# Differentiable Particle Filters: End-to-End Learning with Algorithmic Priors

Rico Jonschkowski Divyam Rastogi Oliver Brock

May 2018

## 1. Motivation

The architecture proposed in this work is motivated by the benefits of:

- **End-to-end learning** over learn the components of an architecture separately since it provides more generality, and
- **Algorithmic Priors** in the context of end-to-end learning, since a naive application of the latter may lead to overfitting.

The authors cite here the success of **CNNs**, an algorithmic prior for end-to-end learning of architectures for image-based tasks like segmentation.

## 2. Problem Statement

The goal we are working towards is estimating the state $z_[1:T]$ in a State Space Model:

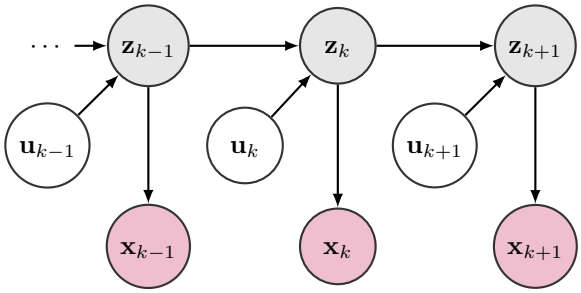


Figure 1. State Space Model

given just the observations $x_{1:T}$ and actions $u_{1:T}$. Essentially, what we want is a probability distribution over states at every time step $t$ (to account for uncertainty). We often encode this in the form of filtering distributions or beliefs:

$$belz_t = p(z_t|x_{1:t}, u_{1:t}) \tag{1}$$

## 3. Theoretical Background for solution

### 3.1. Bayes Filter: Algorithmic Priors for State Estimation in Robotics

The authors argue that the **Bayes Filter** is a suitable algorithmic prior for learning a model that can estimate the system state (for systems that follow a generative model like the one in Figure 1).

The Bayes Filter calculates the belief over states at each point of time using two steps iteratively over the whole sequence:

- **Prediction:** This step predicts a distribution over the next state $z_t$ given the belief over the current state $z_{t-1}$. This is essentially done by calculating an expectation of the generative transition step $p(z_t|z_{t-1}, u_{t-1})$ w.r.t the belief over the current state:

$$bel(z_t) = \int p(z_t|z_{t-1}, u_{t-1})\overline{bel}(z_{t-1})dz_{t-1} \tag{2}$$

- **Measurement:** This step updates the predicted belief $bel(z_t)$ to $\overline{bel}(z_t)$ by incorporating the measurement $x_t$:

$$\overline{bel}(z_t) = \eta p(x_t|z_t)bel(z_t) \tag{3}$$

where $\eta$ is the normalization constant ($p(x_t|x_{1:t-1})$?)

Essentially,

$$bel(z_t) = p(z_t|x_{1:t-1}), \text{ and} \tag{4}$$

$$\overline{bel}(z_t) = p(z_t|x_{1:t}) \tag{5}$$

### 3.2. Particle Filter

Particle Filters are a class of Bayes Filters that *approximate* the belief over states with several particles or samples $z_t^{[1]}, z_t^{[2]}, ...z_t^{[N]}$ that associated with corresponding weights $w_t^{[1]}, w_t^{[2]}, ...w_t^{[N]}$. The Bayes update steps in this context are:

- **Prediction** Sample N particles using a proposal distribution $q(z_t)$ (usually taken to be the generative transition distribution - $p(z_t|z_{t-1})$):

$$z_t^{[1]}, z_t^{[2]}, ...z_t^{[N]} \sim q(z_t) \tag{6}$$

- **Measurement** Calculate the un-normalized weights $\overline{w}$ for each of the particles using the emmision ($p(x_t|z_t)$), transition and the proposal distributions:

$$\overline{w}_t^{[i]} = \frac{p(x_t|z_t)p(z_t|z_{t-1})}{q(z_t)} \tag{7}$$

Re-sample the N particles from a categorical distribution with parameters $w_t^{[1]}, w_t^{[2]}, ...w_t^{[N]}$, the normalized weights.

# 4. Method - DPF

In line with their motivation for end-to-end learning, the authors propose Differentiable Particle Filters, in which one learns the parameters ($\theta$) of the functions for each of the components necessary for a particle filter:

- **Action Sampler** $f$ creates a noisy action to inject uncertainty into the transition model:

$$\hat{u_t} = u_t + f_\theta(u_t, \epsilon \sim \mathcal{N}(0,1)) \tag{8}$$

- **Generative Transition Model** $g$ predicts the next state given the current state and action:

$$z_{t+1} = z_t + g_\theta(z_t, \hat{u_t}) \tag{9}$$

- **Encoder** $h$ encodes the current observation $x_t$ into a low dimensional encoding:

$$e_t = h(x_t) \tag{10}$$

- **Proposer** $k$ predicts the state given the encoding:

$$z_t^{[i]} = k(e_t, \delta^{[i]} \sim B)) \tag{11}$$

where $\delta^{[i]}$ is a dropout vector sampled from a Bernoulli distribution. This is to induce noise while getting the N particles.

- **Emmision** $l$ weighs each particle based on the observation

$$w_t^{[i]} = l(e_t, z_t) \tag{12}$$

The belief $bel(z_t)$ in this context is characterized by a set of particles and their corresponding weights, i.e:

$$bel(z_t) = \{Z_t, W_t\}, \text{ where} \tag{13}$$

$$Z_t = \{z_t^{[i]}\}_{i=1}^N \text{ and } W_t = \{w_t^{[i]}\}_{i=1}^N$$

The previous belief $bel(z_{t-1})$ is updated by the next set of particles which are either obtained by:

- **Prediction and Measurement update / Resampling**: State particles are obtained using the action sampler 8 and the generative prediction model 9, followed by a resampling according to weights that are obtained from 12

- **New particles from the proposer** $k$:
New particles are obtained by directly using the observation $x_t$ at time step $t$ and their corresponding weights are calculated according to equation 12

# 5. Training