SKID RAW: Skill Discovery from Raw Trajectories

Daniel Tanneberg[1] and Kai Ploeger[1] and Elmar Rueckert[2,1] and Jan Peters[1,3]

***Abstract*—Integrating robots in complex everyday environments requires a multitude of problems to be solved. One crucial feature among those is to equip robots with a mechanism for teaching them a new task in an easy and natural way. When teaching tasks that involve sequences of different skills, with varying order and number of these skills, it is desirable to only demonstrate full task executions instead of all individual skills. For this purpose, we propose a novel approach that simultaneously learns to segment trajectories into reoccurring patterns and the skills to reconstruct these patterns from unlabelled demonstrations without further supervision. Moreover, the approach learns a skill conditioning that can be used to understand possible sequences of skills, a practical mechanism to be used in, for example, human-robot-interactions for a more intelligent and adaptive robot behaviour. The Bayesian and variational inference based approach is evaluated on synthetic and real human demonstrations with varying complexities and dimensionality, showing the successful learning of segmentations and skill libraries from unlabelled data.**

***Index Terms*—Deep Learning Methods; Representation Learning; Learning Categories and Concepts**

I. Introduction

WHILE pattern recognition, the ability to find order and regularities in noisy observations, is a necessary skill for intelligent behaviour, learning to abstract and transfer that information is a crucial ability that goes beyond pattern recognition [1]. Such cognition abilities are also helpful to employ intelligent robots into our everyday life, by lowering the barrier to program the desired robotic behaviour. While often trained experts can program robots with complex behaviour, this process requires a lot of expert knowledge in different domains, is cost intensive and often limited to special tasks or domains. Instead, it is easier to teach the robot the desired behaviour instead of programming it and is the main motivation behind the learning from demonstrations paradigm [2]. Teaching the robot by showing the desired behaviour is not only a more natural and intuitive way of programming the robot, but it also drastically reduces the required expert knowledge of the system and time.

Manuscript received: December 2, 2020; Revised February 11, 2021; Accepted March 8, 2021.

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

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. #713010 (GOAL-Robots) and No. #640554 (SKILLS4ROBOTS), and from the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - No #430054590 (TRAIN). This research was supported by NVIDIA.

[1]Intelligent Autonomous Systems, Technische Universität Darmstadt,

[2]Institute for Robotics and Cognitive Systems, Universität zu Lübeck

[3]Robot Learning Group, Max-Planck Institute for Intelligent Systems

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

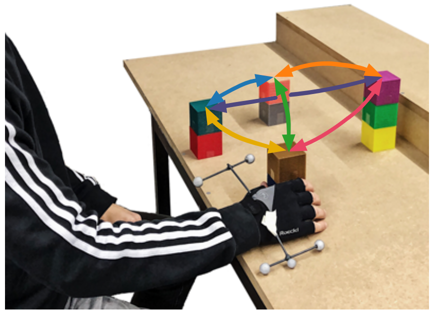


Fig. 1. Snapshot from the data collection process from the human teaching experiment. The human was instructed to move 1-3 cubes from one position to another, while the hand position was tracked. These unlabelled hand position trajectories consisting of a variable number of moved cubes are used to learn to segment these into skills and the skills simultaneously.

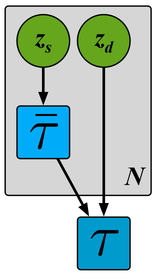
Typically demonstrations consist of single tasks, and one skill, or policy, per such task is learned from these demonstrations. For more complex tasks, that typically consist of multiple subtasks, the task is often either broken down into demonstrated subtasks or the demonstrations are segmented and labelled afterwards. Both methods again require specific knowledge, are time consuming, and scale poorly with the number of (sub)tasks and their arrangements. Subtasks can appear within a demonstration as well as across demonstrations and it can be challenging to decide where to cut or which segments can be considered the same subtask.

A more natural approach is to just demonstrate the full tasks, and let the system automatically learn to segment the full demonstration into subtasks or skills [3], [4]. Such decompositions of movements into different phases were also detected in the primary motor cortex of monkeys performing reaching tasks [5]. When equipping robots with such an automatic segmentation technique, teaching tasks which consist of a sequence of skills becomes easier for non-experts. Furthermore, by learning the segmentation, the robot can *understand* the demonstrated tasks by, for example, learning important (sub)goals or which skills are likely to follow each other. This approach can therefore also be used in human-robot interaction [6] by, for example, learning the sequence of movements of a human, and using this knowledge to predict the humans future behaviour, which allows for a more intelligent and adaptive robot behaviour.

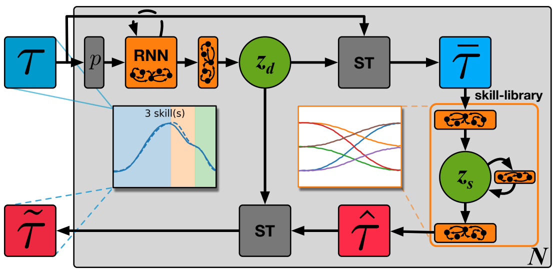
A. Contribution

Here, we propose a novel approach for learning simultaneously to segment trajectories into reoccurring patterns and the skills to reproduce them from unlabelled trajectories without supervision. Our approach is based on the variational autoencoder (VAE) [7] framework and the iterative concept

© 2021 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.



(a) Graphical model.



(b) SKID framework.

Fig. 2. (a) Graphical model representation of the relationship between the N-dimensional latent trajectory description $z = \{z_d, z_s\}$, the sub-trajectory $\bar{\tau}$ and the full trajectory τ. (b) Sketch of the SKID implementation with the learned neural networks in orange. The recurrent neural network (RNN) and the output layer learn the distribution of z_d given the, potentially p transformed, trajectory τ. Then z_d and the raw trajectory are used in a spatial transformer (ST) to extract the sub-trajectory $\bar{\tau}$. This sub-trajectory is used to learn the skill library and the approximated sub-trajectory $\hat{\tau}$ is added to the reconstructed trajectory $\tilde{\tau}$ with a spatial transformer. This process is repeated N times, iteratively segmenting the full trajectory into sub-trajectories utilizing the simultaneously learned skill library. The two inlays show examples from the 1D synthetic task, where the left inlay shows the original trajectory τ in blue and the reconstructed trajectory $\tilde{\tau}$ in red. The coloured background indicates the learned skill z_s used to approximate this segment of size z_d. The right inlay shows the learned skill library.

of AIR [8] to explain sub-parts of the given data per iteration. A similar iterative approach is investigated in the CompILE framework [9], but without an interpretable latent space, skill conditioning and length-independent skills. In contrast to related approaches [10], [11] that learn skills with VAEs from single-skill trajectories, our approach operates on trajectories with a varying and unknown number of skills per demonstration and learns to segment the trajectory and the skill library simultaneously.

Furthermore, the proposed method requires less initial knowledge like heuristic cutting points [3], expert domain knowledge [12], segmented and labelled demonstrations [13], pretraining [14], and segmentation and skill library are trained simultaneously instead of subsequently [4], [15], [16] to use the skill knowledge for segmentation. Trajectory segmentation shares similarities with switching systems and hierarchical skill discovery, investigated, for example, in (inverse) reinforcement learning settings [17], [18], hybrid systems [19] and compositional program induction [20].

We show that our approach can be used to learn skills from unlabelled demonstrations of full task executions involving a varying and unknown number of skills per demonstration, and that the learned model can also be used to understand the relation between skills and, hence, predict possible future skills for adaptive behaviour.

II. Discovering Skills from Raw Trajectories

The proposed skill discovery (SKID) approach is inspired by the AIR [8] model for images. AIR is a generative model that learns to reconstruct visual scenes by learning the properties of individual objects to render the scene. We take this approach as inspiration to construct a similar generative model for trajectories, that learns to segment raw trajectories into reoccurring patterns (subtasks) and the individual skills to reconstruct these simultaneously and without supervision.

We assume that a trajectory τ is composed of N sub-trajectories $\bar{\tau}$ that we call skills. Like AIR [8] did for images, we take a Bayesian perspective of trajectory understanding and treat it as inference in a generative model. In general, for a given trajectory τ, the model parametrized by θ is given by $p^{\tau}_{\theta}(\tau|z)p^{z}_{\theta}(z)$, where the prior $p^{z}_{\theta}(z)$ over the latents z captures the trajectory assumptions, and the likelihood $p^{\tau}_{\theta}(\tau|z)$ describes how the latent trajectory description is composed into the full trajectory. We are interested in recovering the trajectory description z, given by computing the posterior

$$p(z|\tau) = p^{\tau}_{\theta}(\tau|z)p^{z}_{\theta}(z)/p(\tau) \ . \tag{1}$$

As we assume a trajectory τ consists of N sub-trajectories or skills, the trajectory description is structured into a sequence of z^i. Each z^i is a structure that describes the properties of one skill, here, duration and skill type, i.e., $z = \{z_d, z_s\}$. The duration of a skill z_d is used for the segmentation and the skill type z_s is a discrete identifier for each skill. Figure 2(a) shows this relation between τ and z^i in a graphical model. The generative model is given as

$$p_{\theta}(\tau) = \sum_{n=1}^{N} \int p^{z}_{\theta}(z^n|z^{n-1})p^{\tau}_{\theta}(\tau|z)dz \ , \tag{2}$$

where N is given by the sequence of z^n_d summed up until a given threshold is reached, i.e., $\sum_{n=1}^{N} z^n_d \geq t_{\varepsilon}$. The number of skills ($N$) in a trajectory is dependent on the number of available skills in the library (S) and learned without additional feedback from the reconstruction loss. The condition of z^n on z^{n-1} allows to learn a conditional sequencing of the skills, i.e., learning which skills are likely to follow after each other. This conditioning is necessary to sample valid trajectories from the generative model, i.e., trajectories without big jumps between the individual skills. Additionally, by learning this conditioning, the model can be used to *understand* the presented task trajectories, e.g., like predicting which skills are likely to follow each other for adaptive robot behaviour, or be used in planning algorithms when solving new tasks with the learned skills.

A. SKID Instantiation

We use an amortized variational approximation of the true posterior $p(z|\tau)$ which learns a parametrized distribution $q_{\phi}(z|\tau)$ by minimizing the Kullback–Leibler divergence between $KL(q_{\phi}(z|\tau)||p(z|\tau))$. The inference model $q_{\phi}(z|\tau)$, parametrized by ϕ is realized as a recurrent neural network to take previous z_d's into account, i.e., allowing the model to

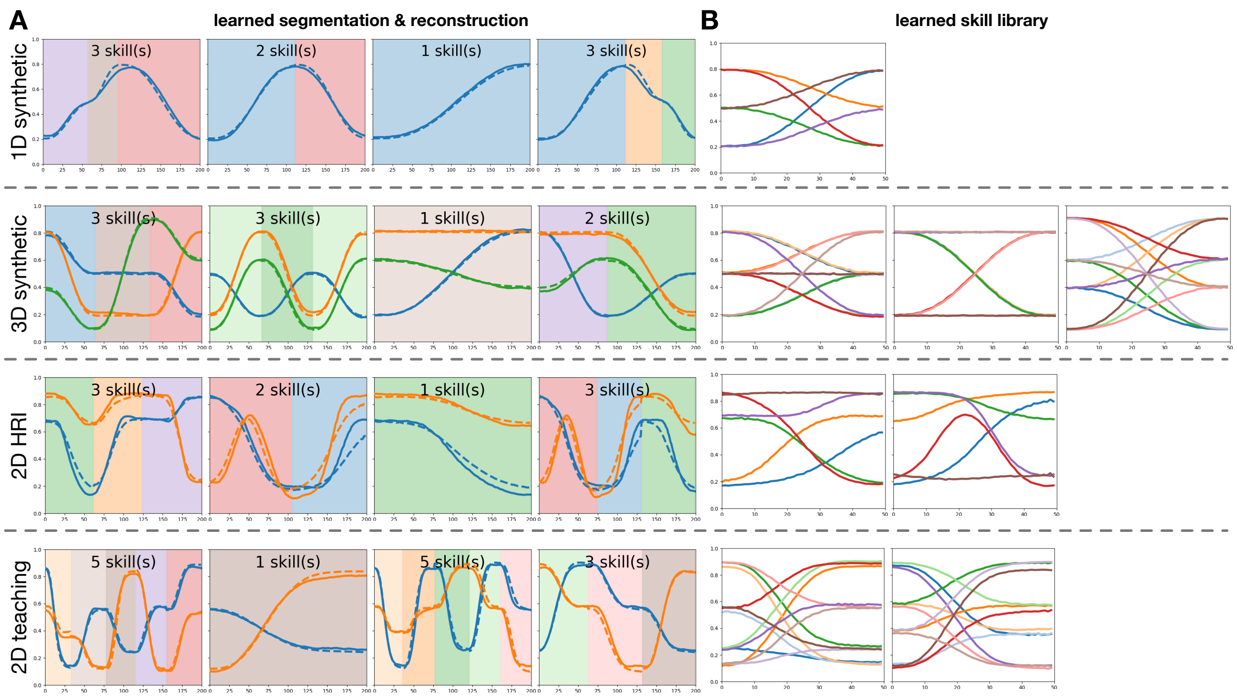


Fig. 3. Showing the learned segmentations and skill libraries by SKID for the four datasets (see Section III). **A** shows segmented and reconstructed sample trajectories. Line colours indicate trajectory dimensions, where solid lines show the original data and dashed lines show the reconstructions. The coloured area highlights the segmentation and the colour indicates which skill is used for that segment. **B** shows the learned skill library with one panel per trajectory dimension and one colour for each skill.

remember which part of the trajectory have been explained already. Before the trajectory τ is fed into the recurrent network, it is preprocessed by a function p. In our experiments, we tested with p as the identity function and p outputting the mean velocity of τ over all dimensions for each timestep. The mean velocity performed slightly better and was used in all evaluations. The skill duration z_d is modelled with a Gaussian distribution with a given prior, i.e., $z_d \sim \mathcal{N}(\mu_d, \sigma_d^2)$, fed into a `sigmoid` activation to get the duration as the fraction of the trajectory length.

To extract the part of the trajectory τ indicated by z_d in a differentiable way, a spatial transformer (ST) [21] is used. Each skill duration z_d is used starting with the remaining part of the trajectory, i.e., the part of the trajectory that has not been explained by all previous z_d. This extracted sub-trajectory $\bar{\tau}$ is then used as the input for learning the skill library, the generative model parametrized by θ. Here, we use discrete β-variational autoencoder (VAE) [7], [22], [23] with skip-connections [24] using the continuous gumbel-softmax/concrete approximation [25], [26] for the discrete skill type z_s, with a latent dimension of size S. This realization of the skill library is very general, which allows to capture any kind of trajectories and is trainable end-to-end within the SKID framework. Similar VAE based approaches but for single skill trajectories were successfully used in [10], [11]. The condition of z^n on z^{n-1} is realized as an additional neural layer, with z^{n-1} as input and the output is added to the logits of the encoder network before sampling z^n. The output of the skill library is the sub-trajectory $\hat{\tau}$ approximated with one activated skill, which is then added to the overall approximated trajectory $\tilde{\tau}$ by a spatial transformer. This iterative process is repeated N times until the sum of z_d^n reaches a given threshold t_ε, reconstructing a fraction z_d of the full trajectory τ per step. The number N is determined by the model through the unsupervised learning process, aiming at reconstructing the data under the constraints of the model. The full framework is shown in Figure 2(b).

B. SKID Learning

Learning is done by jointly optimizing the generative model and the inference network, i.e., the parameters θ and ϕ, to maximize the evidence lower bound (ELBO) given as

$$\begin{aligned}\mathcal{L}(\tau;\theta,\phi) &= \mathbb{E}_{q_\phi(z|\tau)}\left[\log\frac{p_\theta(\tau,z)}{q_\phi(z|\tau)}\right] \\ &= \mathbb{E}_{q_\phi(z|\tau)}\left[\log p_\theta(\tau|z)\right] - \mathrm{KL}(q_\phi(z||\tau)|p(z)) \ , \quad (3)\end{aligned}$$

with the parametrized likelihood $p_\theta(\tau|z)$, the parameterized inference model $q_\phi(z|\tau)$ and the latent prior $p(z)$. The first term aims at reconstructing the data while the KL-divergence forces the model to stay close to a given prior. Due to the reparametrization trick for the Gaussian distributed z_d and the gumbel-softmax/concrete distributed z_s, the ϕ parametrized inference network and the θ parametrized generative model, can be learned jointly via stochastic gradient descent.

To enforce disentangled representations, the β-VAE [22] was introduced, which weights the KL term by the hyper-parameter β. This balancing was further refined by adding capacity terms to the KL [23], [27], which can be seen as a slack variable allowing some distance in the KL, and which is increased during training. Adding these to Equation 3

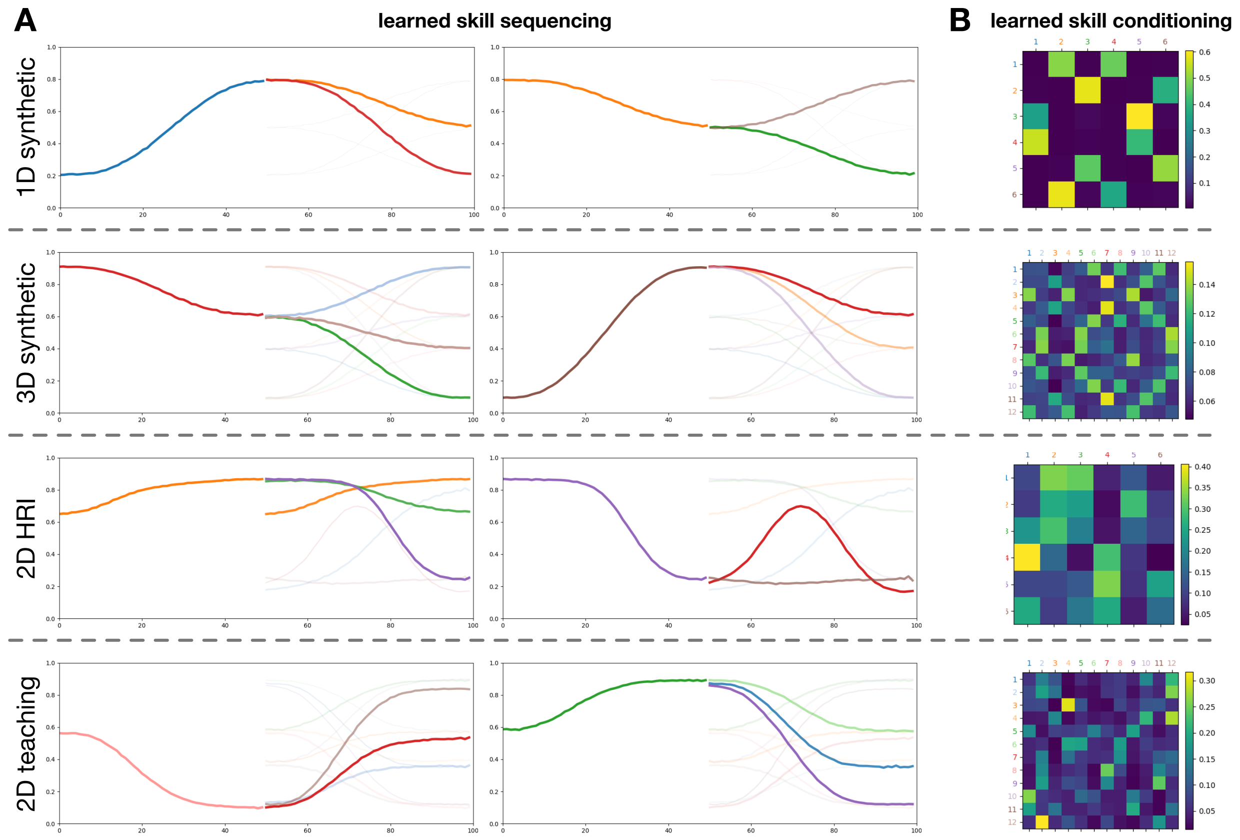


Fig. 4. Showing the learned skill conditioning by SKID for the four datasets (see Section III). **A** shows two examples of learned skills and the sequencing of the subsequent skills in one dimension, where the strength of the second plotted skill indicates how likely that skill follows the first one. **B** shows the learned conditioning for all skills as a matrix heatmap, encoded as how likely the skill in column j follows the skill in row i.

and separating the different latent variables [27], we get our learning objective as

$$\begin{aligned}\mathcal{L}(\tau;\theta,\phi) = & \mathbb{E}_{q_\phi(z|\tau)}\left[\log p_\theta(\tau|z)\right] \\ & -\gamma_d|\mathrm{KL}(q_\phi(z_d||\tau)|p(z_d)) - C_d| \\ & -\gamma_s|\mathrm{KL}(q_\phi(z_s||\tau)|p(z_s)) - C_s| \ ,\end{aligned} \tag{4}$$

with γ_d, γ_s constant scaling factors, and C_d, C_s the information capacities, i.e, the allowed slack.

To tighten the lower bound estimate and get a more complex implicit distribution, we use the importance weighted autoencoders (IWAE) [28] objective. This formulation uses K samples of the latent variables and weights the according gradients by their relative importance.

III. Experiments

In all experiments, SKID is presented unlabelled trajectories and learns to segment those into skills and the required skills simultaneously. The different datasets vary in their complexity in multiple ways: artificially created or real world recorded trajectories, the dimensionality of the trajectories, the number of subsequent skills N, or the total number of different skills. Here skills refer to reaching specific locations. With these different datasets, we test the capacity of SKID to uncover the underlying latent representation from various complex data. Additionally we test SKID's performance in a continuous learning setting, where the number of skills in the demonstrations is increased during training, and the transfer of skills learned from human demonstrations to the robot.

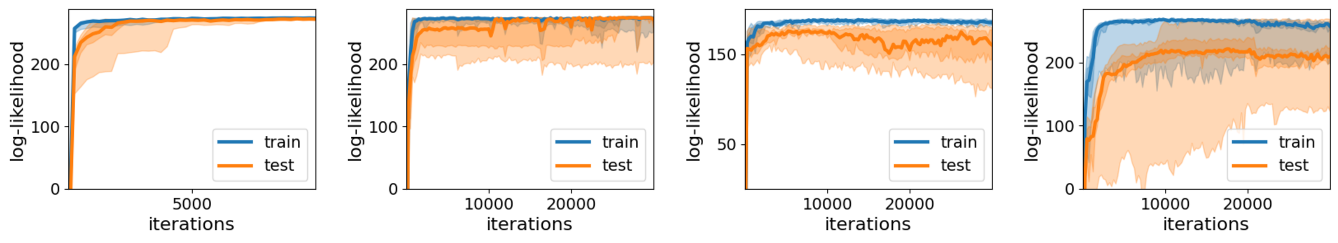
a) 1D Synthetic: This dataset consists of 1-dimensional trajectories with up to $N = 3$ skills per trajectory, and with a 6 different skills. Data was generated by sampling 10.000 trajectories for each sequence length. The individual locations per trajectory are sampled uniformly and Gaussian noise is added to each skill location. The resulting trajectory is generated by creating a minimum jerk trajectory connecting all the locations. To create a diverse dataset, we adapt a trajectory augmentation method [10] to generate the trajectory τ as:

$$\tau = \mathcal{N}(\tau_o, aB^\dagger) \ , \tag{5}$$

with the original trajectory τ_o, the constant a and $B^\dagger$ the Moore-Penrose pseudo-inverse of M, where M is set to

$$M = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & 2 & -1 & 0 & \vdots \\ 0 & -1 & 2 & \ddots & 0 \\ \vdots & 0 & \ddots & \ddots & -1 \\ 0 & \cdots & 0 & -1 & 2 \end{bmatrix} \tag{6}$$

to smoothly propagate the disturbances along the trajectory. This data augmentation method allows for small datasets and hence less physical demonstrations are required.



(a) 1D synthetic. (b) 3D synthetic. (c) 2D HRI. (d) 2D teaching.

Fig. 5. Shown is the median with the 50& and 90% quantiles of the log-likelihood of the train and test model over 10 runs. The test model uses the discrete skill type z_s and is evaluated on a held-out test set, while the train model uses the continuous approximation. The two real world datasets (bottom row) have a bigger gap between train and test due to the higher noise and variability in the data. Moreover, the two datasets (right column) with the highest number of required skills (12), tend to get stuck in local optima due to posterior collapse, i.e., no utilizing all discrete dimensions and hence, miss some skills (see also Figure 6(b)). In addition to these two challenges, the 2D teaching dataset also requires the most complex segmentation of the datasets.

b) 3D Synthetic: This dataset consists of 3-dimensional trajectories with up to $N = 3$ skills per trajectory, with a 12 different skills. Thus, the dataset is more complex in contrast to the 1-dimensional not only in the dimensionality of the trajectories, but also in the number of skills that need to be learned. Generating the dataset was done similar to the 1-dimensional dataset.

c) 2D Human-Robot-Interaction: This dataset is taken from a human-robot-interaction (HRI) scenario presented in [29], where we use the tracked hand movements of the human to learn the movements used by the human. As SKID also learns the conditioning or sequencing of the skills, this can be used to predict the human behaviour for a more intelligent robot adaptation in such collaborative scenarios. The dataset was created by taking the data form one person (10 task executions) and using 1000 random cuts at the four known goal locations, such that each trajectory consists of 2 to 5 of those locations, and use the described trajectory augmentation method to create 30.000 trajectories in total. The augmented dataset consists of two-dimensional x, y trajectories with up to $N = 4$ subsequent skills, and ideally 5 different skills to learn. Due to the heuristic cuts and human variation, the actual dataset includes samples with $N > 4$, and has the biggest variability within skills and the biggest noise across the datasets.

d) 2D Teaching: For this dataset, we recorded the wrist position of a human during an object manipulation task, shown in Figure 1. The human was instructed to move between 1 and 3 boxes per demonstration between the four locations. SKID learns the required skills for such manipulation tasks, than can be used for planning by sequencing the discovered skills, and to directly teach the skills to the robot as task space trajectories. The learned conditioning can be used, for example, to reduce the search complexity of planning algorithms. In total we recorded 223 demonstrations and used the trajectory augmentation method described in the synthetic datasets to create a dataset of 5000 trajectories for each number of moved boxes. Thus, the resulting dataset consists of two-dimensional x, y trajectories with up to $N = 5$ subsequent skills and 12 different skills to learn.

A. Experimental Setup

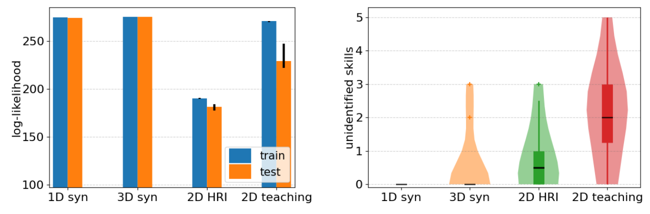
Learning is done by optimizing Equation 5 with the Adam optimizer [30] with learning rate α, weight decay λ [31], with batch prior regularization [32] for the discrete skill type, and the loss is variance normalized. The capacities C_d, C_s are linearly increased during training from (start,end,iterations). Parameters were optimized with grid search for the 1D synthetic data set, and modified for the other datasets with smaller grid searches. The mini-batch size is set to 64 and for the IWAE importance sampling $k = 20$ samples are used. The recurrent neural network (RNN) that learns z_d is a vanilla recurrent network, the threshold is set to $t_\varepsilon = 0.85$ and maximum iterations is set to $N_{\max}$. The skill library VAE encoder and decoder consist of two hidden layers with $[30, 15]$ neurons with `elu` activations, latent layer of size S (skill library size), and skip connections within the decoder. For the gumbel-softmax encoded z_s the temperature ω is decreased from (start,end,iterations) with a cosine schedule, using the continuous approximation for training but hard one-hot vectors for evaluation. Trajectories are normalized to 200 steps and the sub trajectories extracted by the spatial transformer for the skill library consist of 50 steps. The remaining parameters for all datasets are given in Table I.

B. Results

The goal of SKID is to simultaneously learn to segment trajectories into a varying number of sub-trajectories (skills)

TABLE I
HYPERPARAMETERS USED IN THE EXPERIMENTS.

	1D syn.	**3D syn.**	**2D HRI**	**2D teaching**
RNN size	64	64	64	32
linear size	16	16	16	16
$N_{\max}$	3	3	4	5
α	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$5e^{-4}$
λ	$5e^{-2}$	$5e^{-1}$	$5e^{-3}$	$5e^{-1}$
C_d	$(0, 1, 30k)$	$(0, 2, 30k)$	$(0, 2, 30k)$	$(0, 2, 30k)$
C_s	$(0, 1, 30k)$	$(0, 1, 30k)$	$(0, 1, 30k)$	$(0, 1, 30k)$
γ_d	5	3	10	3
γ_s	5	3	10	3
ω	$(1, 0.2, 30k)$	$(1, 0.2, 30k)$	$(1, 0.2, 30k)$	$(2, 0.5, 30k)$
μ_d	0	0	0.5	-0.5
σ_d	1	1	1	1
VAE σ	0.02	0.02	0.02	0.02
S	6	12	6	12
p_θ σ	0.1	0.1	0.15	0.1



(a) Maximum log-likelihood. (b) Unidentified skills.

Fig. 6. (a) Shown is the median and interquartile range of the maximum log-likelihood of the train and test model over 10 runs for each dataset. The test model uses discrete one-hot encodings for z_s and is evaluated on a held-out test set. (b) Shown are the number of missed skills, i.e., skills that were not recognized and learned, for each dataset over 10 runs each. Black horizontal line indicates the mean, the box the interquartile range with whiskers, crosses show outliers, and the shaded area represents the data density. Identifying skills is harder in the real world datasets due to their higher noise and variability. In addition, the 2D teaching dataset also has the most complex segmentation and like the 3D synthetic dataset, the most required skills (12).

and these skills from unlabelled trajectories. Additionally, as part of the skill library, a skill conditioning is learned, that encodes how likely a certain skill follows after another skill. In Figures 3 & 4 qualitative results on the different datasets are summarized, showing the evaluation of the test model on held-out data and using hard one-hot samples of z_s. Evaluations over 10 runs per dataset are shown in Figure 5 & 6, where the log-likelihood and number of unidentified skills are shown.

1) Segmentation & Skill Library: The segmentation and skill library learning results for all four datasets are shown in Figure 3, where each row corresponds to one dataset. The plot shows one exemplary run for each dataset. In the left part of the figure (A), different random trajectories are shown along with their reconstruction and segmentation. Line colours indicate the different dimensions of the trajectories, where solid lines show the input trajectory τ and dashed lines show the reconstructed trajectory $\tilde{\tau}$. Shaded areas indicate the skill used for that segment, where the according learned skills are shown in the right part of the figure (B), with one panel per trajectory dimensions and one coloured line per skill.

For all datasets SKID is able to learn to segment given trajectories into skills and the required skill library simultaneously without supervision. Learning the underlying skills and their segmentation from full plan executions without additional knowledge and feedback, enables users to teach a robot sequential tasks in a natural way by just executing the full task, without having to specify and demonstrate the individual skills. This learned knowledge can then be used by the robot to, for example, solve new task instantiations by planning with the learned skills.

2) Skill Conditioning: In addition to learning the segmentation and skill library, SKID also learns a skill conditioning. This learned conditioning and sequencing is shown for all datasets in Figure 4, with each dataset in one row. In the left part (A), two examples are shown in the two panels. In each panel, first one learned skill is plotted followed by all learned skills, but with the line intensity scaled by how likely this skill follows the first one. This sequencing shows that SKID is able to learn which skills are likely to follow each other, and that, for example, jumps between subsequent skills are unlikely. The right part of the figure (B) shows this learned skill conditioning for the whole skill library with a matrix heatmap, where the values indicate how likely the skill in column j follows the skill in row i.

SKID is able to learn this skill conditioning, i.e., task structures, for all datasets. This information can additionally be used, for example, to support the planning for new tasks with the learned skills by reducing the planning space. Another possibility for human-robot-interaction scenarios, the robot can learn to *understand* the movements of the human, and use the learned structure to predict the future behaviour of the human in order to adapt its own behaviour.

3) Continuous Learning: In many settings not all required skills are known when starting to teach a new task. Hence, the number of skills shown in the demonstrations may increase over time and the train data distribution changes. We tested SKID's ability to learn continuously in such a setting, when the number of skills in the demonstrations is increased during training. The results are shown in Figure 7, using the 1D synthetic dataset and increasing the number of skills in the demonstrations every 3000 iterations from 2 to 4 to 6. The model successfully learns all skills in the corresponding phase and is able to adapt the skill library when new skills are presented. Note, the same parameters and settings as in the previous experiments were used, except the temperature decay is reset when the data distribution changes. By changing the hyperparameters for such settings and incorporating additional mechanisms from lifelong-learning research, the performance and flexibility of SKID in continuous learning scenarios may even be increased.

4) Robot Skill Execution: To test the transfer of the learned skill library, we used the skills learned from the human 2D teaching demonstrations to rearrange objects with a real KUKA robot with a SAKE gripper. Snapshots of the robot execution of a sequence of learned skills are shown in Figure 8. The learned skills – as shown in the skill library in Figure 3B last row – represent the xy-transition trajectories between the four stacking locations. Each skill duration is set to 5 seconds for execution, height adjustment and gripper control is automated. The skills learned autonomously from unlabelled raw human trajectories can be successfully used on the real robot.

C. Limitations

While SKID is able to learn the segmentation, conditioning and skill library for different datasets with varying complexities, the stochastic variational inference setting is challenging. Not all runs achieve perfect results (see Figures 5 & 6), where the major problem that occurs is that the discrete VAE used for the skill library sometimes *misses* one or a few skills (see Figure 6(b)), even with good or perfect segmentation. In other cases the unidentified skills hinder the learning of perfect segmentation (see Figure 6(a)). Due to the continuous approximation during training, the training model can mix multiple skills to achieve good performance, the test model with hard one-hot vectors for the skills type z_s can only use one skill, and thus, this results in a sub-perfect performance. While using lower temperatures for the gumbel-softmax during

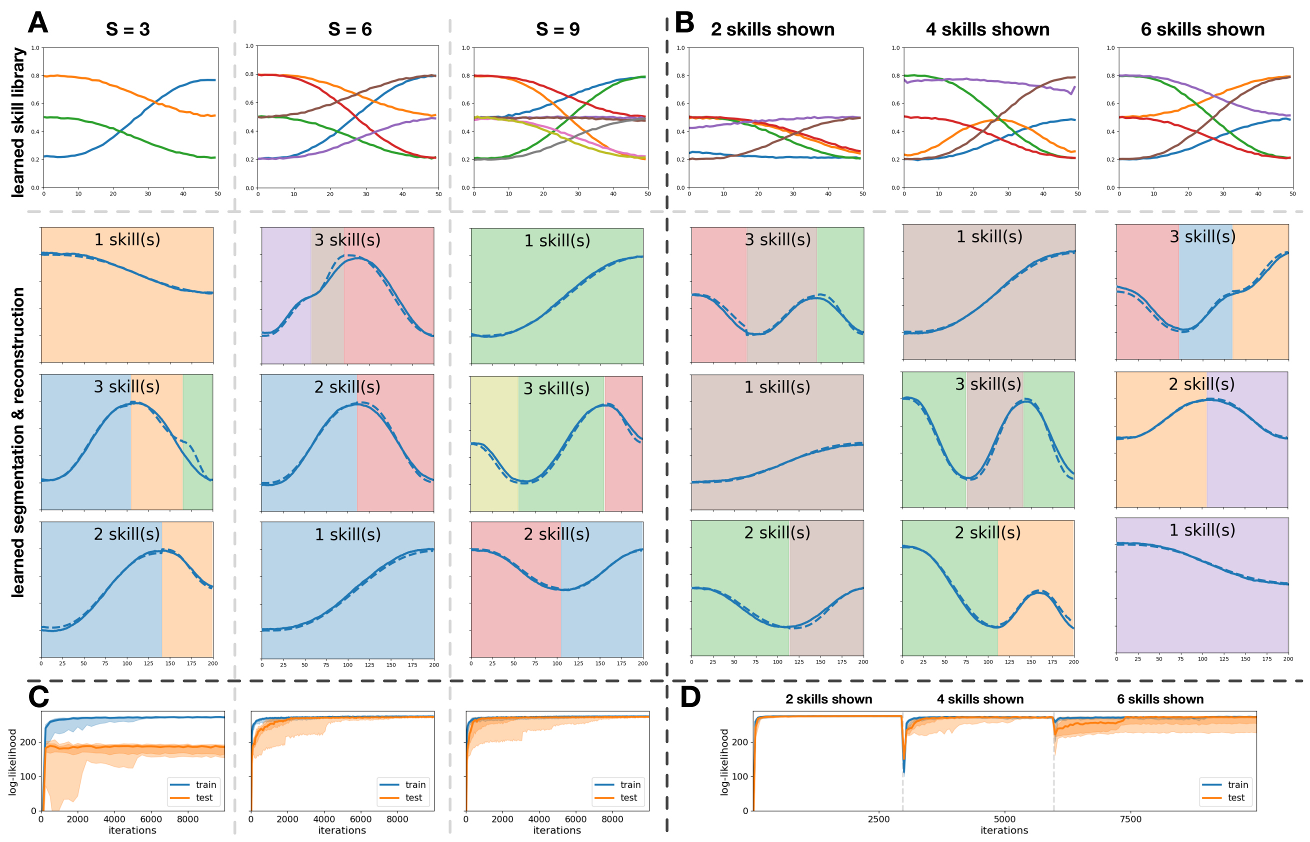


Fig. 7. Showing results on the 1D synthetic dataset with different latent dimension sizes S, i.e., skill library size, (**A**) and when increasing the shown skills in the demonstrations during training (**B**). Each column shows one setting for S or the number of currently shown skills, top row shows the learned skill library, and rows below show sampled demonstrations with their reconstruction (dashed) and segmentation (colored background indicate the corresponding used skill from the library above). **C-D** shows the median with the 50& and 90% quantiles of the log-likelihood of the train and test model over 15 runs. The test model uses the discrete skill type z_s and is evaluated on a held-out test set, while the train model uses the continuous approximation. Dashed lines in **D** indicate when more skills were added to the demonstrations.

training can help with this issue, the lower temperatures create higher variance in the gradients.

As the number of skills has to be specified for the skill library, another possibility is to use more skills than required (higher S), such that unused latent dimensions matter less. This strategy was successful in some experiments (especially in the 2D HRI setting and see Figure 7 for an evaluation of different settings of S), but with more potential skills, the segmentation may become worse, as the model tends to learn too complex skills, i.e., combining multiple skills into one, as the higher number of available skills reduces the necessity of finding the simplest individual skills, or learning *uninformative* skills used as connection skills, e.g., holding a position (see Figure 7), and hence may lead to over-segmentation. The size of the skill library influences the quality of the segmentation.

While the choice of using a VAE for the skill library is very general and allows straightforward learning of any trajectories, it has the disadvantage of a fixed latent space, i.e., number of (maximum) skills, and less flexible skills. The SKID framework is not bound to this choice, and more complex realizations like goal-conditioned VAE's [10] or a set of probabilistic movement primitives [33] can be incorporated if more flexible skills are required.

IV. Conclusion

We proposed a novel Bayesian approach to simultaneously learn trajectory segmentation and skills from unlabelled raw trajectories. The SKID framework builds on a hierarchical VAE structure and learns simultaneously to segment trajectories into reoccurring patterns, the skills to reproduce them, and the temporal relation between these skills. These features were successfully shown on multiple datasets with varying complexities, including two datasets with tracked human motions, and transferred to a robotic setup.

Such automatic skill discovery can be used as a natural interface for teaching a robot complex tasks consisting of sequences of multiple skills. In addition with the learned skill conditioning, the framework can also be used to analyse and predict the behaviour of a human, or another robot, for a more intelligent adaptive behaviour of the agent.

The next challenge is to increase the robustness of the learning, especially in settings with a high number of skills, continuous learning settings, and noisy real world data. Moreover, the framework is not limited to robotic or human movement trajectories, but can be applied to any kind of trajectories, that consist of reoccurring patterns and opens interesting future research. Another promising direction could tackle the limitation of the offline setting, i.e., full trajectories

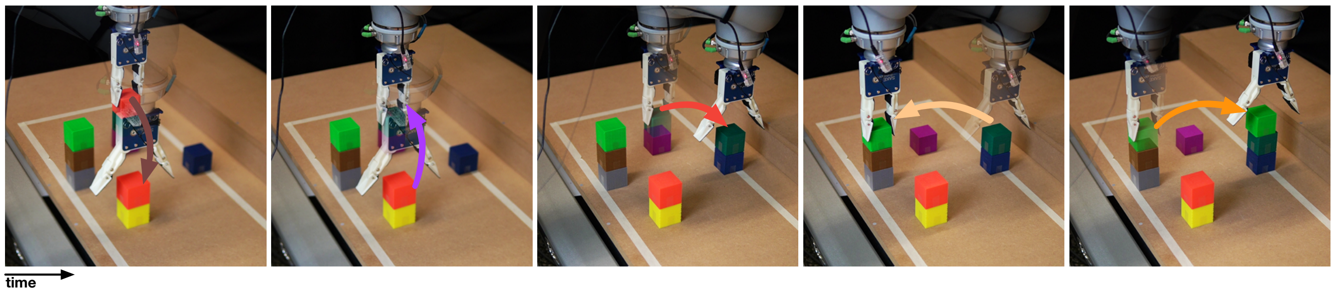


Fig. 8. Snapshots of the robot execution of a sequence of skills learned from human demonstrations (2D teaching) using SKID. Coloured arrows indicate the used skill from the learned library shown in Figure 3B. The learned skills represent the xy-transition trajectories between the four stacking positions.

are used for segmentation and learning, by integrating ideas from online change point detection approaches [34], [35] into the SKID framework and feeding the trajectories step by step to allow the processing of longer sequences.

References

[1] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman, "Building machines that learn and think like people," *Behavioral and Brain Sciences*, vol. 40, 2017.

[2] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.

[3] R. Lioutikov, G. Neumann, G. Maeda, and J. Peters, "Learning movement primitive libraries through probabilistic segmentation," *The International Journal of Robotics Research*, vol. 36, no. 8, pp. 879–894, 2017.

[4] S. Niekum, S. Osentoski, G. Konidaris, and A. G. Barto, "Learning and generalization of complex tasks from unstructured demonstrations," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5239–5246, IEEE, 2012.

[5] N. Kadmon Harpaz, D. Ungarish, N. G. Hatsopoulos, and T. Flash, "Movement decomposition in the primary motor cortex," *Cerebral Cortex*, vol. 29, no. 4, pp. 1619–1633, 2019.

[6] M. A. Goodrich and A. C. Schultz, *Human-robot interaction: a survey*. Now Publishers Inc, 2008.

[7] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *International Conference on Learning Representations*, 2014.

[8] S. A. Eslami, N. Heess, T. Weber, Y. Tassa, D. Szepesvari, G. E. Hinton, *et al.*, "Attend, infer, repeat: Fast scene understanding with generative models," in *Advances in Neural Information Processing Systems*, pp. 3225–3233, 2016.

[9] T. Kipf, Y. Li, H. Dai, V. Zambaldi, A. Sanchez-Gonzalez, E. Grefenstette, P. Kohli, and P. Battaglia, "Compile: Compositional imitation learning and execution," in *International Conference on Machine Learning*, pp. 3418–3428, PMLR, 2019.

[10] T. Osa and S. Ikemoto, "Goal-conditioned variational autoencoder trajectory primitives with continuous and discrete latent codes," *SN Computer Science*, vol. 1, no. 5, pp. 1–10, 2020.

[11] M. Noseworthy, R. Paul, S. Roy, D. Park, and N. Roy, "Task-conditioned variational autoencoders for learning movement primitives," in *Conference on Robot Learning*, pp. 933–944, 2020.

[12] F. Steinmetz, V. Nitsch, and F. Stulp, "Intuitive task-level programming by demonstration through semantic skill recognition," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3742–3749, 2019.

[13] S. Manschitz, J. Kober, M. Gienger, and J. Peters, "Learning movement primitive attractor goals and sequential skills from kinesthetic demonstrations," *Robotics and Autonomous Systems*, vol. 74, pp. 97–107, 2015.

[14] T. Shankar and A. Gupta, "Learning robot skills with temporal variational inference," in *International Conference on Machine Learning*, 2020.

[15] Z. Su, O. Kroemer, G. E. Loeb, G. S. Sukhatme, and S. Schaal, "Learning manipulation graphs from demonstrations using multimodal sensory signals," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2758–2765, IEEE, 2018.

[16] H. Zhang, X. Han, W. Zhang, and W. Zhou, "Complex sequential tasks learning with bayesian inference and gaussian mixture model," in *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 1927–1934, IEEE, 2018.

[17] P. Ranchod, B. Rosman, and G. Konidaris, "Nonparametric bayesian reward segmentation for skill discovery using inverse reinforcement learning," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 471–477, IEEE, 2015.

[18] O. Kroemer, C. Daniel, G. Neumann, H. Van Hoof, and J. Peters, "Towards learning hierarchical skills for multi-phase manipulation tasks," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1503–1510, IEEE, 2015.

[19] M. Burke, Y. Hristov, and S. Ramamoorthy, "Hybrid system identification using switching density networks," in *Conference on Robot Learning*, pp. 172–181, PMLR, 2020.

[20] M. Burke, S. Penkov, and S. Ramamoorthy, "From explanation to synthesis: Compositional program induction for learning from demonstration," in *Robotics: Science and Systems*, 2019.

[21] M. Jaderberg, K. Simonyan, A. Zisserman, *et al.*, "Spatial transformer networks," in *Advances in neural information processing systems*, pp. 2017–2025, 2015.

[22] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," in *International Conference on Learning Representations*, 2017.

[23] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner, "Understanding disentangling in beta-vae," *arXiv preprint arXiv:1804.03599*, 2018.

[24] A. B. Dieng, Y. Kim, A. M. Rush, and D. M. Blei, "Avoiding latent variable collapse with generative skip models," in *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 2397–2405, 2019.

[25] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," in *International Conference on Learning Representations*, 2017.

[26] C. J. Maddison, A. Mnih, and Y. W. Teh, "The concrete distribution: A continuous relaxation of discrete random variables," in *International Conference on Learning Representations*, 2017.

[27] E. Dupont, "Learning disentangled joint continuous and discrete representations," in *Advances in Neural Information Processing Systems*, pp. 710–720, 2018.

[28] Y. Burda, R. Grosse, and R. Salakhutdinov, "Importance weighted autoencoders," in *International Conference on Learning Representations*, 2016.

[29] D. Koert, J. Pajarinen, A. Schotschneider, S. Trick, C. Rothkopf, and J. Peters, "Learning intention aware online adaptation of movement primitives," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3719–3726, 2019.

[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations*, 2015.

[31] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *International Conference on Learning Representations*, 2019.

[32] T. Zhao, K. Lee, and M. Eskenazi, "Unsupervised discrete sentence representation learning for interpretable neural dialog generation," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1098–1107, 2018.

[33] A. Paraschos, C. Daniel, J. Peters, and G. Neumann, "Using probabilistic movement primitives in robotics," *Autonomous Robots*, vol. 42, no. 3, pp. 529–551, 2018.

[34] R. P. Adams and D. J. MacKay, "Bayesian online changepoint detection," *arXiv preprint arXiv:0710.3742*, 2007.

[35] D. Agudelo-España, S. Gomez-Gonzalez, S. Bauer, B. Schölkopf, and J. Peters, "Bayesian online prediction of change points," in *Conference on Uncertainty in Artificial Intelligence*, pp. 320–329, PMLR, 2020.