Multimodal Sensor Fusion in Differentiable Filters Using A Learned Proposal Distribution - Final Report

Rabeh, Ali
Technical University of Munich
Munich, Germany
ali.rabeh@tum.de

Gerstewitz, Tim
Technical University of Munich
Munich, Germany
tim.gerstewitz@tum.de

I. Motivation

While classical Bayesian filters rely on analytical models, the recently proposed class of differentiable filters (DF) [1] learn transition or measurement models from data and can hence be employed when process models are not known exactly. This is the case for the often-encountered task of pushing, which has been shown to exhibit stochastic nature [4] due to a variety of unknown parameters such as friction coefficient or loss of contact. Pushing an object is furthermore a frequent task when manipulating large or heavy objects in the real-world; for robotic systems however, pushing requires the robot to know about the current system state to manipulate an object effectively.

Recent work [2] in differentiable filters has therefore investigated state estimation on a real-world pushing dataset [4] and combining sensor modalities to enhance reliability of the estimate and reduce uncertainty. In [3] on the other hand, a method to improve the efficiency of a DF for state estimation using a learned proposal distribution was presented and successfully validated in an in-hand manipulation experiment employing only tactile sensors.

The goal of this work was hence to combine the modified DF in [3] with the pushing scenario in [2]. Specifically, we wanted to investigate the modified DFs' performance using a learned proposal distribution incorporating multimodal information on the MIT Push dataset [4] compared to the standard DF used in [2].

II. Methods

A. Differentiable Particle Filters for Sensor Fusion

The dynamics of pushing in a real-world setting are inherently discontinuous due to a possible loss of contact. Additionally, it has been shown that the motion distribution of a pushed object exhibits multiple modes when pushed under identical initial conditions [4]. This motivates the choice of a particle filter for state estimation when pushing, since these filters can cope with both strong non-linearities and non-Gaussian distributions. As such, we chose a differentiable particle filter (DPF) to investigate sensor fusion in a pushing scenario.

Traditionally, fusing measurements in particle filters is done by combining the each sensor modality's observation model into a combined measurement model with which the filter state can be updated. However, by already conditioning the forward model on the current measurement one can also equip the filter with an informed proposal distribution while fusing measurements. This steers the particle distribution towards regions of higher likelihood given the current measurement, which in turn makes the filter more efficient and has been shown to also boost performance in DPFs.

To compare these two approaches, we chose to implement two DPFs: on the one hand a filter with a learned, measurement-informed proposal distribution, which takes in the current image of the pushed object into its forward model and the force acting on the pusher into its observation model. In the following this will be referred to as "learned proposal distribution filter". On the other hand, we also evaluate a filter which takes in both current image and current force via its observation model. Henceforth, this filter will be called "classic sensor fusion filter".

Additionally, we also report the performance of two unimodal DPFs, each incorporating either forces or images via their observation models.

B. MIT Push dataset

For evaluating the filters, we use the MIT Push dataset [4], which consists of over a million individual pushing sequences of eleven different-shaped objects pushed across four different surface materials recorded with a sampling rate of 250Hz. Each pushing sequence contains recordings of the pushed object's pose (x, y and orientation) as well as the end effector's pose of the robotic pusher and its forces and torques used during the push.

Since including visual information into the filter was one of the project's goals, we also render a binary image of the object's current pose as seen from a virtual camera 40cm above the pushing surface. The resulting images are 128 by 128 dimensions big, where each pixel corresponds to about 0.3cm in length.

For the results presented in this paper, we only used a subset of the MIT Push dataset, specifically trajectories for object ”rect1” (a small steel cuboid with length=width=45mm and depth=13mm) with a constant pushing velocity of 100mm/s and no acceleration during the push. We split this subset according to a $60/20/20$ split, resulting in 20 pushing sequences for training, 10 for validation and 10 for testing, with sequences typically lasting 2.5s and the longest push being roughly 5cm in length. All signals are downsampled to 50Hz for both training and evaluation.

III. Experimental Setup

A. Filter Architectures

When implementing the forward model in our filters, we used two different networks depending on the filter type: For the learned proposal distribution, we used an image encoder consisting of five convolutional layers with batch normalization and rectified linear units (ReLU) as activation functions. This reduces the $128x128x1$ input image to a $3x3x8$ feature map. This map is then flattened and concatenated with the current particle states and the relative pusher displacement compared to the last timestep. After that, this information is given into a multi-layer perceptron (MLP) with one hidden layer and four output units, which represent the displacement from the current timestep to the next in a continuous representation. For the classic sensor fusion filter, we then used the same MLP as forward model with the current particle states and the pusher displacements as inputs.

For its observation model, the learned proposal distribution filter used another multilayer perceptron, this time with two hidden layers, one output unit and ReLU activation functions. Here, we input both a particle's state in continuous representation and the current force measurement, receiving an estimate of the particle's log-likelihood at the output of the network. The classic sensor fusion filter on the other hand first encodes the current image in its observation model, using the same encoding architecture as described in the forward model of the learned proposal distribution filter. The encoded image is then again flattened and stacked together with the current force reading and a particle's state before being input into a multilayer perceptron with two hidden layers and a single output unit which again represents the particle's log-likelihood.

Both unimodal filter implementations derive from the classic sensor fusion filter; they use the same forward model and a similar observation model, with the force and image input missing for the filters using only images and only forces, respectively.

B. Filter Training

We first pretrained the filter forward model to minimize the error when propagating the current system state $1, 2, 4, 8$ and 16 steps into the future, using a mean-squared error loss function. Subsequently, we trained the complete DPF in an end-to-end fashion to again minimize the prediction error over these sequence lengths and employed the same loss. For the end-to-end training, we used an initial learning rate of $1e-5$ with an exponential decay of 0.9 per epoch and trained for 50 epochs per sequence length. Furthermore, we used 100 particles here and enabled soft-resampling, since this empirically improved the performance of the observation models and so we can ensure that particles with higher weights have a greater influence on the estimation process and are therefore are more likely to be selected, while particles with lower weights have a lower chance of being selected.

IV. Results

We evaluated our filters' performance on 10 test sequences and report results based on the tracking root-mean squared error compared to the ground truth. Since particle filters exhibit stochasticity, we evaluated each filter three times across our test dataset and averaged the resulting RMSE metrics. The results can be seen in Table I. During testing, we ran the filters with soft-resampling enabled and used 1000 particles.

		Models			
		only Forces	only Images	Forces and Images	Proposal Distribution
RMSE in cm and rad	Pose x	0.92	1.01	**0.79**	0.87
	Pose y	0.79	**0.77**	0.79	1.15
	Pose theta	0.24	0.24	0.26	**0.23**
	Position	0.86	0.90	**0.79**	1.02

TABLE I: Root-mean squared error of all trained filter models on the test set. The results are averaged over three runs across the entire test set for each model.

As can be seen, the ”classic” sensor fusion filter performs best with respect to the position error, while the learned proposal distribution filter surprisingly performs worst on our test set.

To make the filter behavior more intuitive, we show the results of all filter models on test sequence 4. The unimodal filter using only forces as observations can be seen in Fig. 1: While it tracks the trend of the x-position reasonably well, the y-position and the orientation start to diverge after around 20 and 10 steps respectively. This diverging behavior can also be seen in the other unimodal filter which uses only images as measurements in Fig. 2. Here, the ground truth y-position is tracked comparably to the only-force filter, starting to diverge after around 20 steps, but both x-position and orientation are tracked poorly, resulting in an overall worse estimate.

Combining the sensor modalities through a shared observation model then leads to an increase in filtering performance, as depicted in Fig. 3. While the pushed object's y-position is still tracked similarly to the unimodal filters, the x-position is tracked with significantly more accuracy. Additionally, although the orientation starts to diverge here after 10 to 15 steps too, the overall trend is still captured by the model.

This also holds true for the orientation tracking in the learned proposal distribution filter up to step 21, which can be seen on the bottom right of Fig. 4. The estimated orientation is maintained quite close to the true object orientation up until

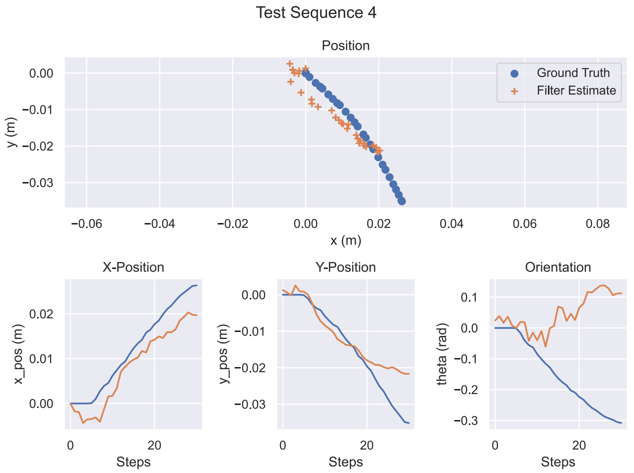

Fig. 1: Component-wise results of the unimodal filter using *only forces* as observations on test sequence 4.

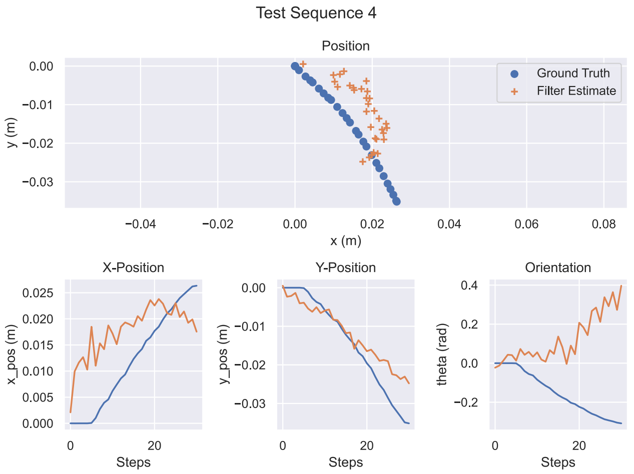

Fig. 2: Component-wise results of the unimodal filter using *only images* as observations on test sequence 4.

this point, before it starts to diverge quickly. Furthermore, in this instance, the model estimates the x-position with a, almost constant offset, although this is likely a random effect based on our observations. Lastly, the y-position is tracked quite poorly here compared to the other models, leading to an overall worse estimate than the classic sensor fusion filter on this sequence.

V. Discussion and Conclusion

The proposal distribution filter's larger error compared to the classic sensor fusion architecture is surprising given that both models take in the same information. One possible reason might be that the proposal distribution is harder to learn than a simple forward model. However, the relatively poor tracking of the unimodal filter using only images suggests that the chosen architecture for the image encoding might not be able to extract the pose information from the image input. This could

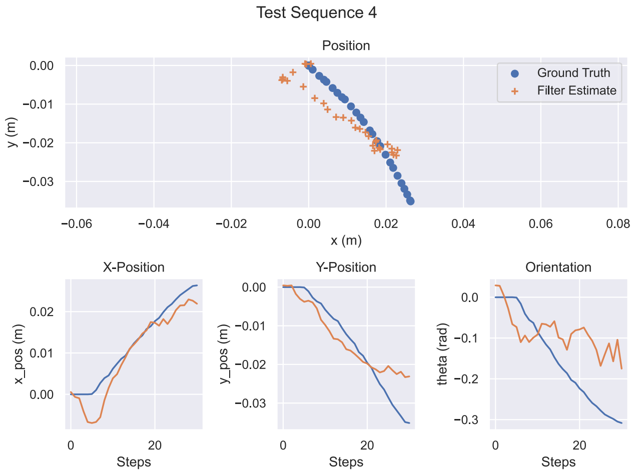

Fig. 3: Component-wise results of the multimodal filter using *both forces and images* as observations on test sequence 4.

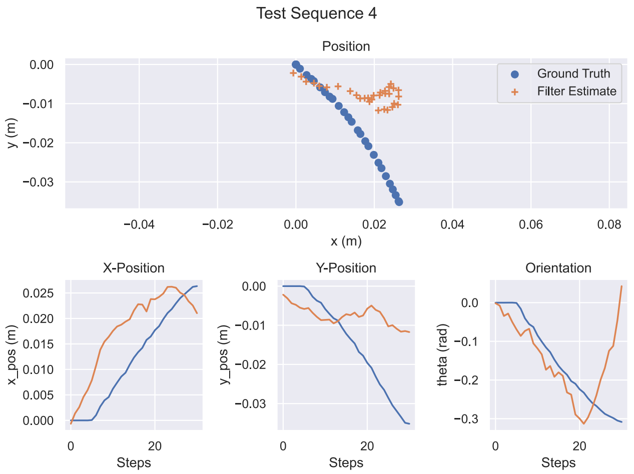

Fig. 4: Component-wise results of the multimodal filter using *images for the proposal distribution and forces as observations* on test sequence 4.

be checked by trying to predict the object position directly from the image input and this could be used as a benchmark.

Another possible investigation direction relates to the rapid divergence of some estimates after around 20 steps. While we initially thought this was related to training the filters to only predict 16 steps into the future, however this behavior did not change after training for 32 steps.

Based on our project's results, it is lastly advantageous to combine sensor modalities via a shared observation model compared to learning a measurement-informed proposal distribution. This seems to be highly architecture-dependent with no clearly optimal architecture however, motivating further research.

References

[1] A. Kloss, G. Martius, and J. Bohg, “How to Train Your Differentiable Filter”, in Autonomous Robots, Vol. 45, 2021, pp. 562—578.

[2] M. A. Lee, B. Yi, R. Martin-Martin. S. Salvarese, and J. Bohg, “Multimodal Sensor Fusion with Differentiable Filters”, in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020, pp. 10444–10451.

[3] L. Roestel, L. Sievers, J. Pitz, and B. Baeuml, “Learning a State Estimator for Tactile In-Hand Manipulation”, in 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2022, pp. 4749–4756.

[4] K. T. Yu, M. Bauza, N. Fazeli, and A. Rodriguez, “More than a Million Ways to Be Pushed: A High-Fidelity Experimental Data Set of Planar Pushing”, in 2016 IEEE/RSJ Internation Conference on Intelligent Robots and Systems (IROS), 2016, pp.30–37.