARROU, and Youcef Mezouar. Robotic manipulation and sensing of deformable objects in domestic and industrial applications: A survey. In *The International Journal of Robotics Research*, volume 37, pages 688–716, 2018.

Kevin Sun, Gerardo Aragon-Camarasa, Paul Cockshott, Simon Rogers, and Jan Siebert. A heuristic-based approach for flattening wrinkled clothes. In *Towards Autonomous Robotic Systems Conference*, volume 8069, pages 148–160, 2013.

Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically deformable models. In *SIGGRAPH Comput. Graph*, volume 21, pages 205–214, 1987.

Jack M Wang, Aaron Hertzmann, and David J Fleet. Gaussian process dynamical models. In *Advances in neural information processing systems*, volume 18, pages 1441–1448, 2005.

Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Gaussian process dynamical models for human motion. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 30, pages 283–298, 2008.

Ce Xu Zheng. Gaussian process applied to modeling the dynamics of a deformable material. In *Universitat Politècnica de Catalunya*, 2021.