KalMamba: Towards Efficient Probabilistic State Space Models for RL under Uncertainty

Philipp Becker* **Niklas Freymuth** **Gerhard Neumann**
Karlsruhe Institute of Technology

Abstract

Probabilistic State Space Models (SSMs) are essential for Reinforcement Learning (RL) from high-dimensional, partial information as they provide concise representations for control. Yet, they lack the computational efficiency of their recent deterministic counterparts. We propose *KalMamba*, an efficient architecture to learn representations for RL that combines the strengths of probabilistic SSMs with the scalability of deterministic SSMs. *KalMamba* leverages *Mamba* to learn the dynamics parameters of a linear Gaussian SSM in a latent space. Inference in this latent space amounts to standard Kalman filtering and smoothing. We realize these operations using parallel associative scanning, similar to *Mamba*, to obtain a principled, highly efficient, and scalable probabilistic SSM. Our experiments show that *KalMamba* competes with state-of-the-art SSM approaches in RL while significantly improving computational efficiency, especially on longer interaction sequences.

1 Introduction

Deep probabilistic State Space Models (SSMs) are integral in reinforcement learning with uncertain, complex observations (Hafner et al., 2023). In contrast, deterministic SSMs efficiently parallelize and show promise in sequence modeling (Smith et al., 2022; Gu & Dao, 2023). Yet, blending both models' advantages remains a key challenge. We propose an efficient architecture that equips probabilistic SSMs with the efficiency of recent deterministic SSMs. Our approach, *KalMamba*, uses (extended) Kalman filtering and smoothing to infer belief states over a linear Gaussian SSM in a latent space that uses a dynamics model based on *Mamba* (Gu & Dao, 2023). Figure 1 provides a schematic overview. *Mamba* is efficient for long sequences as it uses parallel associative scans, which allow parallelizing associative operators on highly parallel hardware accelerators such as GPUs (Sengupta et al., 2007). Similarly, we build efficient parallel scans for filtering and smoothing (Särkkä & García-Fernández, 2020). With both *Mamba* and the Kalman Smoother being parallelizable, *KalMamba* achieves time-parallel computation of belief states required for model learning and control. Thus, unlike previous approaches for efficient SSM-based RL (Samsami et al., 2024), which rely on simplified inference assumptions, *KalMamba* enables end-to-end model training under high levels of uncertainty using a smoothing inference and tight variational lower bound (Becker & Neumann, 2022). While using smoothed beliefs for model learning, our architecture ensures a tight coupling between filtered and smoothed belief states. This inductive bias ensures the filtered beliefs are meaningful, allowing their use for policy learning and execution where future observations are unavailable.

We evaluate *KalMamba* on several tasks from the DeepMind Control (DMC) Suite (Tassa et al., 2018), training a *Soft Actor-Critic* (Haarnoja et al., 2018) on beliefs inferred from both images and states. We compare against *Recurrent State Space Models* (Hafner et al., 2019) and the *Variational Recurrent Kalman Network* (Becker & Neumann, 2022) and provide an overview of these and other related works in Appendix A. Our preliminary experiments show that *KalMamba* is competitive to these state-of-the-art SSMs while being significantly faster to train and scaling gracefully to long sequences. These results indicate *KalMamba*'s potential for foundation models that require forming accurate belief states over long sequences under uncertainty.

*Correspondence to philipp.becker@kit.edu.

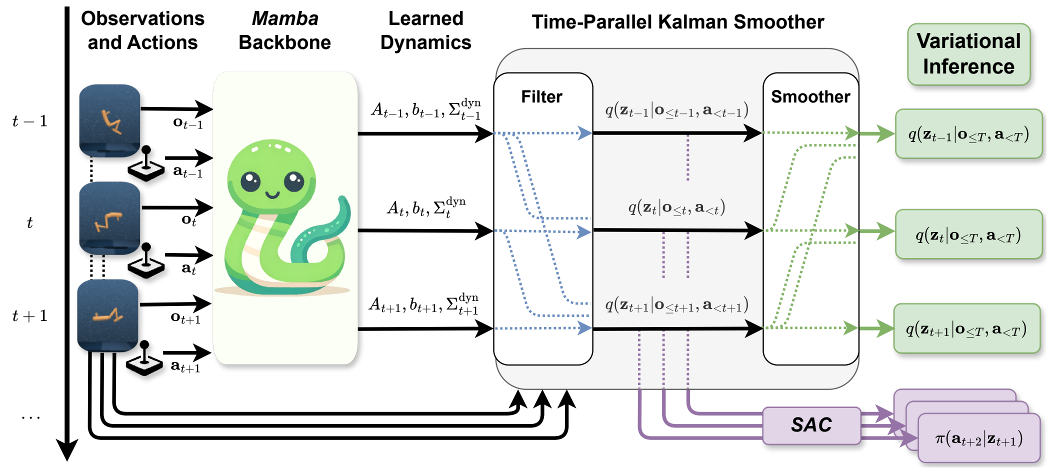

Figure 1: Overview of *KalMamba*. The observation-action sequences are first fed through a dynamics backbone built on *Mamba* to learn a linear dynamics model for each step. *KalMamba* then uses time-parallel Kalman filtering to infer filtered beliefs $q(\mathbf{z}_t|\mathbf{o}_{\leq t}, \mathbf{a}_{\leq t-1})$ which can be used for control with a *Soft Actor Critic (SAC)*. For model training, *KalMamba* employs an additional time-parallel Kalman smoothing step to obtain smoothed beliefs $q(\mathbf{z}_t|\mathbf{o}_{\leq T}, \mathbf{a}_{\leq T})$. These beliefs allow training a model that excels in modeling uncertainties due to a tight variational lower bound.

2 *KalMamba*

SSMs (Murphy, 2012) generally assume observations $\mathbf{o}_{\leq T} = \{\mathbf{o}_t\}_{t=0 \cdots T}$ which are generated by latent states $\mathbf{z}_{\leq T} = \{\mathbf{z}_t\}_{t=0 \cdots T}$, given actions $\mathbf{a}_{\leq T} = \{\mathbf{a}_t\}_{t=0 \cdots T}$. through generative models $p(\mathbf{o}_t|\mathbf{z}_t)$, and $p(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{a}_{t-1})$. To learn such models, we need to infer latent belief states given observations and actions. We differentiate between the filtered belief $\mathbf{q}(\mathbf{z}_t|\mathbf{o}_{\leq t}, \mathbf{a}_{\leq t-1})$ and the smoothed belief $\mathbf{q}(\mathbf{z}_t|\mathbf{o}_{\leq T}, \mathbf{a}_{\leq T})$. Computing these beliefs is usually intractable, but an autoencoding variational Bayes approach allows joint training of the generative and an approximate inference model using a lower bound objective (Kingma & Welling, 2013). Intuitively, *KalMamba* embeds a linear Gaussian SSM into a latent space and learns its dynamics model's parameters using a backbone consisting of several mamba layers. It employs a time-parallel Kalman smoother in this space to infer latent beliefs for training and acting, which is parallelized with parallel scans. *KalMamba* employs a tight variational lower bound that allows appropriate modeling of uncertainties in noisy, partial-observable systems. Appendix B provides additional details and compares *KalMamba* to existing SSMs.

The *KalMamba* Model. To connect the original, high-dimensional observations $\mathbf{o}_t$ to the latent space for inference, we introduce an intermediate auxiliary observation $\mathbf{w}_t$, which is connected to the latent state by an observation model $q(\mathbf{w}_t|\mathbf{z}_t) = \mathcal{N}\left(\mathbf{w}_t|\mathbf{z}_t, \mathbf{\Sigma}_t^{\mathbf{w}}\right)$ (Becker et al., 2019). Here, we assume $\mathbf{w}_t$ to be observable and extract it, together with the diagonal observation covariance $\mathbf{\Sigma}_t^{\mathbf{w}}$ from the observation using an encoder network; $(\mathbf{w}_t, \mathbf{\Sigma}_t^{\mathbf{w}}) = \phi(\mathbf{o}_t)$. This approach allows us to model the complex dependency between $\mathbf{z}_t$ and $\mathbf{o}_t$ using the encoder while having a simple observation model for inference in the latent space. We parameterize the dynamics model as

$$p(\mathbf{z}_{t+1}|\mathbf{z}_t, \mathbf{a}_t) = \mathcal{N}\left(\mathbf{z}_{t+t}|\mathbf{A}_t(\mathbf{o}_{\leq t}, \mathbf{a}_{\leq t})\mathbf{z}_t + \mathbf{b}_t(\mathbf{o}_{\leq t}, \mathbf{a}_{\leq t}), \mathbf{\Sigma}_t^{\text{dyn}}(\mathbf{o}_{\leq t}, \mathbf{a}_{\leq t})\right) \tag{1}$$

where both $\mathbf{A}_t$ and $\mathbf{\Sigma}_t^{\text{dyn}}$ are diagonal matrices. This approach effectively linearizes the dynamics parameters $\mathbf{A}_t, \mathbf{b}_t$ and $\mathbf{\Sigma}_t^{\text{dyn}}$ around all past observations and actions. Crucially, the resulting dynamics are linear in $\mathbf{z}_t$ enabling the closed-form inference of beliefs using standard Kalman filtering and smoothing. For parameterization, we use an *Mamba*-based backbone described in Appendix C and incorporate Monte-Carlo Dropout (Gal & Ghahramani, 2016) to model epistemic uncertainty effectively. The generative observation model is given by a decoder network $p(\mathbf{o}_t|\mathbf{z}_t)$. The observations are modeled as Gaussian with learned mean and fixed standard deviation. Finally, we assume an initial state distribution $p(\mathbf{z}_0)$ that is a zero mean Gaussian with a learned variance $\mathbf{\Sigma}_0$.

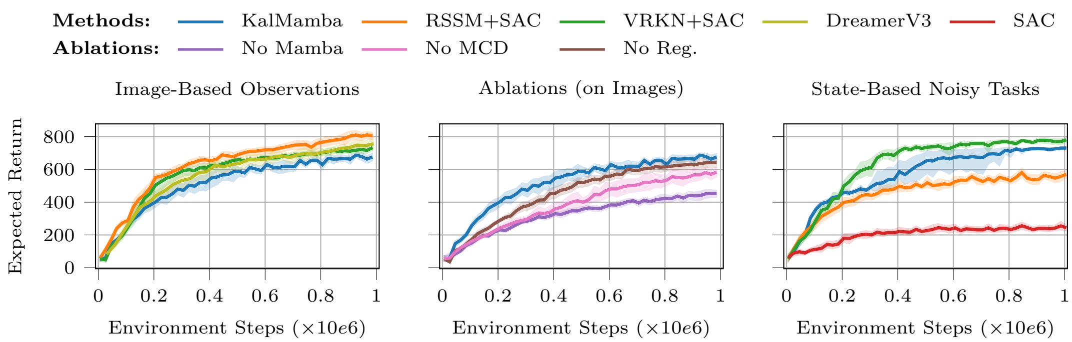

Figure 2: Aggregated expected returns for all considered environemnts. (**Left:**) On images, *KalMamba* is slightly worse but overall competitive with the different baselines. (**Middle:**) Using *Mamba* to learn the dynamics is crucial for good model performance. Monte-Carlo Dropout and the regularization loss stabilize the training process and lead to higher expected returns. (**Right:**) *KalMamba* outperforms the *RSSM* and almost matches the *VRKN*'s performance. Naive *SAC* is insufficient due to the noise added to the tasks.

Given the latent observation model $q(\mathbf{w}_t|\mathbf{z}_t)$, and the pre-computable, linear dynamics model, we can infer belief states using extended Kalman filtering and smoothing. Särkkä & García-Fernández (2020) show how to formulate such filtering and smoothing as associative operations amenable to temporal parallelization using associative scans, yielding a logarithmic time complexity, given sufficiently many parallel cores. Additionally, all involved matrices, i.e., $\mathbf{A}_t$, $\mathbf{\Sigma}_t^{\mathrm{dym}}$, $\mathbf{\Sigma}_t^{\mathrm{obs}}$, and $\mathbf{\Sigma}_0$, are diagonal which avoids costly matrix operations during Kalman filtering and smoothing.

Training the Model and Policy. After inserting the state space assumptions of our generative and inference models, the standard variational lower bound to the data marginal log-likelihood (Kingma & Welling, 2013) for a single sequence simplifies to (Becker & Neumann, 2022) $\mathcal{L}_{\mathrm{ssm}}(\mathbf{o}_{\leq T}, \mathbf{a}_{\leq T}) =$

$$\sum_{t=1}^{T}\left(\mathbb{E}_{q(\mathbf{z}_t|\mathbf{o}_{\leq T},\mathbf{a}_{\leq T})}\left[\log p(\mathbf{o}_t|\mathbf{z}_t)\right] - \mathbb{E}_{q(\mathbf{z}_{t-1}|\mathbf{o}_{\leq T},\mathbf{a}_{\leq T})}\left[\mathrm{KL}\left[q(\mathbf{z}_t|\mathbf{z}_{t-1},\mathbf{a}_{\geq t-1},\mathbf{o}_{\geq t}) \parallel p(\mathbf{z}_t|\mathbf{z}_{t-1},\mathbf{a}_{t-1})\right]\right]\right).$$

Due to the smoothing inference, this lower bound is tight and allows accurate modeling of the underlying system's uncertainties. To evaluate the lower bound we need the smoothed dynamics $q(\mathbf{z}_t|\mathbf{z}_{t-1}, \mathbf{a}_{\geq t-1}, \mathbf{o}_{\geq t})$ whose parameters we can compute given the equations provided in (Becker & Neumann, 2022). We add a reward model $p(r_t|\mathbf{z}_t)$, predicting the current reward from the latent state using a small neural network and the Mahalanobis regularization term $R(\mathbf{o}_{\leq T}, \mathbf{a}_{\leq T})$, detailed in Appendix C. Thus, the full maximization objective for a single sequence is given as

$$\mathcal{L}_{\mathrm{KalMamba}}(\mathbf{o}_{\leq T}, \mathbf{a}_{\leq T}) = \mathcal{L}_{\mathrm{ssm}}(\mathbf{o}_{\leq T}, \mathbf{a}_{\leq T}) + \mathbb{E}_{q(\mathbf{z}_t|\mathbf{o}_{\leq T},\mathbf{a}_{\leq T})}\left[\log p(r_t|\mathbf{z}_t)\right] - \alpha R(\mathbf{o}_{\leq T}, \mathbf{a}_{\leq T}).$$

We learn a *Soft Actor Critic (SAC)* (Haarnoja et al., 2018) policy on top of the *KalMamba* state space representation. Here, we use the mean of the variational filtered belief $q(\mathbf{z}_t|\mathbf{o}_{\leq t}, \mathbf{a}_{\leq t-1})$ as input for the actor and, together with the action $\mathbf{a}_t$ for the critic and stop the actor's and critic's gradients from propagating through the world model.

3 Experiments

We evaluate *KalMamba* on 4 tasks from the DeepMind Control (DMC) Suite, namely `cartpule_swingup`, `quardruped_walk`, `walker_walk`, and `walker_run`. We train each task for 1 million environment steps with sequences of length 32 and report the expected return using the mean and 95% stratified bootstrapped confidence intervals (Agarwal et al., 2021) for 4 seeds per environment. We compare against *Recurrent State Space Models (RSSMs)* and the *Variational Recurrent Kalman Network (VRKN)* on images and low-dimensional state representations with noise, as explained in Appendix D.2. To isolate the effect of the SSMs' representations, we combine both

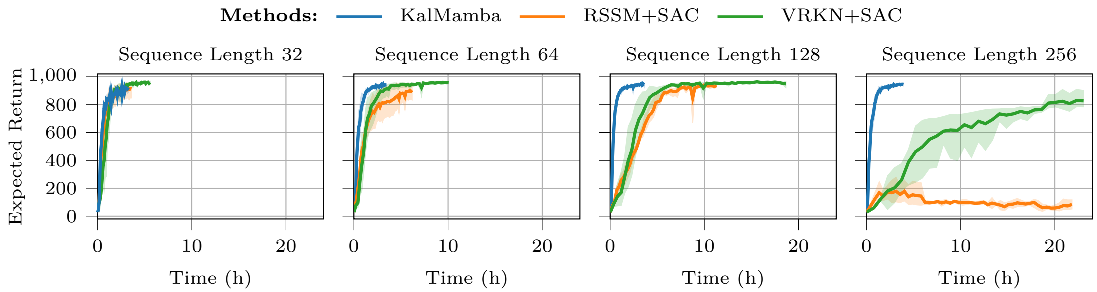

Figure 3: Wall-clock time evaluations on the state-based noisy `walker-walk` for *KalMamba*, the *RSSM*, and the *VRKN* for different training context lengths for 1 million environment steps or up to 24 hours. This time limitation only affected the *VRKN* training for 256 steps, which reached 650 thousand steps after 24 hours. While all methods work well for short sequences of length 32 (**Left**), the efficient parallelization of *KalMamba* allows it to scale gracefully to and even improve performance for longer sequences of up to 256 steps, where the other methods fail (**Right**).

with *SAC* (Haarnoja et al., 2018) as the RL algorithm, instead of using latent imagination (Hafner et al., 2020). We include *SAC* in our low-dimensional experiments, and add *DreamerV3* (Hafner et al., 2023) results for image-based observations for reference. Appendix D lists all hyperparameters.

Figure 2 shows the aggregated expected returns across setups, while Appendix E provides per-task results for all experiments. On images, *KalMamba* is slightly worse, but overall competitive to the two baseline SSMs and *DreamerV3*, while being parallelizable and thus much more efficient to train. Naively using *SAC* fails when trained on the noisy low-dimensional states. While the *RSSM* manages to improve performance it is still significantly outperformed by *VRKN* and *KalMamba*, which both use the robust smoothing inference scheme. *KalMamba* needs slightly longer to converge, but almost matches the *VRKN*'s performance while being significantly faster to run. Our ablations show that omitting the *Mamba* backbone and instead linearizing the dynamics around the current actions and observations is insufficient. Further, we find that both the Mahalanobis regularization and Monte-Carlo Dropout greatly boost performance.

We compare the runtime of the different SSMs on the state-based noisy version of `walker-walk` across varying sequence lengths in Figure 3. The models share a PyTorch implementation and differ only in the SSM. We run each experiment on a single Nvidia Tesla H100 GPU, for up to 1 million steps or 24 hours. All models work well for sequences of length 32 used for the experiments in Figure 2. Yet, only *KalMamba* scales to longer sequences, uniquely *improving* performance with sequence length while also maintaining a low training cost. These results showcase *KalMamba*'s efficient use of long-term context information through its *Mamba* backbone. We further show in Appendix E.1 that *KalMamba* scales gracefully to very long context sizes on individual SSM forward passes and training batches, whereas the baseline SSMs quickly become prohibitively expensive.

4 Conclusion

We proposed *KalMamba*, an efficient State Space Model (SSM) for Reinforcement Learning (RL) under uncertainty. It combines the uncertainty awareness of probabilistic SSMs with recent deterministic SSMs' scalability by embedding a linear Gaussian SSM into a latent space. We use *Mamba* (Gu & Dao, 2023) to learn the linearized dynamics in this latent space efficiently. Inference in this SSM amounts to standard Kalman filtering and smoothing and is amenable to full parallelization using associative scans (Särkkä & García-Fernández, 2020). Our experiments indicate that *KalMamba* can match the performance of state-of-the-art stochastic SSMs for RL under uncertainty. *KalMamba* scales gracefully to longer training sequences in terms of runtime, and improves performance with sequence length while the baseline SSMs degrade. In future work, we aim to explore *KalMamba* as a foundation model on diverse, more realistic scenarios, comparing to existing time-efficient SSMs with simplified, non-smoothing inference schemes (Samsami et al., 2024).

References

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in neural information processing systems*, 34:29304–29320, 2021.

Evan Archer, Il Memming Park, Lars Buesing, John Cunningham, and Liam Paninski. Black box variational inference for state space models. *arXiv preprint arXiv:1511.07367*, 2015.

Ershad Banijamali, Rui Shu, Hung Bui, Ali Ghodsi, et al. Robust locally-linear controllable embedding. In *International Conference on Artificial Intelligence and Statistics*, pp. 1751–1759. PMLR, 2018.

Philipp Becker and Gerhard Neumann. On uncertainty in deep state space models for model-based reinforcement learning. *Transactions on Machine Learning Research*, 2022.

Philipp Becker, Harit Pandya, Gregor Gebhardt, Cheng Zhao, C James Taylor, and Gerhard Neumann. Recurrent kalman networks: Factorized inference in high-dimensional deep feature spaces. In *International Conference on Machine Learning*, pp. 544–552, 2019.

Philipp Becker, Sebastian Markgraf, Fabian Otto, and Gerhard Neumann. Joint representations for reinforcement learning with multiple sensors. *arXiv preprint arXiv:2302.05342*, 2023.

Philip Becker-Ehmck, Jan Peters, and Patrick Van Der Smagt. Switching linear dynamics for variational Bayes filtering. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 553–562. PMLR, 09–15 Jun 2019.

Chang Chen, Yi-Fu Wu, Jaesik Yoon, and Sungjin Ahn. Transdreamer: Reinforcement learning with transformer world models. *arXiv preprint arXiv:2202.09481*, 2022.

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

Andreas Doerr, Christian Daniel, Martin Schiegg, Nguyen-Tuong Duy, Stefan Schaal, Marc Toussaint, and Trimpe Sebastian. Probabilistic recurrent state-space models. In *International Conference on Machine Learning*, pp. 1280–1289. PMLR, 2018.

Stefanos Eleftheriadis, Tom Nicholson, Marc Peter Deisenroth, and James Hensman. Identification of gaussian process state space models. In *NIPS*, pp. 5309–5319, 2017.

Marco Fraccaro, Simon Kamronn, Ulrich Paquet, and Ole Winther. A disentangled recognition and nonlinear dynamics model for unsupervised learning. In *Advances in Neural Information Processing Systems*, pp. 3601–3610, 2017.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Albert Gu, Karan Goel, and Christopher Re. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2021.

S Gu, Z Ghahramani, and RE Turner. Neural adaptive sequential monte carlo. *Advances in Neural Information Processing Systems*, 2015:2629–2637, 2015.

Ankit Gupta, Albert Gu, and Jonathan Berant. Diagonal state spaces are as effective as structured state spaces. *Advances in Neural Information Processing Systems*, 35:22982–22994, 2022.

Tuomas Haarnoja, Anurag Ajay, Sergey Levine, and Pieter Abbeel. Backprop kf: learning discriminative deterministic state estimators. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pp. 4383–4391, 2016.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pp. 2555–2565. PMLR, 2019.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

Danijar Hafner, Timothy P Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. In *International Conference on Learning Representations*, 2021.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Ramin Hasani, Mathias Lechner, Tsun-Hsuan Wang, Makram Chahine, Alexander Amini, and Daniela Rus. Liquid structural state-space models. In *The Eleventh International Conference on Learning Representations*, 2022.

AH Jazwinski. *Stochastic processes and filtering theory*. ACADEMIC PRESS, INC.„ 1970.

Maximilian Karl, Maximilian Soelch, Justin Bayer, and Patrick van der Smagt. Deep variational bayes filters: Unsupervised learning of state space models from raw data. *arXiv preprint arXiv:1605.06432*, 2016.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Alexej Klushyn, Richard Kurle, Maximilian Soelch, Botond Cseke, and Patrick van der Smagt. Latent matters: Learning deep state-space models. *Advances in Neural Information Processing Systems*, 34, 2021.

Rahul Krishnan, Uri Shalit, and David Sontag. Structured inference networks for nonlinear state space models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

Antonio Moretti, Zizhao Wang, Luhuan Wu, and Itsik Pe'er. Smoothing nonlinear variational objectives with sequential monte carlo, 2019.

Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

Christian Naesseth, Scott Linderman, Rajesh Ranganath, and David Blei. Variational sequential monte carlo. In *International Conference on Artificial Intelligence and Statistics*, pp. 968–977. PMLR, 2018.

Tung D Nguyen, Rui Shu, Tuan Pham, Hung Bui, and Stefano Ermon. Temporal predictive coding for model-based planning in latent space. In *International Conference on Machine Learning*, pp. 8130–8139. PMLR, 2021.

Mohammad Reza Samsami, Artem Zholus, Janarthanan Rajendran, and Sarath Chandar. Mastering memory tasks with world models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=1vDArHJ68h.

Simo Särkkä and Ángel F García-Fernández. Temporal parallelization of bayesian smoothers. *IEEE Transactions on Automatic Control*, 66(1):299–306, 2020.

Florian Schmidt and Thomas Hofmann. Deep state space models for unconditional word generation. *Advances in Neural Information Processing Systems 31*, 31:6158–6168, 2018.

Shubhabrata Sengupta, Mark Harris, Yao Zhang, and John D Owens. Scan primitives for gpu computing, 2007.

Vaisakh Shaj, Philipp Becker, Dieter Buchler, Harit Pandya, Niels van Duijkeren, C James Taylor, Marc Hanheide, and Gerhard Neumann. Action-conditional recurrent kalman networks for forward and inverse dynamics learning. *Conference on Robot Learning*, 2020.

Vaisakh Shaj, Dieter Büchler, Rohit Sonker, Philipp Becker, and Gerhard Neumann. Hidden parameter recurrent state space models for changing dynamics scenarios. In *International Conference on Learning Representations*, 2022.

Robert H Shumway and David S Stoffer. An approach to time series smoothing and forecasting using the em algorithm. *Journal of time series analysis*, 3(4):253–264, 1982.

Jimmy TH Smith, Andrew Warrington, and Scott Linderman. Simplified state space layers for sequence modeling. In *The Eleventh International Conference on Learning Representations*, 2022.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in neural information processing systems*, pp. 2746–2754, 2015.

Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. Daydreamer: World models for physical robot learning. In *6th Annual Conference on Robot Learning*, 2022.

Li Yingzhen and Stephan Mandt. Disentangled sequential autoencoder. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5670–5679. PMLR, 10–15 Jul 2018.

Linqi Zhou, Michael Poli, Winnie Xu, Stefano Massaroli, and Stefano Ermon. Deep latent state space models for time-series generation. In *International Conference on Machine Learning*, pp. 42625–42643. PMLR, 2023.

A Related Work

Deterministic State Space Models in Deep Learning. Structured deterministic State Space approaches (Gu et al., 2021; Smith et al., 2022; Gu & Dao, 2023) recently emerged as an alternative to the predominant Transformer (Vaswani et al., 2017) architecture for general sequence modeling (Gu & Dao, 2023). Their main benefit is combining compute and memory requirements that scale linearly in sequence length with efficient and parallelizable implementations. While earlier approaches, such as the *Structured State Space Sequence Model S4* (Gu et al., 2021) and others (Gupta et al., 2022; Hasani et al., 2022) used a convolutional formulation for efficiency, more recent approaches (Smith et al., 2022; Gu & Dao, 2023) use associative scans. Such associative scans allow for parallel computations over sequences if all involved operators are associative, which yields a logarithmic runtime, given enough parallel cores. However, all these models are deterministic, i.e., they do not model uncertainties or allow sampling without further modifications. As a remedy, *Latent S4 (LS4)* (Zhou et al., 2023) extends *S4* for probabilistic generative sequence modeling and forecasting. However, in LS4, the latent states are not Markovian and are thus hard to use for control. *KalMamba* exploits the fact that filtering and smoothing in linear Gaussian state space models can also be formulated as a set of associative operations, which makes it amenable to parallel scans (Särkkä & García-Fernández, 2020). To our knowledge, it is the first deep-learning model to do so. Further, it relies on *Mamba* (Gu & Dao, 2023), a state-of-the-art deterministic state space model, to precompute the dynamics models required for filtering and smoothing.

Probabilistic State Space Models for Reinforcement Learning. Probabilistic state space models are commonly and successfully used for reinforcement learning from high dimensional or multimodal observations (Nguyen et al., 2021; Wu et al., 2022; Hafner et al., 2023; Becker et al., 2023), under partial observability (Becker & Neumann, 2022), and for memory tasks (Samsami et al., 2024). Arguably, the most prominent approach is the *Recurrent State Space Model (RSSM)* (Hafner et al., 2019). After their original introduction as the basis of a standard planner, they have been improved with more involved parametric policy learning approaches (Hafner et al., 2020) and categorical latent variables for categorical domains (Hafner et al., 2021). During inference, the *RSSMs* conditions the latent state on past observations and actions, resulting in a filtering inference scheme. Here, the key architectural feature of *RSSMs* is splitting the latent state into stochastic and deterministic parts. The deterministic part is then propagated through time using a standard recurrent architecture. In its original formulation, the *RSSM* uses a Gated Recurrent Unit (GRU) (Cho et al., 2014). One line of research focuses on replacing this deterministic path with more efficient architectures with the *TransDreamer* (Chen et al., 2022) approach using a transformer (Vaswani et al., 2017) and *Recall to Image* (Samsami et al., 2024) using S4 (Gu et al., 2021). However, to fully exploit the efficiency of these backbone architectures, both need to simplify the inference assumptions and can only consider the current observation, which makes them highly susceptible to noise or missing observations. Opposed to that, the *Variational Recurrent Kalman Network (VRKN)* (Becker & Neumann, 2022) proposes using a smoothing inference scheme that conditions both past and future actions. This scheme allows the *VRKN* to work with a fully stochastic latent state and lets it excel in tasks where modeling uncertainty is crucial. The *VRKN* uses a locally linear Gaussian State Space Model in a latent space, performing closed-form Kalman Filtering and smoothing. *KalMamba* holistically combines smoothing inference in a fully probabilistic SSM with an efficient temporally parallaelized implementation, resulting in an approach that is robust to noise and efficient.

Probabilistic State Space Models in Deep Learning. Probabilistic state space models are versatile and commonly used tools in machine learning. Besides classical approaches using linear models (Shumway & Stoffer, 1982) and works using Gaussian Processes (Eleftheriadis et al., 2017; Doerr et al., 2018), most recent methods build on Neural Networks (NNs) to parameterize generative and inference models using the SSM assumptions (Archer et al., 2015; Watter et al., 2015; Gu et al., 2015; Karl et al., 2016; Fraccaro et al., 2017; Krishnan et al., 2017; Banijamali et al., 2018; Yingzhen & Mandt, 2018; Schmidt & Hofmann, 2018; Naesseth et al., 2018; Becker et al., 2019; Becker-Ehmck et al., 2019; Moretti et al., 2019; Shaj et al., 2020; Klushyn et al., 2021; Shaj et al., 2022).

Table 1: Comparing the inference models and capabilities for smoothing (Smooth) and time-parallel (Parallel) execution of recent SSMs for RL.

Method	Inference Model	Smooth	Parallel
RSSM (Hafner et al., 2019)	$q(\mathbf{z}_t \mid \mathbf{h}_t, \mathbf{o}_t)$	×	×
R2I (Samsami et al., 2024)	$q(\mathbf{z}_t \mid \mathbf{o}_t)$	×	✓
VRKN (Becker & Neumann, 2022)	$q(\mathbf{z}_t \mid \mathbf{o}_{\leq T}, \mathbf{a}_{\leq T})$	✓	×
KalMamba	$q(\mathbf{z}_t \mid \mathbf{o}_{\leq T}, \mathbf{a}_{\leq T})$	✓	✓

Out of these approaches, those that embed linear-Gaussian SSMs into latent spaces (Watter et al., 2015; Haarnoja et al., 2016; Fraccaro et al., 2017; Banijamali et al., 2018; Becker-Ehmck et al., 2019; Becker et al., 2019; Shaj et al., 2020; Klushyn et al., 2021; Shaj et al., 2022) are of particular relevance to *KalMamba*. Doing so allows for closed-form inference using (extended) Kalman Filtering and Smoothing. However, with the notable exception of the *VRKN*, these models usually cannot be used to control or even model systems of similar complexity to those controlled with *RSSM*-based approaches. Furthermore, some of them (Karl et al., 2016; Becker-Ehmck et al., 2019) do not allow smoothing, while others (Fraccaro et al., 2017; Klushyn et al., 2021) model observations in the latent space as additional random variables which complicates inference and training and prevents principled usage of the observation uncertainty for filtering. Another class of approaches (Haarnoja et al., 2016; Becker et al., 2019; Shaj et al., 2020; 2022) trains using regression and are thus not generative. Notably, none of these approaches uses a temporally parallelized formulation of the filtering and smoothing operations.*KalMamba* takes inspiration from many of these approaches and partly follows the *VRKN*'s design to enable reinforcement learning for complex systems. However, it combines those ideas with the efficiency of recent deterministic SSMs using an architecture that enables time-parallel computations.

B State Space Models for Reinforcement Learning

In Reinforcement Learning (RL) under uncertainty and partial observability, State Space Models (SSMs) generally assume sequences of observations $\mathbf{o}_{\leq T} = \{\mathbf{o}_t\}_{t=0 \cdots T}$ which are generated by a sequence of latent state variables $\mathbf{z}_{\leq T} = \{\mathbf{z}_t\}_{t=0 \cdots T}$, given a sequence of actions $\mathbf{a}_{\leq T} = \{\mathbf{a}_t\}_{t=0 \cdots T}$. The corresponding generative model factorizes according to the hidden Markov assumptions (Murphy, 2012), i.e., each observation $\mathbf{o}_t$ only depends on the current latent state $\mathbf{z}_t$ through an observation model $p(\mathbf{o}_t \mid \mathbf{z}_t)$, and each latent state $\mathbf{z}_t$ only depends on the previous state $\mathbf{z}_{t-1}$ and the action $\mathbf{a}_{t-1}$ through a dynamics model $p(\mathbf{z}_t \mid \mathbf{z}_{t-1}, \mathbf{a}_{t-1})$.

In order to learn the state space model from data and use it for downstream RL, we need to infer latent belief states given observations and actions. Depending on the information provided for inference, we differentiate between the filtered belief $\mathbf{q}(\mathbf{z}_t \mid \mathbf{o}_{\leq t}, \mathbf{a}_{\leq t-1})$ and the smoothed belief $\mathbf{q}(\mathbf{z}_t \mid \mathbf{o}_{\leq T}, \mathbf{a}_{\leq T})$. The filtered belief conditions only on past information, while the smoothed belief also depends on future information. Computing these beliefs is intractable for models of reasonable complexity. Thus, we resort to an autoencoding variational Bayes approach that allows joint training of the generative and an approximate inference model using a lower bound objective (Kingma & Welling, 2013).

The *Recurrent State Space Model (RSSM)* (Hafner et al., 2019) assumes a nonlinear dynamics model, splitting the state $\mathbf{z}_t$ into a stochastic $\mathbf{s}_t$ and a deterministic part $\mathbf{h}_t$ which evolve according to $\mathbf{h}_t = f(\mathbf{h}_{t-1}, \mathbf{a}_{t-1}, \mathbf{s}_{t-1})$ and $\mathbf{s}_t \sim p(\mathbf{s}_t \mid \mathbf{h}_t)$. Here f is implemented using a *Gated Recurrent Unit (GRU)* (Cho et al., 2014). This results in a nonlinear, autoregressive process that cannot be parallelized over time. Further, *RSSMs* assume a filtering inference model $q(\mathbf{s}_t \mid \mathbf{h}_t, \mathbf{o}_t)$, where $\mathbf{h}_t$ accumulates all information from the past. The *RSSM*'s inference scheme struggles with correctly estimating uncertainties as the resulting lower bound is not tight (Becker & Neumann, 2022). In

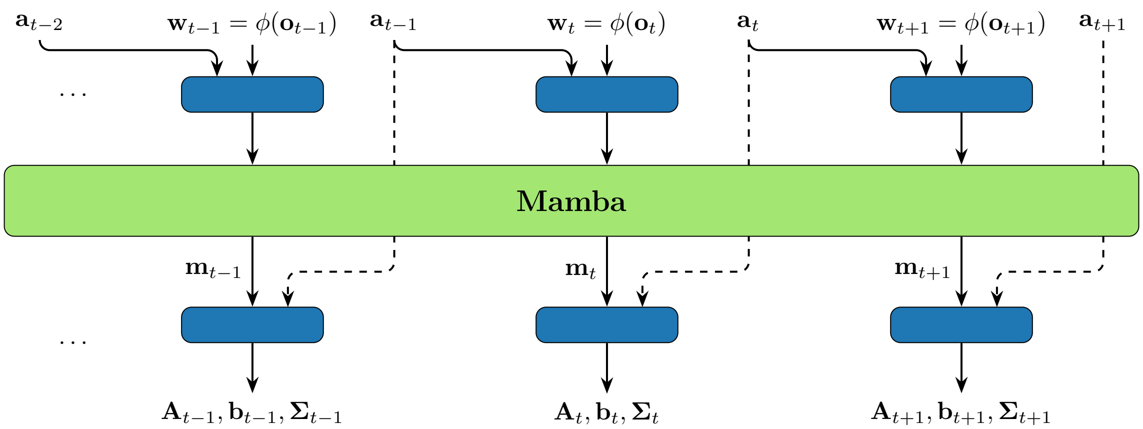

Figure 4: Schematic of the Mamba Gu & Dao (2023) based backbone to learn the system dynamics. It shares the inference model's encoder $\phi(\mathbf{o}_t)$ and intermediate representation $\mathbf{w}_t$. Each $\mathbf{w}_t$ is then concatenated to the previous action $\mathbf{a}_{t-1}$, fed through a small Neural Network (NN) and given to *Mamba* model which accumulates information over time and emits a representation $\mathbf{m}_t(\mathbf{o}_{t\leq}, \mathbf{a}_{\leq t-1})$ containing the same information as the filtered belief $q(\mathbf{z}_t|\mathbf{o}_{t\leq}, \mathbf{a}_{\leq t-1})$. We then concatenate each $\mathbf{m}_t$ with the current action $\mathbf{a}_t$ and use another small NN to compute the dynamics parameters $\mathbf{A}_t, \mathbf{b}_t$ and $\mathbf{\Sigma}_t$. This scheme allows us to use the intermediate representation $\mathbf{m}_t$ for regularization and we regularize it towards the filtered belief's mean using a Mahalanobis regularizer (c.f. Equation 2). Finally, the small NNs include Monte-Carlo Dropout Gal & Ghahramani (2016) to model epistemic uncertainty.

tasks where such uncertainties are relevant, this lack of principled uncertainty estimation causes poor performance for downstream applications.

As a remedy, the *Variational Recurrent Kalman Network (VRKN)* (Becker & Neumann, 2022) builds on a linear Gaussian SSM in a latent space which allows inferring smoothed belief states $\mathbf{q}(\mathbf{z}_t|\mathbf{o}_{\leq T}, \mathbf{a}_{\leq T})$ required for a tight bound. The *VRKN* removes the need for a deterministic path and improves performance under uncertainty. However, it linearizes the dynamics model around the mean of the filtered belief, resulting in a nonlinear autoregressive process that cannot be parallelized.

In contrast, *Recall to Image (R2I)* (Samsami et al., 2024) builds on the *RSSM* and improves computational efficiency at the cost of a more simplistic inference scheme. It uses *S4* (Gu et al., 2021) instead of a *GRU* to parameterize the deterministic path f but additionally has to remove the inference's dependency on $\mathbf{h}_t$ to allow efficient parallel computation. The resulting inference model, $q(\mathbf{z}_t|\mathbf{o}_t)$ is non-recurrent and neglects all information from other time steps. Thus, while *R2I* excels on memory tasks, it is highly susceptible to noise and partial-observability as the inference cannot account for inconsistent or missing information in $\mathbf{o}_t$.

Our approach, *KalMamba*, combines the tight variational lower bound of the *VRKN* with a parallelizable *Mamba* (Gu & Dao, 2023) backbone to learn the parameters of the dynamics. It thus omits the nonlinear autoregressive linearization process. Combined with our custom PyTorch routines for time-parallel filtering and smoothing (Särkkä & García-Fernández, 2020), this approach allows efficient training with the *VRKN*s principled, uncertainty-capturing objective.

C Mamba Backbone and Regularization

Parameterizing the dynamics model of Equation 1 naively can lead to poor representations, as information can bypass the actual SSM through the linearization backbone. To counter this, we design the backbone architecture as depicted in Figure 4. For each timestep, we concatenate $\mathbf{w}_t$ and $\mathbf{a}_{t-1}$, transform each resulting vector using a small neural network, feed it through a *Mamba* (Gu

& Dao, 2023) model and linearly project the output to a vector $\mathbf{m}_t$ of the same dimension as the latent state $\mathbf{z}_t$. Each $\mathbf{m}_t$ now accumulates the same observations and actions used to form the corresponding filtered belief $q(\mathbf{z}_t|\mathbf{o}_{\leq t}, \mathbf{a}_{\leq t-1})$. We then take $\mathbf{m}_t$ and the action $\mathbf{a}_t$ to compute the dynamics parameters using another small neural network. This bottleneck introduced by $\mathbf{m}_t$ allows us to regularize the *Mamba*-based backbone. We incentivize $\mathbf{m}_t$ to correspond to the filtered mean using a Mahalonobis distance

$$R(\mathbf{o}_{\leq T}, \mathbf{a}_{\leq T}) = \sum_{t=1}^{T} \left(\mathbf{m}_t(\mathbf{o}_{\leq t}, \mathbf{a}_{\leq t-1} - \boldsymbol{\mu}_t^+\right)^T \left(\boldsymbol{\Sigma}_t^+\right)^{-1} \left(\mathbf{m}_t(\mathbf{o}_{\leq t}, \mathbf{a}_{\leq t-1}) - \boldsymbol{\mu}_t^+\right), \tag{2}$$

$\boldsymbol{\mu}_t^+$ and $\boldsymbol{\Sigma}_t^+$ denote the mean and variance of the filtered belief $q(\mathbf{z}_t|\mathbf{o}_{\leq t}, \mathbf{a}_{\leq t-1})$. This regularization discourages the model from bypassing information over the *Mamba* backbone. This mirrors many established models such as the classical extend Kalman Filter (Jazwinski, 1970), which linearize directly around this mean, but still allows associative parallel scanning.

D Hyperparameters and Implementation Details

Table 2 lists all hyperparameters of the *KalMamba* model and Table 3 lists the hyperparameters of *Soft Actor Critic (SAC)* Haarnoja et al. (2018) used for control. For all experiments, we run 20 evaluation runs every $20,000$ steps.

Table 2: World Model Hyperparameters

Hyperparameter	Low Dimensional DMC	Image Based DMC
	World Model	
Encoder	2×256 Unit NN with ELU	ConvNet from Hafner et al. (2020) with ReLU
Decoder	2×256 Unit NN with ELU	ConvNet from Hafner et al. (2020) with ReLU
Reward Decoder	2×256 Unit NN with ELU	
Latent Space Size	230 (30 Stoch. + 200 Det. for RSSM	
	Mamba Backbone	
num blocks	2	
d_model	256	
d_state	64	
d_conv	2	
dropout probability	0.1	
activation	SiLU	
pre mamba layers	2×256 Unit NN with SiLU	
post mamba layers	VRKN Dynamics Model from Becker & Neumann (2022) with SiLU	
	Loss	
KL Balancing	0.8 for RSSM, 0.5 for VRKN, KalMamba	
Free Nats	3	
α (regularization scale)	1, KalMamba only	
	Optimizer (Adam Kingma & Ba (2015))	
Learning Rate	$3 \cdot 10^{-4}$	

D.1 Baselines.

Both *RSSM+SAC* and *VRKN+SAC* use the same hyperparameters as *KalMamba* where applicable. For all other hyperparameters, we use the defaults from Hafner et al. (2020) and Becker & Neumann

Table 3: SAC Hyperparameters

Hyperparameter	Low Dimensional DMC	Image Based DMC
Actor-Network	2×256 Unit NN with ReLU	3×1024 Unit NN with ELU
Critic-Network	2×256 Unit NN with ReLU	3×1024 Unit NN with ELU
Actor Optimizer	Adam with learning rate 3×10^{-4}	
Critic Optimizer	Adam with learning rate 3×10^{-4}	
Target Critic Update Fraction	0.005	
Target Critic Update Interval	1	
Target Entropy	$-d_{\text{action}}$	
Entropy Optimizer	Adam with learning rate 3×10^{-4}	
Initial Learning Rate	0.1	
discount γ	0.99	

(2022) respectively. The *SAC* baseline uses the hyperparameters listed in Table 3 and the results for *DreamerV3* Hafner et al. (2023) are provided by the authors[1].

D.2 Low Dimensional Tasks with Observation and Dynamics Noise.

To test the models' capabilities under uncertainties, we use the state-based versions of the tasks and add both observation and dynamics noise. The observation noise is sampled from $\mathcal{N}(0, 0.3)$ and added to the observation. The dynamics noise is also sampled from $\mathcal{N}(0, 0.3)$ and added to the action before execution. However, unlike exploration noise, this addition happens inside the environment and is invisible to the world model and the policy.

E Additional Results

We provide results for the individual tasks of the Deepmind Control Suite for image-based observations in Figure 5 and the different *KalMamba* ablations in Figure 6. Figure 7, shows the per-task results for the noisy state-based environments.

E.1 Runtime Analysis

To further investigate the runtime, we visualize the wall-clock time of a single SSM forward pass and a single training batch for different sequence lengths in Figure 8. While both the *RSSM* and *VRKN* scale linearly with the sequence length, *KalMamba* shows near-logarithmic scaling even for longer sequences thanks to its efficient parallelism. We expect further significant speedups for *KalMamba* with a potential custom CUDA implementation, similar to *Mamba*.

[1] https://github.com/danijar/dreamerv3

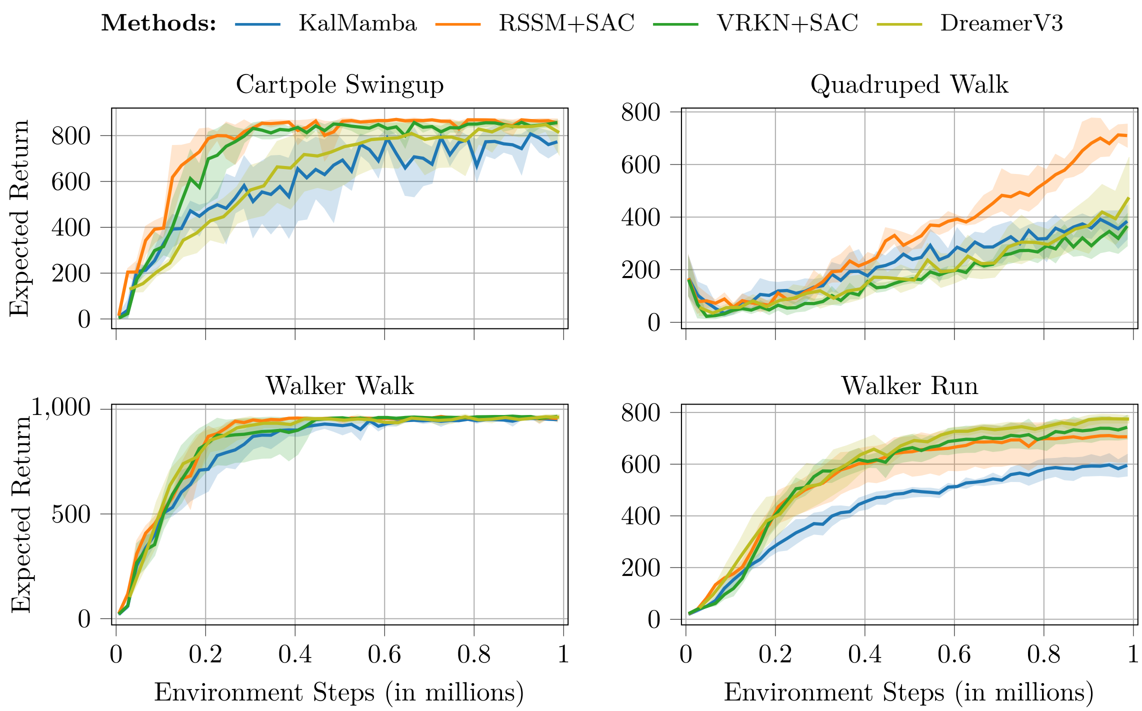

Figure 5: Task-wise evaluations of the DeepMind Control Suite on image-based observations. Dreamer-v3 shows a performance similar to RSSM+SAC.

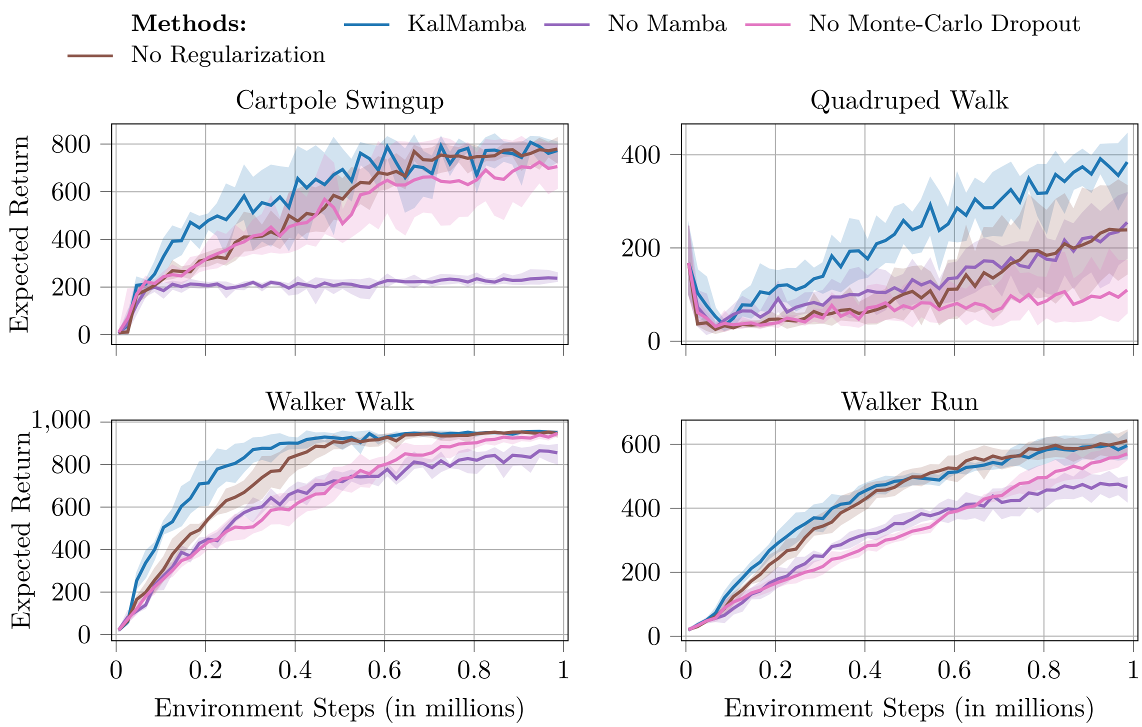

Figure 6: Task-wise evaluations of the DeepMind Control Suite for different *KalMamba* ablations. Monte-Carlo Dropout and the Mahalanobis regularization make the largest difference for the hardest task in the suite, i.e., `quadruped_walk`.

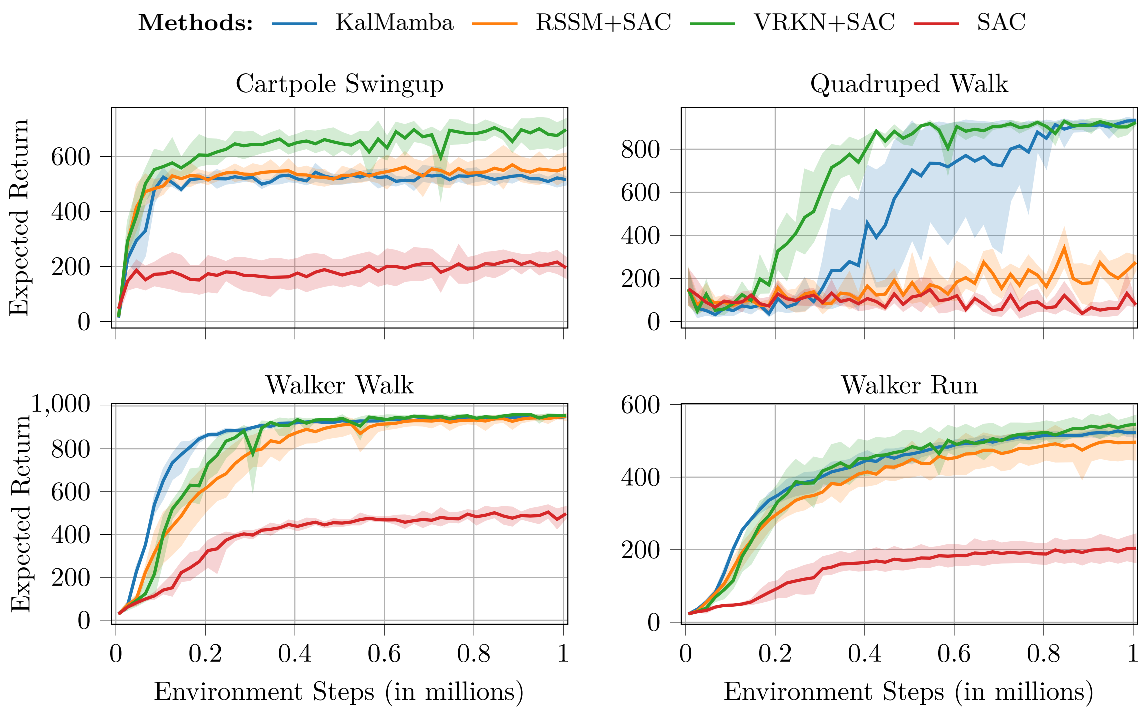

Figure 7: Task-wise evaluations of the DeepMind Control Suite on low-dimensional state representations. *KalMamba* performs on par with or better than the RSSM on all tasks, and is only outperformed by the computationally more expensive VRKN on `cartpole_ swingup`.

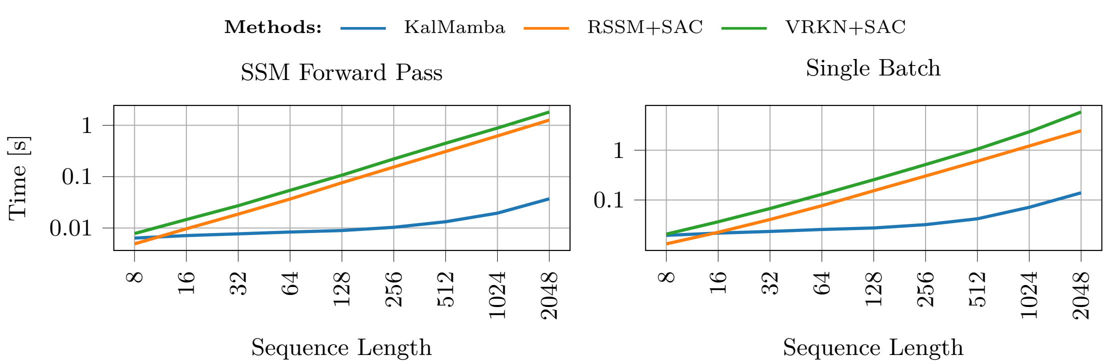

Figure 8: Runtime comparison of *KalMamba*, the RSSM and the VRKN for (**Left**) a SSM forward pass and (**Right**) a single training batch. While the computational cost of both baseline models scales linearly in the sequence length, *KalMamba* utilizes associative scans for efficient parallelism and thus near-logarithmic runtime on modern accelerator hardware.