**IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.**

# Task-Parameterized Motion Learning with Time-Sensitive Constraints

Julian Richter[1,2], João Oliveira[3], Christian Scheurer[1], Jochen J. Steil[2] and Niels Dehio[1]

***Abstract*—Teaching motion skills to robots through demonstrations has becomes widely popular. However, precise execution of start-, via-, and end-poses at given times is often not guaranteed, limiting the technology transfer to industrial application. To address this issue, we propose the novel *Constrained Expectation Maximization* (CEM) algorithm, which enforces *time-sensitive constraints* (TSC) when learning Gaussian Mixture Models (GMM). Our approach applies to data on Riemannian manifolds and extends to task-parameterized scenarios. We validate CEM against state-of-the-art methods on handwritten data and real robot applications utilizing the KUKA LBR iiwa. By enforcing constraints within the learning process, CEM achieves improved and more efficient reproduction of the demonstration data.**

## I. Introduction

Robot manufacturers seek to provide easy-to-use interfaces, allowing non-experts to teach robot skills intuitively. Underlying these efforts is research in *Learning from Demonstration* (LfD) [1], also referred to as *Programming by Demonstration* (PbD) [2] or *Imitation Learning* [3]. At its core, it relies on demonstration data, collected via hand-guiding [4], teleoperation [5], handheld input devices [6], or camera-based techniques [7]. The objective is to find suitable model parameters that reproduce the demonstrations accurately. Through the resulting model, the robot is then controlled autonomously. Modern techniques furthermore allow the robot behavior to adapt to variations of the task, often referred to as *Task-Parameterization* [8].

Many applications require precise execution of start-, via- or end-poses. Therefore, a wide range of research on learning state-dependent dynamical systems (DS) $\dot{\mathbf{x}} = f(\mathbf{x})$ focuses on enforcing Lyapunov stability at a desired attractor [9]–[11]. However, learning a DS may suffer from the high-dimensional input dimension and restrictive stability constraints that compromise its performance. For example, a globally stable first-order DS can not generate a trajectory that crosses its own path. On the contrary, time-dependent systems $\mathbf{x} = f(t)$ provide more flexibility in representing arbitrarily shaped trajectories [4], [12], [13].

Manuscript received: February 7, 2025; Revised: June 18, 2025; Accepted: August 30, 2025

This paper was recommended for publication by Editor Aleksandra Faust upon evaluation of the Associate Editor and Reviewers' comments.

This work was partly supported by KUKA Deutschland GmbH, the state of Bavaria through the OPERA project DIK-2107-0004/DIK0374/01 and FCT (2020.07160.BD), through IDMEC, under LAETA, project UIDB/50022/2020.

[1]Technology and Innovation Center (TIC), KUKA Deutschland GmbH, Germany, e-mail: Julian.Richter@kuka.com

[2]Institute of Robotics and Process Control (IRP), Technische Universität Braunschweig, Germany

[3]IDMEC, Instituto Superior Técnico, Portugal

Digital Object Identifier (DOI): see top of this page.




Fig. 1. Experimental setup for a scenario with task variance. Demonstrations are provided through hand-guiding. The variable motion skill includes picking a board out of the non-stationary rack, holding it in front of the camera (left) and inserting it into the shelf in different slots (right).

Probabilistic methods are often used in the context of LfD [1]. However, state-of-the-art Gaussian Mixture Regression (GMR) based on Expectation Maximization (EM) [14] fails when dealing with applications that require precise task execution at specific parts of the process.

Our main contribution is the generic formulation of the *Constrained Expectation Maximization* algorithm for data on Riemannian manifolds that enforces precision where required through prior task knowledge. The approach also extends to the more generic Task-Parameterized Gaussian Mixture Models (TP-GMM) [8]. In this letter, we demonstrate probabilistic learning of precise pick & place motions in end-effector space, guaranteeing the exact reaching of desired positions and orientations. The proposed approach is validated on handwritten data and in real robot applications utilizing the KUKA LBR iiwa, see Fig. 1. Experiments reveal that enforcing the constraints within the learning process results in an improved and more efficient reproduction of the demonstration data compared to related works, which are limited to enforcing these constraints at inference-time. As an additional minor contribution, we extend Task-Parameterized Kernalized Movement Primitives (TP-KMP) [15] for $\mathrm{SE}(3)$ in order to conduct a meaningful baseline comparison.

## II. Related Work

Over the last decades, various techniques have been developed for learning robot motions from demonstrations. In Dynamic Movement Primitive (DMP) [16], a dynamical system is coupled with a learned forcing term that encodes the desired behavior to generate point-to-point motions, guaranteeing global stability at the attractor point [9]. Probabilistic Movement Primitives (ProMP) [17] extend the DMP-concept and additionally capture the variation in the demonstrations.

**IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.**

Alternatively, the encoding of demonstration data using a GMM remains popular in recent research. Utilizing the learned joint probability model, GMR provides a fast and efficient way to compute a multi-variate output distribution for a given input. The time needed for regression is independent on the number demonstrations, and hence, robot commands are generated in an online manner [18]. The GMM representation has been utilized in developing the Stable Estimator of Dynamical Systems (SEDS) [9]. The approach introduces constraints from Lyapunov theory into the learning procedure, thereby enforcing global stability at the desired attractor point. Addressing the SEDS limitations resulting from the quadratic Lyapunov function, the extended $\tau$-SEDS [10] applies a diffeomorphic transformation $\tau$ to the demonstration data. An alternative diffeomorphism achieves better results [19]. In [20] a Linear Parameter Varying Dynamical System (LPV-DS) was introduced that achieved better reproduction quality compared to SEDS.

Despite recent research advancing in learning globally stable dynamical systems (DS) from demonstrations, the range of learnable robot motions remains limited for first-order DS. On the other hand, time-dependent systems encode trajectories with arbitrary complexity. A GMM representation for time-dependent systems was introduced in [4]. The Task-Parameterized Gaussian Mixture Model (TP-GMM) [21] framework is a direct extension of this approach for scenarios with task variance, creating interpolation and extrapolation capabilities by modeling the demonstration data from multiple local frame perspectives that are adjusted to the current environment and fused at run-time [13].

Riemannian manifolds provide a convenient way to extend methods originally developed for Euclidean space [18]. By considering unit quaternions as elements of the $\mathbb{S}^3$ manifold, the original GMM and TP-GMM framework have been extended for orientation learning in [13]. In addition, the TP-GMM approach has been further improved by autonomous tuning of task-parameters and manual adjustment of their contribution during motion generation [22]. Identifying local frames that are irrelevant to the overall task was proposed in [23]. Adding a weighting term during fusion to regard the importance of each perspective differently based on their distance to the input was investigated in [24]. However, this method is limited when start and end-pose with their associated task-parameters are close to each other.

Enforcing desired properties such as specific start, target, and/or via-poses within motion generation utilizing GMR is not straightforward, as observed in [15]. To address this issue, a Bagging-GMM approach was introduced in [25], where multiple base learners are trained on different subsets of the data. Combining their predictions through suitable weights, time-sensitive constraints (TSC) are enforced on the generated motion. Kernelized Movement Primitives (KMP) [15] on the other hand, build upon a reference trajectory obtained from GMR and allows to enforce desired target or via-poses through trajectory modulation [15], i.e., manual modification of the reference trajectory. The approach has been extended for unit quaternions [26], SPD-matrices [27] and Task-Parameterization in Euclidean Space [28]. Alternatively, taking mathematical constraints into account during data-driven optimization facilitates the encoding of prior knowledge into the learning process [29].

In this letter, we extend the EM algorithm to account for constraints by introducing a novel CEM formulation. This formulation enforces TSC during learning of a GMM, unlike previous works which enforce these constraints at inference-time. It applies to data on Riemannian manifolds and task-parameterized scenarios. Note that an adjustment of the E-step was investigated in [30], resulting in more precise control over data assignment to specific Gaussian components, by enforcing prior knowledge on the posterior probability distribution. For learning robot motion from demonstrations, however, we propose a modification of the M-step instead.

## III. Motion Learning on Riemannian Manifolds

Given a set of demonstration data $\boldsymbol{\Upsilon} = \{\boldsymbol{\xi}_n\}_{n=1}^N$ with $N > 1$ samples, we assume each sample $\boldsymbol{\xi}_n \in \mathbb{R}^D$ is composed of time input $t \in \mathbb{R}$ and state output $\mathbf{x} \in \mathbb{R}^{D-1}$. In this section, we reiterate how to express a motion skill as a regression problem based on this data encoded in a GMM.

### A. Gaussian Mixture Model on Riemannian Manifolds

A Gaussian Mixture Model consists of $K$ Gaussian components that approximate the data's joint probability [31]

$$\mathcal{P}(\boldsymbol{\xi}) = \mathcal{P}(t, \mathbf{x}) = \textstyle\sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{\xi} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\,, \tag{1}$$

where each $k$-th Gaussian is described by a mean vector $\boldsymbol{\mu}_k$, an SPD covariance matrix $\boldsymbol{\Sigma}_k$, and is weighted by a prior probability $\pi_k \in \mathbb{R}$. Furthermore, all prior probabilities must sum up to one, i.e., $\sum_{k=1}^{K} \pi_k = 1$.

End-effector orientations are often described using unit quaternions, which represent elements on $\mathbb{S}^3$, a 3-dimensional Riemannian manifold embedded in 4-dimensional space $\mathbb{R}^4$ [13], [32]. Therefore, GMR/GMR, initially defined for vectors in Euclidean space, has been extended by considering Gaussians with mean vectors $\boldsymbol{\mu}_k \in \mathcal{M}$ on the manifold and covariance matrices $\boldsymbol{\Sigma} \in (\mathcal{T}_{\boldsymbol{\mu}_k} \times \mathcal{T}_{\boldsymbol{\mu}_k})$ defined in the tangent space associated with their mean vectors. The latter describes the dispersion of the original data projected onto the tangent space [33]. The $k$-th Gaussian probability density function (PDF) writes

$$\mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{\exp\left(-\frac{1}{2}\mathbf{Log}_{\boldsymbol{\mu}_k}(\mathbf{x})^T \boldsymbol{\Sigma}_k^{-1} \mathbf{Log}_{\boldsymbol{\mu}_k}(\mathbf{x})\right)}{\sqrt{(2\pi)^D \det(\boldsymbol{\Sigma}_k)}}\,, \tag{2}$$

where $\mathbf{Log}_{\boldsymbol{\mu}}(\mathbf{x}) : \mathcal{M} \to \mathcal{T}_{\boldsymbol{\mu}}$ describes the Logarithmic Map that projects an element $\mathbf{x} \in \mathcal{M}$ from the $\mathbb{S}^3$ onto the tangent space $\mathcal{T}_{\boldsymbol{\mu}}$ associated with point $\boldsymbol{\mu} \in \mathcal{M}$. The Exponential Map $\mathbf{Exp}_{\boldsymbol{\mu}}(\mathbf{u}) : \mathcal{T}_{\boldsymbol{\mu}} \to \mathcal{M}$ projects an element $\mathbf{u} \in \mathcal{T}_{\boldsymbol{\mu}}$ from the tangent space $\mathcal{T}_{\boldsymbol{\mu}}$ back onto $\mathbb{S}^3$. Additionally, the parallel transport operation $\Gamma_{\boldsymbol{\mu}_i \mapsto \boldsymbol{\mu}_j}(\mathbf{u}) : \mathcal{T}_{\boldsymbol{\mu}_i} \to \mathcal{T}_{\boldsymbol{\mu}_j}$ has to be applied, when combining geometric information from different tangent spaces. For further details, we refer the interested reader to [13] and [18].

**IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.**

### *B. Expectation Maximization on Riemannian Manifolds*

The set of model parameters $\mathbf{\Omega} = \{\pi_k, \boldsymbol{\mu}_k, \mathbf{\Sigma}_k\}_{k=1}^K$ for a GMM are commonly estimated by utilizing the Expectation Maximization (EM) algorithm, thereby maximizing the log-likelihood for the given set of demonstration data $\mathbf{\Upsilon}$

$$\log \mathcal{L}(\mathbf{\Upsilon}|\mathbf{\Omega}) = \log \prod_{n=1}^{N} \mathcal{P}(\boldsymbol{\xi}_n) = \sum_{n=1}^{N} \log \mathcal{P}(\boldsymbol{\xi}_n)\,. \tag{3}$$

It is an iterative procedure consisting of two steps. Starting from an initial set of parameters[1], an *expectation* step evaluates responsibilities $r_{k,n} \in \mathbb{R}$ of each component generating the data given the current set of model parameters $\mathbf{\Omega}$ [31]

$$r_{k,n} = \frac{\pi_k \mathcal{N}(\mathbf{Log}_{\boldsymbol{\mu}_k}(\mathbf{x}_n)|\boldsymbol{\mu}_k, \mathbf{\Sigma}_k)}{\sum_{i=1}^{K} \pi_i \mathcal{N}(\mathbf{Log}_{\boldsymbol{\mu}_i}(\mathbf{x}_n)|\boldsymbol{\mu}_i, \mathbf{\Sigma}_i)}\,. \tag{4a}$$

These are then used in a *maximization* step to compute new model parameters $\mathbf{\Omega}$. Prior probabilities $\pi_k$ are given by

$$\pi_k = \frac{\sum_{n=1}^{N} r_{k,n}}{N}\,, \tag{4b}$$

and means $\boldsymbol{\mu}_k$ are updated through Gauss-Newton iterations

$$\mathbf{u}_k = \frac{\sum_{n=1}^{N} r_{k,n} \mathbf{Log}_{\boldsymbol{\mu}_k}(\mathbf{x}_n)}{\sum_{n=1}^{N} r_{k,n}}\,,\ \boldsymbol{\mu}_k \leftarrow \mathbf{Exp}_{\boldsymbol{\mu}_k}(\mathbf{u}_k)\,. \tag{4c}$$

After convergence, new covariance matrices $\mathbf{\Sigma}_k$ are obtained by projecting all samples $\mathbf{x}_n$ from $\mathbf{\Upsilon}$ onto the tangent spaces $\mathcal{T}_{\boldsymbol{\mu}_k}$ associated with updated means $\boldsymbol{\mu}_k$

$$\mathbf{\Sigma}_k = \frac{\sum_{n=1}^{N} r_{k,n} \mathbf{Log}_{\boldsymbol{\mu}_k}(\mathbf{x}_n) \mathbf{Log}_{\boldsymbol{\mu}_k}(\mathbf{x}_n)^T}{\sum_{n=1}^{N} r_{k,n}}\,. \tag{4d}$$

The procedure is repeated until convergence, guaranteeing that the log-likelihood (3) is non-decreasing in each iteration.

### *C. Gaussian Mixture Regression on Riemannian Manifolds*

Gaussian Mixture Regression (GMR) is utilized to compute the most probable output $\mathbf{x} \in \mathcal{M}$ for an input $t \in \mathbb{R}$, i.e., estimating the conditional probability distribution [14]

$$\mathcal{P}(\mathbf{x}|t, \boldsymbol{\mu}, \mathbf{\Sigma}) = \frac{\mathcal{P}(t, \mathbf{x}|\boldsymbol{\mu}, \mathbf{\Sigma})}{\mathcal{P}(t|\boldsymbol{\mu}, \mathbf{\Sigma})} = \sum_{k=1}^{K} \hat{h}_k \mathcal{N}(\hat{\boldsymbol{\mu}}_k, \hat{\mathbf{\Sigma}}_k)\,. \tag{5}$$

The resulting distribution is described by $K$ Gaussians with mean vectors $\hat{\boldsymbol{\mu}}_k \in \mathcal{M}$ and uncertainty covariance matrices $\hat{\mathbf{\Sigma}}_k \in (\mathcal{T}_{\hat{\boldsymbol{\mu}}_k} \times \mathcal{T}_{\hat{\boldsymbol{\mu}}_k})$, where each Gaussian's contribution is given by activation weights $0 \leq \hat{h}_k \leq 1$. The multi-modal distribution (5) is often approximated by a single Gaussian $\mathcal{N}(\hat{\boldsymbol{\mu}}, \hat{\mathbf{\Sigma}}) \approx \mathcal{P}(\mathbf{x}|t, \boldsymbol{\mu}, \mathbf{\Sigma})$ [8]. Decomposing means $\boldsymbol{\mu}_k$ and covariances $\mathbf{\Sigma}_k$ into input and output dimensions

$$\boldsymbol{\mu}_k = \begin{bmatrix} \mu_k^t \\ \boldsymbol{\mu}_k^{\mathbf{x}} \end{bmatrix} \quad , \quad \mathbf{\Sigma}_k = \begin{bmatrix} \Sigma_k^{tt} & \mathbf{\Sigma}_k^{t\mathbf{x}} \\ \mathbf{\Sigma}_k^{t\mathbf{x}} & \mathbf{\Sigma}_k^{\mathbf{xx}} \end{bmatrix}\,, \tag{6}$$

GMM parameters are directly related to $\hat{h}_k$, $\hat{\boldsymbol{\mu}}$ and $\hat{\mathbf{\Sigma}}$. First, activation weights $\hat{h}_k$ are computed through

$$\hat{h}_k = \frac{\pi_k \mathcal{N}(t|\mu_k^t, \Sigma_k^{tt})}{\sum_{i=1}^{K} \pi_i \mathcal{N}(t|\mu_i^t, \Sigma_i^{tt})} \tag{7a}$$

which are used to approximate $\hat{\boldsymbol{\mu}}$ and $\hat{\mathbf{\Sigma}}$. Starting from an initial estimate $\hat{\boldsymbol{\mu}}$[2], covariance matrices $\mathbf{\Sigma}_k$ are expressed in the tangent space $\mathcal{T}_{\hat{\boldsymbol{\mu}}}$ by utilizing the parallel transport operation $\mathbf{E}_k = \Gamma_{\boldsymbol{\mu}_k \mapsto \hat{\boldsymbol{\mu}}}(\mathbf{\Sigma}_k)$, to compensate for the relative rotation between tangent spaces. Then, all Gaussian's predictions $\hat{\mathbf{u}}_k$ are locally evaluated in the tangent space of $\hat{\boldsymbol{\mu}}$

$$\hat{\mathbf{u}}_k = \mathbf{Log}_{\hat{\boldsymbol{\mu}}}(\boldsymbol{\mu}_k^{\mathbf{x}}) + \mathbf{E}_k^{t\mathbf{x}} {E_k^{tt}}^{-1}(t - \mu_k^t)\,, \tag{7b}$$

which are then approximated into a single prediction $\hat{\mathbf{u}}$ that is used to update $\hat{\boldsymbol{\mu}}$ through few Gauss-Newton iterations

$$\hat{\mathbf{u}} = \textstyle\sum_{k=1}^{K} \hat{h}_k \hat{\mathbf{u}}_k \quad , \quad \hat{\boldsymbol{\mu}} \leftarrow \mathbf{Exp}_{\hat{\boldsymbol{\mu}}}(\hat{\mathbf{u}})\,. \tag{7c}$$

After convergence, uncertainties $\hat{\mathbf{\Sigma}}_k$ are computed locally in the tangent space of $\hat{\boldsymbol{\mu}}$ for all Gaussians through

$$\hat{\mathbf{\Sigma}}_k = \mathbf{E}_k^{\mathbf{xx}} - \mathbf{E}_k^{t\mathbf{x}} {E_k^{tt}}^{-1} \mathbf{E}_k^{t\mathbf{x}}\,, \tag{7d}$$

which are then approximated by the single matrix $\hat{\mathbf{\Sigma}}$

$$\hat{\mathbf{\Sigma}} = \textstyle\sum_{k=1}^{K} \hat{h}_k \left(\hat{\mathbf{\Sigma}}_k + \hat{\mathbf{u}}_k \hat{\mathbf{u}}_k^T\right) - \hat{\mathbf{u}} \hat{\mathbf{u}}^T\,. \tag{7e}$$

### *D. Task-Parameterization*

To improve interpolation and extrapolation capabilities of the learned model, the Task-Parameterized Gaussian Mixture Model (TP-GMM) [21] utilizes $F > 1$ reference frames to create a mixture of experts for a variable skill. These frames are referred to as task-parameters $\{\mathbf{A}_f, \mathbf{b}_f\}_{f=1}^F$ that are specified by positions $\mathbf{b}_f$ and orientations $\mathbf{A}_f$. They describe relevant parts of the motion skill, e.g., start, via or target poses, and allow projection of the demonstration data $\mathbf{\Upsilon}$ from the global frame onto different local viewpoints $\mathbf{\Upsilon}_f$. Local GMMs are trained, and the model parameters become $\{\pi_k, \{{}^f\boldsymbol{\mu}_k, {}^f\mathbf{\Sigma}_k\}_{f=1}^F\}_{k=1}^K$, under the assumption that all data samples were generated from the same source [8].

During run-time, new task-parameters that describe the current scenario are supplied. Then, for each local perspective $f \in \{1, \ldots, F\}$, predictions $\mathcal{N}({}^f\tilde{\boldsymbol{\mu}}, {}^f\tilde{\mathbf{\Sigma}})$ are computed according to (7b)-(7e). Utilizing the task-parameters, these predictions are projected into the global perspective $\mathcal{N}({}^f\hat{\boldsymbol{\mu}}, {}^f\hat{\mathbf{\Sigma}})$ through

$${}^f\hat{\boldsymbol{\mu}} = \mathbf{Exp}_{\mathbf{A}_f}(\mathbf{Log}_{\mathbf{I}}({}^f\tilde{\boldsymbol{\mu}}))\,, \tag{8a}$$

$${}^f\hat{\mathbf{\Sigma}} = \Gamma_{{}^f\tilde{\boldsymbol{\mu}} \mapsto {}^f\hat{\boldsymbol{\mu}}}({}^f\tilde{\mathbf{\Sigma}})\,, \tag{8b}$$

where $\mathbf{I}$ describes the identity orientation.

With all predictions being expressed in the same global reference frame, the information from each perspective is combined using the Product of Gaussians. A single mean $\hat{\boldsymbol{\mu}}$ is computed through Gauss-Newton iterations according to

$$\mathbf{E}_\Sigma = \textstyle\sum_{f=1}^{F} \left({}^f\mathbf{E}^{-1}\right)\,, \tag{9a}$$

$$\mathbf{u}_\Sigma = \textstyle\sum_{f=1}^{F} \left({}^f\mathbf{E}^{-1} \mathbf{Log}_{\hat{\boldsymbol{\mu}}}({}^f\hat{\boldsymbol{\mu}})\right)\,, \tag{9b}$$

$$\hat{\mathbf{u}} = {\mathbf{E}_\Sigma}^{-1} \hat{\mathbf{u}}_\Sigma \quad , \quad \hat{\boldsymbol{\mu}} \leftarrow \mathbf{Exp}_{\hat{\boldsymbol{\mu}}}(\hat{\mathbf{u}})\,, \tag{9c}$$

where ${}^f\mathbf{E} = \Gamma_{{}^f\hat{\boldsymbol{\mu}} \mapsto \hat{\boldsymbol{\mu}}}({}^f\hat{\mathbf{\Sigma}})$ describes the uncertainty covariance matrices ${}^f\hat{\mathbf{\Sigma}}$, which have been parallel transported from tangent spaces $\mathcal{T}_{{}^f\hat{\boldsymbol{\mu}}}$ to $\mathcal{T}_{\hat{\boldsymbol{\mu}}}$. After convergence, a single uncertainty $\hat{\mathbf{\Sigma}} = {\mathbf{E}_\Sigma}^{-1}$ is computed.

[1]For example, obtained from random sampling, $K$-means, or $K$-bins.

[2]For example, mean $\boldsymbol{\mu}_k^{\mathbf{x}}$ with highest activation $\hat{h}_k$, or previous solutions.

**IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.**

## IV. Enforcing Time-Sensitive Constraints

Let us define a trajectory as $\{(t_m, \mathbf{x}_m)^T\}_{m=1}^M$ with $M$ time-indexed poses. Many industrial applications require the robot's end-effector to start and stop exactly at pre-defined times at the desired start and target pose. However, this can not be guaranteed when learning a time-dependent system utilizing standard GMM and GMR. In this paper, we propose to include task knowledge into the learning process by enforcing time-sensitive constraints (TSC). This allows us to generate trajectories that pass through desired poses, without effecting the computational complexity during run-time.

### A. Learning a GMM subject to Constraints

To enforce a desired pose $\mathbf{x}_{\text{des}}$ at time $t_{\text{des}}$, we investigate the single-Gaussian approximation of the conditional probability distribution. According to (7c), the approximated mean $\hat{\boldsymbol{\mu}}$ results from the sum of each Gaussian's prediction $\hat{\mathbf{u}}_k$ multiplied by an activation weight $\hat{h}_k$. Therefore, as shown in Lemma A.1, we aim for a model with conditions

$$\hat{h}_\lambda \approx 1 \quad , \quad \hat{h}_k \approx 0, \forall k \neq \lambda \,, \tag{10a}$$

$$\hat{\boldsymbol{\mu}} = \mathbf{x}_{\text{des}} \,, \tag{10b}$$

where $\lambda \in \{1, \dots, K\}$ describes the index of the Gaussian component that is used to enforce the desired pose. Before describing our novel approach, we present the main theorem here with supporting lemmas in the appendix.

**Theorem IV.1.** *Given $\mu_\lambda^t = t_{des}$ and $\boldsymbol{\mu}_\lambda^{\mathbf{x}} = \mathbf{x}_{des}$, there exists a set of $K$ scaling factors $\gamma_k \in (0, 1]$, i.e. $\gamma_k \Sigma_k^{tt}, \forall k \in K$, such that activations become $\hat{h}_\lambda \approx 1$, $\hat{h}_k \approx 0, \forall k \neq \lambda$ for a given input $t = t_{des}$.*

*Proof.* If $\mathcal{N}(t|\mu_\lambda^t, \Sigma_\lambda^{tt}) \gg \mathcal{N}(t|\mu_k^t, \Sigma_k^{tt}), \forall k \neq \lambda$, it holds that $\hat{h}_\lambda \approx 1$, $\hat{h}_k \approx 0, \forall k \neq \lambda$ (see Lemma A.2). Given $\mu_\lambda^t = t_{\text{des}}$ and $\boldsymbol{\mu}_\lambda^{\mathbf{x}} = \mathbf{x}_{\text{des}}$, $\mathcal{N}(t|\mu_\lambda^t, \Sigma_\lambda^{tt})$ is maximized for $t = t_{\text{des}}$ (see Lemma A.3). Applying $\gamma_k \in (0, 1), \forall k \in K$ as a scaling factor to the covariance of a Gaussian increases $\mathcal{N}(t_{\text{des}}|\mu_\lambda^t, \Sigma_\lambda^{tt})$ and decreases $\mathcal{N}(t_{\text{des}}|\mu_k^t, \Sigma_k^{tt}), \forall k \neq \lambda$, thus increases $\hat{h}_\lambda$ (see Lemma A.4). Hence, scaling factors $\gamma_k \in (0, 1], \forall k \in K$ exist, such that overlapping regions of influence, i.e., $\hat{h}_k \gg 0, \exists k \neq \lambda$, are removed at $t = t_{\text{des}}$. Therefore, the $\lambda$-th component is activated with $\hat{h}_\lambda \approx 1$ and other components with $\hat{h}_k \approx 0, \forall k \neq \lambda$. □

*1) Proposed EM Modification:* We suggest a Constrained EM (CEM) algorithm that includes task knowledge into the learning process, as summarized in Algorithm 1. To find an optimal GMM with the desired properties specified in (10), we propose to modify the *maximization* step of the EM algorithm, such that the $\lambda$-th mean $\boldsymbol{\mu}_\lambda$ remains fixed in each iteration with $\mu_\lambda^t = t_{\text{des}}$ and $\boldsymbol{\mu}_\lambda^{\mathbf{x}} = \mathbf{x}_{\text{des}}$. Note that the property of the non-decreasing likelihood measure in each EM iteration is not affected by this modification.

After CEM has converged, we compute and apply scaling factors $\gamma_k \in (0, 1], \forall k \in K$ for each covariance matrix[3]. This modifies their activations $\hat{h}_k$ such that $\hat{h}_\lambda \geq 1 - (K-1)\epsilon$, $\hat{h}_k \leq \epsilon, \forall k \neq \lambda$, considering a small predefined threshold $\epsilon$ for numerical reasons. Appropriate scaling factors $\gamma_k$ are obtained by solving the optimization problem

$$\begin{aligned} \max_{\gamma_1, \dots, \gamma_K} \quad & \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{\xi}_n | \boldsymbol{\mu}_k, \gamma_k \boldsymbol{\Sigma}_k) \\ \text{s.t.} \quad & \left\{ \frac{\pi_k \mathcal{N}(\mu_\lambda^t | \mu_k^t, \gamma_k \Sigma_k^{tt})}{\sum_{i=1}^{K} \pi_i \mathcal{N}(\mu_i^t | \mu_i^t, \gamma_i \Sigma_i^{tt})} \leq \epsilon \right\}_{\forall k \neq \lambda} \\ & 0 < \gamma_k \leq 1 \quad \forall k \in K \end{aligned} \tag{11}$$

*2) Proposed Initialization:* Before executing the CEM, we need to specify the $\lambda$-th component that enforces the TSC. We sort the demonstration data based on timestamps and split all samples into $K$ equal bins. Based on these assignments, initial means and covariances are computed. The most suitable $\lambda$-th component is then identified through

$$\lambda = \underset{k}{\operatorname{argmin}} \left( \{ \| \boldsymbol{\mu}_k^{\mathbf{x}} - \mathbf{x}_{\text{des}} \| \}_{k=1}^{K} \right) \tag{12}$$

**Remark:** It is straightforward to extend the described approach for enforcing multiple desired poses at multiple given times, by constraining multiple Gaussian components.

**Algorithm 1** Constrained Expectation Maximization (CEM)

**Require:**
Initialized model parameters $\{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$
Set of demonstration data $\{\boldsymbol{\xi}_n\}_{n=1}^N$
Desired time $t_{\text{des}}$, pose $\mathbf{x}_{\text{des}}$ and threshold $\epsilon$ (TSC)
Index $\lambda$ of Gaussian component to enforce TSC

**Ensure:**
Model parameters $\{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}_{k=1}^K$ with enforced TSC

```
while not converged do
    for n ← 1 to N do                                   ▷ E-step
        for k ← 1 to K do
            Compute responsibility r_{k,n} (4a)
        end for
    end for
    for k ← 1 to K do                                   ▷ Modified M-step
        Compute prior probability π_k (4b)
        if k is λ then                                  ▷ Fix λ-th mean μ_λ
            Set μ_λ^t = t_des
            Set μ_λ^x = x_des
        else
            Compute mean μ_k (4c)
        end if
        Compute covariance Σ_k (4d)
    end for
end while
Solve optimization problem (11)                         ▷ Covariance scaling
```

### B. Learning a TP-GMM subject to Constraints

We here extend CEM to the TP-GMM framework. Note that due to the task-parameterized projection of data into local perspectives $\{\boldsymbol{\Upsilon}_f\}_{f=1}^F$, it is particularly well suited to enforce multiple desired poses ${}^f\mathbf{x}_{\text{des}}$ at given times ${}^f t_{\text{des}}$.

[3]We scale the full covariance matrix $\boldsymbol{\Sigma}_k$, as scaling $\Sigma_k^{tt}$ would eventually violate the SPD matrix property of $\boldsymbol{\Sigma}_k$, refer to Sylvester's criterion. All statements above remain valid.

**IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.**

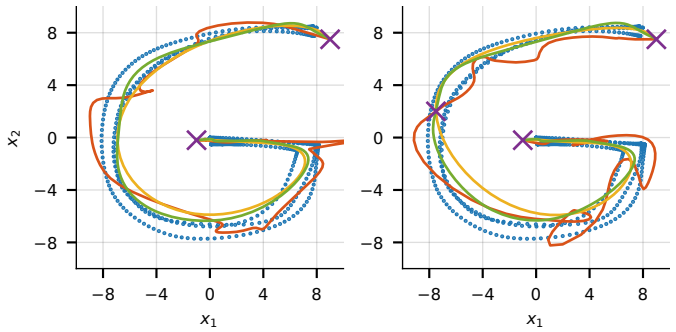


Fig. 2. Comparison between Bagging-GMM (red), KMP (yellow) and the proposed CEM (green) approach, when enforcing two (left) and three (right) time-sensitive constraints (purple).

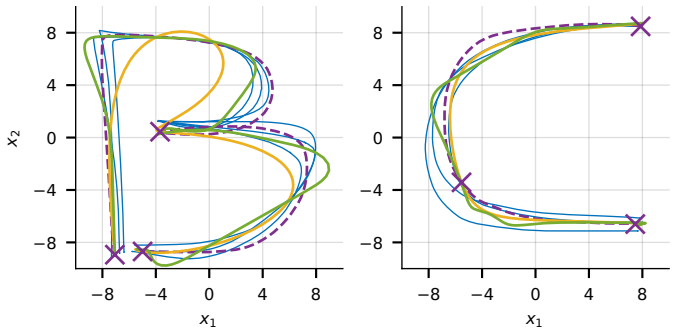


Fig. 3. Comparison between KMP (yellow) and CEM (green) in reproducing the fourth demonstration (purple) after learning a model from three demonstrations (blue) given three time-sensitive constraints (purple).

*1) Proposed Extension to Local Frames:* To enforce the desired pose ${}^f\mathbf{x}_{\text{des}}$ in the $f$-th local perspective, we identify the most suitable Gaussian component by evaluating

$$\lambda_f = \underset{k}{\operatorname{argmin}} \left( \{ \left\| {}^f\boldsymbol{\mu}_k^{\mathbf{x}} - {}^f\mathbf{x}_{\text{des}} \right\| \}_{k=1}^{K} \right) \tag{13}$$

similar to (12). Means ${}^f\boldsymbol{\mu}_{\lambda f}$ are fixed with ${}^f\mu_{\lambda f}^t = {}^f t_{\text{des}}$ and ${}^f\boldsymbol{\mu}_{\lambda f}^{\mathbf{x}} = {}^f\mathbf{x}_{\text{des}}$ when learning local GMMs for each perspective using CEM. This ensures ${}^f\hat{\boldsymbol{\mu}}_{\lambda f} = {}^f\mathbf{x}_{\text{des}}$ in local frames for corresponding inputs $t = {}^f t_{\text{des}}$.

*2) Proposed Fusion:* CEM enforces TSC in the local frames. To retain these properties in the global frame, we adapt the fusion of local predictions by introducing a weighting term $\omega(t, f) : \mathbb{R} \times \{1, \dots, F\} \mapsto [0, 1]$ in the Product of Gaussians. Given inputs $\{t_m\}_{m=1}^M$, $F$ corresponding poses $\{{}^f\hat{\boldsymbol{\mu}}_m\}_{m=1}^M$ are predicted for each local perspective through (7a)-(7e). When fusing these predictions utilizing (8a)-(9c), equations (9a) and (9b) become

$$\mathbf{E}_\Sigma = \textstyle\sum_{f=1}^F \left( \omega(t_m, f) {}^f\mathbf{E}^{-1} \right), \tag{14a}$$

$$\mathbf{u}_\Sigma = \textstyle\sum_{f=1}^F \left( \omega(t_m, f) {}^f\mathbf{E}^{-1} \mathbf{Log}_{\hat{\boldsymbol{\mu}}}({}^f\hat{\boldsymbol{\mu}}) \right), \tag{14b}$$

which adapts the importance of each perspective differently.

*3) Proposed Weighting:* Each local mean ${}^f\boldsymbol{\mu}_{\lambda f}$ enforces a desired pose ${}^f\mathbf{x}_{\text{des}}$ at a desired time ${}^f t_{\text{des}}$. Hence, for all inputs ${}^f t_{\text{des}}$, only one perspective has to be responsible when generating the global prediction to keep the desired properties, i.e., for each $f \in \{1, \dots, F\}$ it has to hold $\omega({}^f t_{\text{des}}, f) = 1 \implies \omega({}^f t_{\text{des}}, i) = 0, \forall i \neq f$. Therefore, we choose weightings $\omega(t, f) \in [0, 1]$ that generate linear transitions between two perspectives based on

$$\omega(t, f) = \begin{cases} \max\left(0 \ , \ 1 - \frac{\|t - \eta_f\|}{\eta_f - \eta_{f-1}}\right) & \text{if } t \leq \eta_f \\ \max\left(0 \ , \ 1 - \frac{\|t - \eta_f\|}{\eta_{f+1} - \eta_f}\right) & \text{if } t > \eta_f \end{cases}, \tag{15}$$

where $\eta_f = {}^f\mu_{\lambda_f}^t$ is used for clarity, and we assume that perspectives $f \in \{1, \dots, F\}$ are sorted, i.e., $\eta_1 < \dots < \eta_F$.

## V. Experiments

This section evaluates the proposed CEM approach by comparing it with baselines on handwritten data and real robot experiments using a KUKA LBR iiwa robot.

### A. Simulated Evaluation on Handwritten Data

In a first experiment, we evaluate CEM on 2D handwritten data, i.e., $\boldsymbol{\xi} = [t, x_1, x_2]^T$, with four demonstrations from the LASA data set [9] for each letter. As baselines, we chose Bagging-GMM [25] and KMP [15]. All models were trained with $K = 8$ Gaussians. For Bagging-GMM, 8 subsets of the data were created from random permutations to train the base learners. For KMP we utilized a penalty weighting $\lambda = 1$, desired precisions $\bar{\boldsymbol{\Sigma}}_m = 10^{-5}$ for all constraints and the Gaussian kernel with length scale $\ell = 6$.

*1) Reproduction of Letter G:* Bagging-GMM, KMP and CEM all allow to enforce TSC when reproducing the letter *G*. Fig. 2 shows the results, where we aim for two and three desired positions, respectively. Bagging-GMM successfully enforces all constraints, but the randomization of the data subsets may result in unexpected deviations from the demonstrated mean. The trajectory obtained from KMP and CEM accurately reproduces the data. Note, however, that CEM includes the constraints within the training process, which allows to utlize standard and efficient GMR for inference. All other methods enforce the constraints at inference time, resulting in a significantly increased computational load compared to CEM as shown in Table I. Also, CEM outperforms in terms of combined training and inference time.

*2) Quantitative Analysis:* Next, we compare KMP and CEM on the full LASA dataset. For each letter, three demonstrations are used to train the models. The result is then compared to a fourth demonstration, to analyze the reproduction error. In order to receive more general results, this process is repeated four times, such that each demonstration has been reproduced once, where the first, middle and last points of that demonstration, i.e., points at $t_{\text{des}} \in \{t_1, t_{\frac{N}{2}}, t_N\}$, have been used as constraints.

Table II and Table III show that CEM outperforms KMP.

TABLE I
Average Training ($N = 600$) and Inference ($N = 200$) time

| Method | Bagging | KMP | CEM |
|---|---|---|---|
| Training time [s] | 0.82 | 0.54 | **0.21** |
| Inference time [s] | 2.89 | 1.48 | **0.35** |
| Total time [s] | 3.71 | 2.02 | **0.56** |

**IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.**

However, CEM struggles in reproducing curved motions, e.g., see Fig 3. During GMR, linear predictions from each Gaussian are combined through non-linear weighting. This becomes especially visible for the CEM approach when reproducing letter *C*. KMP on the other hand is not operating on the demonstration data directly, but utilizes a probabilistic encoding instead (e.g. from GMR). This may lead to larger deviations from the mean trajectory as seen for letter *B*.

*3) Hyper-Parameter Tuning:* Finally, we investigate the hyper-parameter tuning for KMP and CEM. Both methods require a number of $K$ Gaussians to be defined. For CEM, the only additional parameter is the threshold $\epsilon$ that is used during optimization for numerical reasons. The parameters of KMP are the penalty weighting $\lambda$, desired precision $\bar{\mathbf{\Sigma}}_m$ for each $m$-th TSC, and the parameters of the kernel, e.g., the length scale $\ell$ for the Gaussian kernel. Fig. 4 shows an example for the letter *B*, evaluating how accurate the three TSC have been enforced. While the parameter $\epsilon$ in the CEM approach is intuitive to tune, i.e., a decreased threshold enforces the TSC more strictly, finding optimal parameters for KMP is more complex. Furthermore, a decreased penalty weighting $\lambda$ results in a solution closer to the data's mean, but may result in numerical instability.

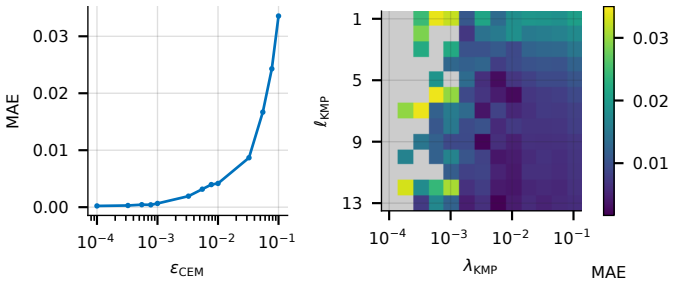


Fig. 4. Evaluation of the mean absolute error (MAE) for three TSC enforced in reproducing the letter *B* utilizing CEM (threshold $\epsilon$) on the left and KMP (length scale $\ell$ and penalty weighting $\lambda$) on the right. For KMP, values $> 0.35$ have been clamped (gray) for improved visualization.

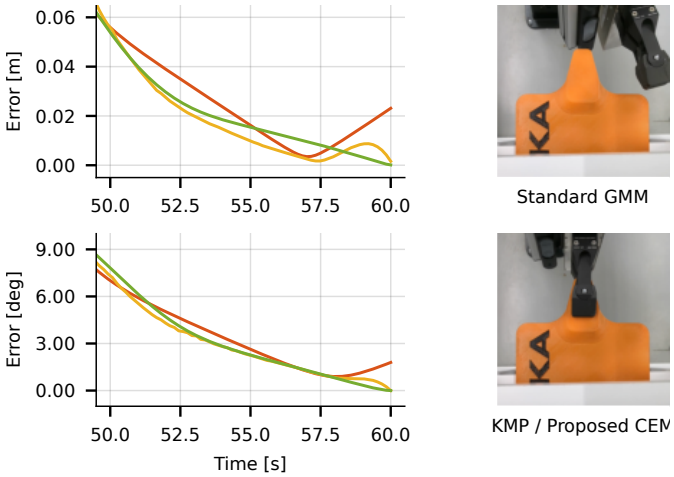


Fig. 5. Results of the grasping experiment. Position and orientation errors resulting from GMM (red), KMP (yellow) and CEM (green) are shown on the left. The final end-effector pose is shown on the right for all methods.

## B. Evaluation in Real Robot Applications

Finally, we apply CEM in two real robot applications, 1) a grasping task, and 2) an inspection task with Task-Parameterization. Demonstrations were recorded for the end-effector at 30 Hz utilizing hand-guiding, and their duration was normalized to 60 seconds. Learned end-effector trajectories are executed utilizing state-of-the-art QP-control [34].

*1) Grasping Task in* $\mathrm{SE}(3)$*:* We chose standard GMM [4] and KMP [15] as baselines. for KMP, we use the same parameters as in subsection V-A, but set $\ell = 0.1$ as it depends on the input domain. Four demonstrations are recorded, where the end-effector approaches a tablet from the side and grasps it at its handle. Data points are projected onto the frame of the target pose and all models are trained with $K = 5$. KMP and CEM enforce a TSC at $t_{\text{des}} = 60$ and $\mathbf{x}_{\text{des}} = \mathbf{I}$. During motion execution, predicted trajectories are converted back into the global frame. The resulting error to the target is shown in Fig. 5. As expected, utilizing standard GMM, the prediction deviates from the desired target in both position and orientation. For both, KMP and CEM, the generated trajectory stops precisely at the desired target pose, enabling the robot to grasp the tablet successfully.

TABLE II
REPRODUCTION PERFORMANCE ON THE FULL LASA DATASET

| | $K = 8$ | $K = 12$ | $K = 16$ |
|---|---|---|---|
| CEM outperforms KMP | **16 (62%)** | **21 (81%)** | **23 (88%)** |
| KMP outperforms CEM | 10 (38%) | 5 (19%) | 3 (12%) |
| Average error KMP | 0.48 | 0.45 | 0.43 |
| Average error CEM | **0.42** | **0.33** | **0.31** |

TABLE III
REPRODUCTION ERROR FOR THE FULL LASA DATASET WITH $K = 8$

| Letter | KMP | CEM | Letter | KMP | CEM |
|---|---|---|---|---|---|
| A | 0.59 | **0.47** | B | 0.91 | **0.63** |
| C | **0.29** | 0.33 | D | 0.33 | **0.29** |
| E | **0.47** | 0.48 | F | 0.51 | **0.47** |
| G | **0.43** | 0.45 | H | 0.67 | **0.41** |
| I | **0.33** | 0.37 | J | **0.28** | 0.32 |
| K | 0.86 | **0.50** | L | **0.15** | 0.17 |
| M | 0.81 | **0.61** | N | 0.40 | **0.36** |
| O | **0.38** | 0.40 | P | **0.31** | 0.37 |
| Q | 0.63 | **0.55** | R | 0.69 | **0.43** |
| S | **0.30** | 0.46 | T | 0.43 | **0.29** |
| U | **0.35** | 0.45 | V | 0.37 | **0.34** |
| W | 0.53 | **0.50** | X | 0.64 | **0.59** |
| Y | 0.44 | **0.29** | Z | 0.48 | **0.44** |

*2) Inspection Task with Task-Parameterization in* $\mathrm{SE}(3)$*:* We chose TP-GMM [21] and TP-KMP [15] as baselines. Note that TP-KMP is not available for $\mathrm{SE}(3)$, wherefore, to conduct meaningful comparisons, we had to formulate and implement the extension ourselves. First, we enforce the constraints within local KMPs for each perspective as in [15], [26]. Local predictions are then transported and fused similar to (8a)-(9c). For numerical reasons we utilized $\lambda = 4$. Furthermore, we include a modified version TP-KMP$^*$, that utilizes the proposed fusion (14a), (14b) and weighting (15).

For the inspection task (see Fig. 1), six demonstrations are provided with $F = 3$ task-parameters representing the start,

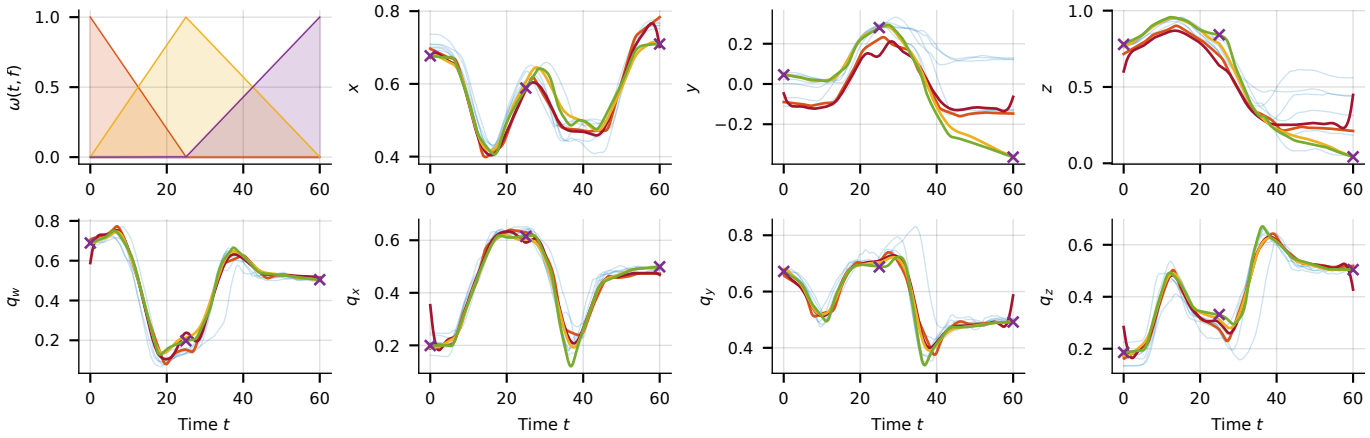


Fig. 6. Results for the inspection task with TP-GMM (red), TP-KMP (brown), modified TP-KMP* (yellow) and the proposed TP-CEM approach (green). Desired task-parameters defined at run-time are marked in purple. Weightings $\omega_{t,f}$ during fusion for each local perspective are visualized at the top left.

camera inspection and target pose. All models are trained with $K = 9$, and a TSC is enforced at the origin of each local perspective at ${}^{f}t_{\text{des}} \in \{0, 25, 60\}$ utilizing TP-KMP and TP-CEM. As shown in Fig. 6, the trajectory obtained from TP-CEM succeeds in reaching all three desired poses. TP-KMP enforces the constraints within the local models, but this property is lost during fusion. Results are improved by adapting the proposed fusion and weighting as in TP-KMP*. However, due to the penalty term required for numerical reasons in TP-KMP and TP-KMP*, the constraints are not enforced as strictly as with TP-CEM, see Table IV.

## VI. Conclusion

In this letter, we propose the novel *Constrained Expectation Maximization* algorithm to enforce *time-sensitive constraints* during learning of a GMM, by constraining dedicated model parameters in the *maximization*-step, and optimizing covariance scalings to enforce exclusive activation of specific components at desired times. The novel CEM algorithm guarantees TSC in probabilistic imitation learning. Contrary to previous works, these constraints are considered within the learning process, instead of enforcing them at inference-time, allowing other model parameters to adapt. By adjusting the fusion of local predictions in the context of Task-Parameterization, the enforced properties are retained in the global frame. Experiments on handwritten data and real-robot applications validate our approach.

Note that the proposed CEM approach effectively enforces TSC within the learning process, but it does not directly address variance during run-time beyond Task-Parameterization, e.g., perception noise. However, as KMP builds upon a probabilistic encoding of the demonstration data, e.g. from GMR, it could potentially benefit from utilizing CEM instead of standard GMM, allowing to leverage the advantages of both methods and handle varying variances during run-time.

TABLE IV
Avg. Deviation from Desired Poses for the Inspection Task

| | TP-GMM | TP-KMP | TP-KMP* | TP-CEM |
|---|---|---|---|---|
| Error [cm] | 19.387 | 31.629 | 2.7190 | **0.0082** |
| Error [deg] | 5.0787 | 15.015 | 1.8707 | **> 0.0000** |

## Appendix

Here we present the lemmas for the proof of Theorem IV.1.

**Lemma A.1.** *If $\hat{h}_\lambda \approx 1$, $\hat{h}_k \approx 0, \forall k \neq \lambda$, $\mu_\lambda^t = t_{des}$ and $\boldsymbol{\mu}_\lambda^{\mathbf{x}} = \mathbf{x}_{des}$, then $\hat{\boldsymbol{\mu}} = \mathbf{x}_{des}$ for a given input $t = t_{des}$.*

*Proof.* According to (7b), for the $\lambda$-th Gaussian it yields $\hat{\mathbf{u}}_\lambda = \mathbf{Log}_{\hat{\boldsymbol{\mu}}}(\boldsymbol{\mu}_\lambda^{\mathbf{x}}) + \mathbf{E}_\lambda^{t\mathbf{x}} E_\lambda^{tt\,-1}(t_{\text{des}} - \mu_\lambda^t) = \mathbf{Log}_{\hat{\boldsymbol{\mu}}}(\boldsymbol{\mu}_\lambda^{\mathbf{x}})$. Furthermore, given (7c) it yields $\hat{\mathbf{u}} = \sum_{k=1}^K \hat{h}_k \hat{\mathbf{u}}_k \approx \hat{\mathbf{u}}_\lambda$ and $\hat{\boldsymbol{\mu}} \leftarrow \mathbf{Exp}_{\hat{\boldsymbol{\mu}}}(\hat{\mathbf{u}}) \approx \mathbf{Exp}_{\hat{\boldsymbol{\mu}}}(\mathbf{Log}_{\hat{\boldsymbol{\mu}}}(\boldsymbol{\mu}_\lambda^{\mathbf{x}}))$. Hence, the Gauss Newton iterations will converge to $\hat{\boldsymbol{\mu}} = \mathbf{x}_{\text{des}}$. □

**Lemma A.2.** *Component $\lambda$ is activated with $\hat{h}_\lambda \approx 1$ and $\hat{h}_k \approx 0, \forall k \neq \lambda$, if $\mathcal{N}(t|\mu_\lambda^t, \Sigma_\lambda^{tt}) \gg \mathcal{N}(t|\mu_k^t, \Sigma_k^{tt}), \forall k \neq \lambda$.*

*Proof.* Given $\mathcal{N}(t|\mu_\lambda^t, \Sigma_\lambda^{tt}) \gg \mathcal{N}(t|\mu_k^t, \Sigma_k^{tt}), \forall k \neq \lambda$, the ratio in (7a) for determining the activation weight $\hat{h}_\lambda$ is dominated by $\mathcal{N}(t|\mu_\lambda^t, \Sigma_\lambda^{tt})$. For the $\lambda$-th component holds

$$\hat{h}_\lambda = \frac{\pi_\lambda \mathcal{N}(t|\mu_\lambda^t, \Sigma_\lambda^{tt})}{\sum_{i=1}^K \pi_i \mathcal{N}(t|\mu_i^t, \Sigma_i^{tt})} \approx \frac{\pi_\lambda \mathcal{N}(t|\mu_\lambda^t, \Sigma_\lambda^{tt})}{\pi_\lambda \mathcal{N}(t|\mu_\lambda^t, \Sigma_\lambda^{tt})} \approx 1$$

and for all other components $k \neq \lambda$ holds

$$\hat{h}_k = \frac{\pi_k \mathcal{N}(t|\mu_k^t, \Sigma_k^{tt})}{\sum_{i=1}^K \pi_i \mathcal{N}(t|\mu_i^t, \Sigma_i^{tt})} \approx \frac{\pi_k \mathcal{N}(t|\mu_k^t, \Sigma_k^{tt})}{\pi_\lambda \mathcal{N}(t|\mu_\lambda^t, \Sigma_\lambda^{tt})} \approx 0$$

□

In the following, let us denote

$$\mathcal{N}(t|\mu_k^t, \Sigma_k^{tt}) = \frac{1}{\alpha_k} \exp(\beta_k) \in \mathbb{R} \tag{16}$$

with a constant factor $\alpha_k = \sqrt{2\pi \det(\Sigma_k^{tt})}$ and the exponential argument $\beta_k = -\frac{1}{2}(t-\mu_k^t)^T \Sigma_k^{tt\,-1}(t-\mu_k^t)$ for Euclidean inputs $t \in \mathbb{R}$. Recall that the covariance $\boldsymbol{\Sigma}_k$ is a SPD matrix, i.e., $\det(\boldsymbol{\Sigma}_k) > 0$, hence also $\det(\Sigma_k^{tt}) > 0$ and $\alpha_k > 0$. The scalar $\beta_k$ constitutes a negative squared distance between input $t$ and mean $\mu_k^t$ (which is scaled by $\frac{1}{2}\Sigma_k^{tt\,-1} > 0$). It holds $\beta_k \leq 0$ and $0 < \exp(\beta_k) \leq 1$.

**IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.**

**Lemma A.3.** *If $\mu_\lambda^t = t_{des}$ and $\boldsymbol{\mu}_\lambda^{\mathbf{x}} = t_{des}$ (see Lemma A.1), then $\mathcal{N}(t|\mu_\lambda^t, \Sigma_\lambda^{tt})$ reaches its maximum at $t = t_{des}$.*

*Proof.* For $t = t_{\text{des}}$, it holds $\beta_\lambda = 0$ and $\exp(\beta_\lambda) = 1$. Furthermore, $\alpha_\lambda > 0$ is a constant positive factor. Therefore, $\mathcal{N}(t|\mu_\lambda^t, \Sigma_\lambda^{tt})$ has its maximum at $t = t_{\text{des}}$. □

**Lemma A.4.** *Given $\boldsymbol{\mu}_\lambda^{\mathbf{x}} = \mathbf{x}_{des}$ and $\mu_\lambda^t = t_{des}$ as discussed in Lemma A.1. For any of the $K$ Gaussian components, multiplying $\Sigma_k^{tt}$ in covariance $\boldsymbol{\Sigma}_k$ with a scaling factor $\gamma_k \in (0, 1)$ increases the $\lambda$-th activation $\hat{h}_\lambda$ at $t = t_{des}$.*

*Proof.* Note that $(\gamma_k \Sigma_k^{tt})^{-1} = \frac{1}{\gamma_k} {\Sigma_k^{tt}}^{-1}$ and due to $0 < \gamma_k < 1$ it yields $\frac{1}{\gamma_k} > 1$. Also, with $\beta_k \leq 0$ it holds $\exp(\frac{\beta_k}{\gamma_k}) \leq \exp(\beta_k)$. We have $\det(\gamma_k \Sigma_k^{tt}) = \gamma_k \det(\Sigma_k^{tt})$. Hence, the constant factor $\frac{1}{\alpha_k}$ in (16) becomes $\frac{1}{\sqrt{\gamma_k}\alpha_k} > \frac{1}{\alpha_k}$. In the following, we distinguish two cases, $k = \lambda$ and $k \neq \lambda$.

Consider first $k = \lambda$: For $t_{\text{des}}$, the $\lambda$-th component has an exponential argument $\beta_\lambda = 0$. Therefore, it holds $\mathcal{N}(t_{\text{des}}|\mu_\lambda^t, \gamma_\lambda \Sigma_\lambda^{tt}) > \mathcal{N}(t_{\text{des}}|\mu_\lambda^t, \Sigma_\lambda^{tt})$.

Consider now $k \neq \lambda$ (and $\mu_k^t \neq t_{\text{des}}$): For $t = t_{\text{des}}$ the exponential argument $\beta_k < 0$ is strictly negative and after scaling it yields $\gamma_k \beta_k < \beta_k < 0$. Therefore, it holds $\mathcal{N}(t_{\text{des}}|\mu_k^t, \gamma_k \Sigma_k^{tt}) < \mathcal{N}(t_{\text{des}}|\mu_k^t, \Sigma_k^{tt}), \forall k \neq \lambda$.

Thus, in both cases $k = \lambda$ and $k \neq \lambda$, a scaling factor $\gamma_k \in (0, 1)$ causes activation $\hat{h}_\lambda$ to increase at $t = t_{\text{des}}$. □

## REFERENCES


[1] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annu. Rev. Control, Robot. and Auton. Syst.*, vol. 3, no. 1, pp. 297–330, 2020.

[2] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, *Robot Programming by Demonstration*, pp. 1371–1394. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.

[3] A. Billard and D. Grollman, *Imitation Learning in Robots*, pp. 1494–1496. Boston, MA: Springer US, 2012.

[4] S. Calinon, F. Guenter, and A. Billard, "On learning, representing, and generalizing a task in a humanoid robot," *IEEE Trans. Systems, Man, and Cybernetics, Part B*, vol. 37, no. 2, pp. 286–298, 2007.

[5] P. Kormushev, S. Calinon, and D. G. Caldwell, "Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input," *Advanced Robotics*, vol. 25, no. 5, pp. 581–603, 2011.

[6] H.-I. Lin and Y.-H. Lin, "A novel teaching system for industrial robots," *Sensors*, vol. 14, no. 4, pp. 6012–6031, 2014.

[7] D. Vogt, S. Stepputtis, S. Grehl, B. Jung, and H. Ben Amor, "A system for learning continuous human-robot interactions from human-human demonstrations," 05 2017.

[8] S. Calinon, "A tutorial on task-parameterized movement learning and retrieval," *Intelligent Service Robotics*, vol. 9, pp. 1–29, 2016.

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

[10] K. Neumann and J. Steil, "Learning robot motions with stable dynamical systems under diffeomorphic transformations," *Robotics and Autonomous Systems*, vol. 70, 04 2015.

[11] M. Saveriano, F. J. Abu-Dakka, and V. Kyrki, "Learning stable robotic skills on Riemannian manifolds," *Robotics and Autonomous Systems*, vol. 169, p. 104510, 2023.

[12] J. Silvério, L. Rozo, S. Calinon, and D. G. Caldwell, "Learning bimanual end-effector poses from demonstrations using task-parameterized dynamical systems," in *2015 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 464–470, 2015.

[13] M. J. Zeestraten, I. Havoutis, J. Silvério, S. Calinon, and D. G. Caldwell, "An approach for imitation learning on Riemannian manifolds," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1240–1247, 2017.

[14] Z. Ghahramani and M. Jordan, "Supervised learning from incomplete data via an EM approach," *Advances in Neural Information Processing Systems*, vol. 6, 1993.

[15] Y. Huang, L. Rozo, J. Silvério, and D. Caldwell, "Kernelized movement primitives," *The International Journal of Robotics Research*, vol. 38, pp. 833–852, 05 2019.

[16] S. Schaal, "Dynamic movement primitives - A framework for motor control in humans and humanoid robots," in *Proc. Int. Symp. on Adaptive Motion of Animals and Machines*, pp. 261–280, Mar. 2003.

[17] A. Paraschos, C. Daniel, J. R. Peters, and G. Neumann, "Probabilistic movement primitives," *Advances in Neural Information Processing Systems*, vol. 26, 2013.

[18] S. Calinon, "Gaussians on Riemannian manifolds: Applications for robot learning and adaptive control," *IEEE Robotics & Automation Magazine*, vol. 27, no. 2, pp. 33–45, 2020.

[19] N. Perrin and P. Schlehuber-Caissier, "Fast diffeomorphic matching to learn globally asymptotically stable nonlinear dynamical systems," *Syst. Control. Lett.*, vol. 96, pp. 51–59, 2016.

[20] N. Figueroa and A. Billard, "A physically-consistent bayesian non-parametric mixture model for dynamical system learning," in *Proceedings of The 2nd Conf. on Robot Learning* (A. Billard, A. Dragan, J. Peters, and J. Morimoto, eds.), vol. 87 of *Proceedings of Machine Learning Research*, pp. 927–946, PMLR, 29–31 Oct 2018.

[21] S. Calinon, D. Bruno, and D. G. Caldwell, "A task-parameterized probabilistic model with minimal intervention control," in *IEEE Int. Conf. Robotics and Automation*, pp. 3339–3344, 2014.

[22] Y. Huang, J. Silvério, L. Rozo, and D. G. Caldwell, "Generalized task-parameterized skill learning," in *IEEE Int. Conf. Robotics and Automation*, pp. 5667–5474, 2018.

[23] T. Alizadeh and M. Malekzadeh, "Identifying the relevant frames of reference in programming by demonstration using task-parameterized Gaussian mixture regression," in *2016 IEEE/SICE Int. Symp. on System Integration*, pp. 453–458, 2016.

[24] J. Sun, J. Kober, M. Gienger, and J. Zhu, "Learning from few demonstrations with frame–weighted motion generation," in *Int. Symp. Experimental Robotics*, pp. 32–41, 2023.

[25] C. Ye, J. Yang, and H. Ding, "Bagging for Gaussian mixture regression in robot learning from demonstration," *Journal of Intelligent Manufacturing*, vol. 33, no. 3, pp. 867–879, 2022.

[26] Y. Huang, F. Abu-Dakka, J. Silvério, and D. G. Caldwell, "Generalized orientation learning in robot task space," in *IEEE Int. Conf. on Robotics and Automation*, (Montreal, Canada), pp. 2531–2537, May 2019.

[27] F. J. Abu-Dakka, Y. Huang, J. Silvério, and V. Kyrki, "A probabilistic framework for learning geometry-based robot manipulation skills," *Robotics and Autonomous Systems*, vol. 141, p. 103761, 2021.

[28] M. Knauer, A. Albu-Schäffer, F. Stulp, and J. Silvério, "Interactive incremental learning of generalizable skills with local trajectory modulation," *IEEE Robotics and Automation Letters*, 2025.

[29] K. Neumann, M. Rolf, and J. Steil, "Reliable integration of continuous constraints into extreme learning machines," *Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 21, pp. 35–50, 12 2013.

[30] K. Ganchev, B. Taskar, and J. a. Gama, "Expectation maximization and posterior constraints," in *Advances in Neural Information Processing Systems* (J. Platt, D. Koller, Y. Singer, and S. Roweis, eds.), vol. 20, Curran Associates, Inc., 2007.

[31] C. McFarlane, G. Raheja, C. Malings, E. K. Appoh, A. F. Hughes, and D. M. Westervelt, "Application of Gaussian mixture regression for the correction of low cost pm2.5 monitoring data in accra, ghana," *ACS Earth and Space Chemistry*, vol. 5, no. 9, pp. 2268–2279, 2021.

[32] S. Kim, R. Haschke, and H. Ritter, "Gaussian mixture model for 3-dof orientations," *Robotics and Autonomous Systems*, vol. 87, pp. 28–37, 2017.

[33] X. Pennec, P. Fillard, and N. Ayache, "A riemannian framework for tensor computing," *Int. Journal of Computer Vision*, vol. 66, pp. 41–66, 2006.

[34] K. Bouyarmane, K. Chappellet, J. Vaillant, and A. Kheddar, "Quadratic programming for multirobot and task-space force control," *IEEE Trans. Robotics*, vol. 35, no. 1, pp. 64–77, 2018.