Fusing Vision & Proprioception with Differentiable Particle Filtering

Brent Yi // brentyi@stanford.edu

(1) Overview

Our project explores the application of differentiable particle filtering to sensor fusion, particularly in the context of robot manipulation.

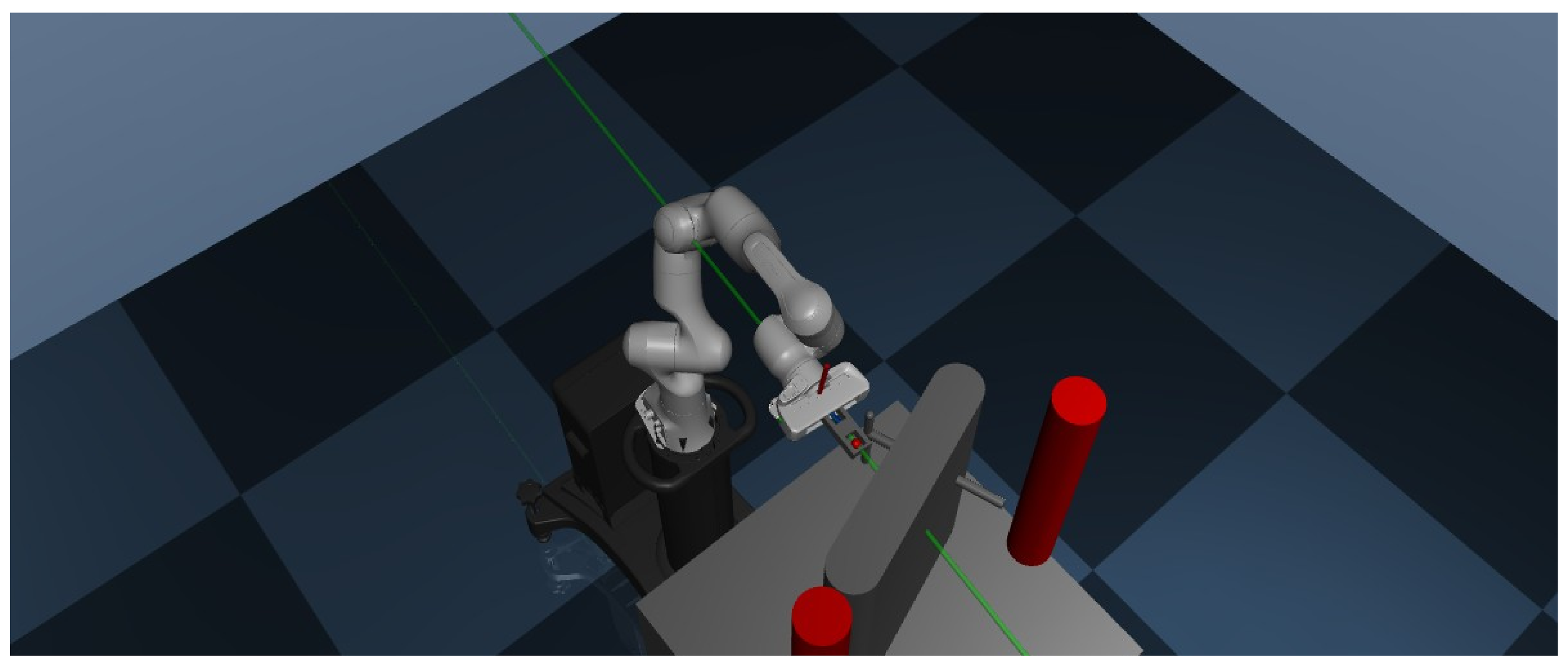
Specifically, we're interested in answering:

- How much can a DPF infer about the state of its environment by fusing two very different sensing modalities: vision & proprioception?

- How does this compare to a simpler model? How does training end-to-end compare to learning PF components in isolation?

- Can we use our estimate as a policy/controller input?

(2) Task & Data Collection



Environment: door opening with a Franka Panda in MuJoCo

Goal: learn a model that estimates the door state given continuous proprioception observations (every timestep) and sparse, low-resolution camera observations (every 10 timesteps, 32x32)

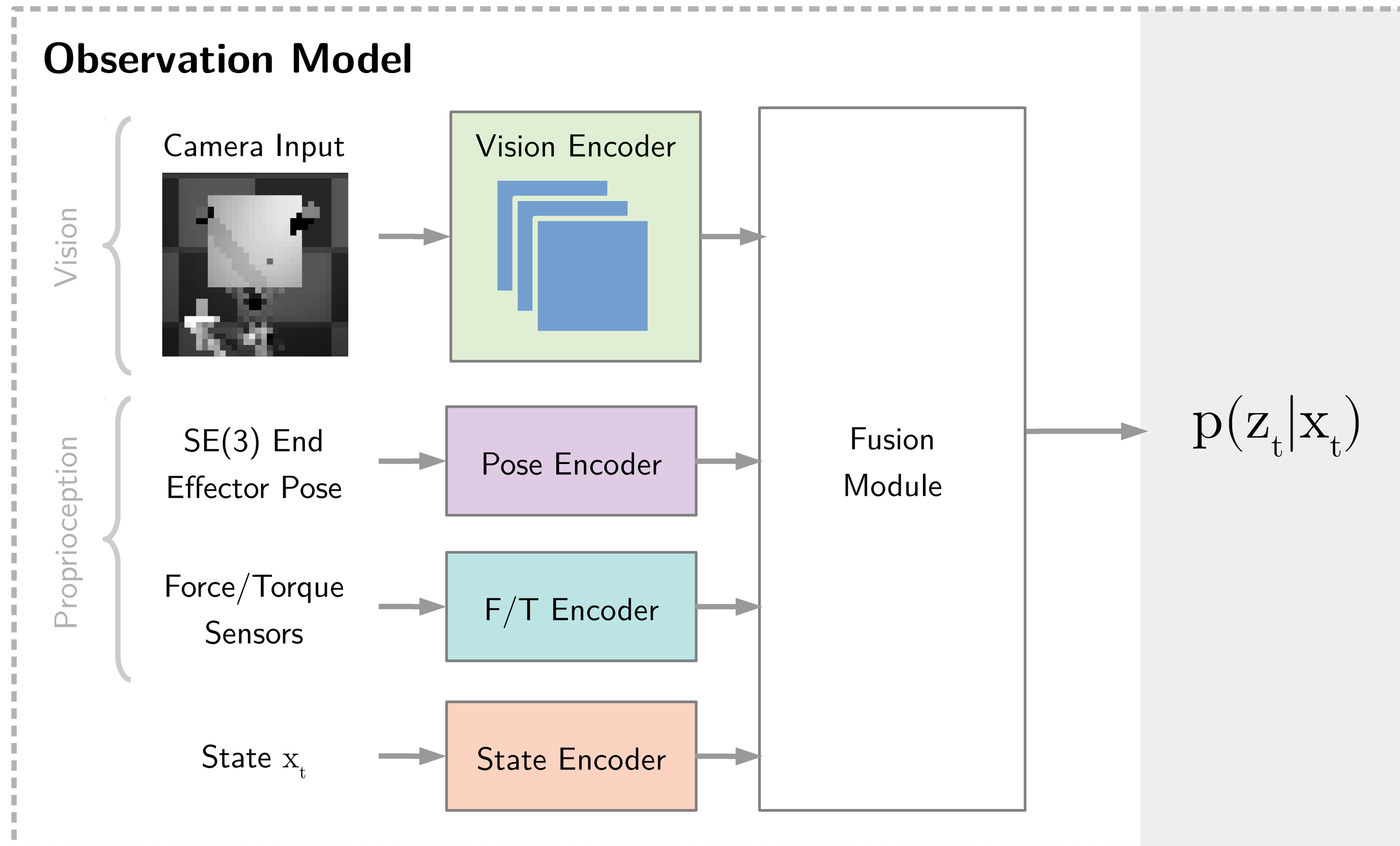
Data: we randomly generated ~900 trajectories from two categories:

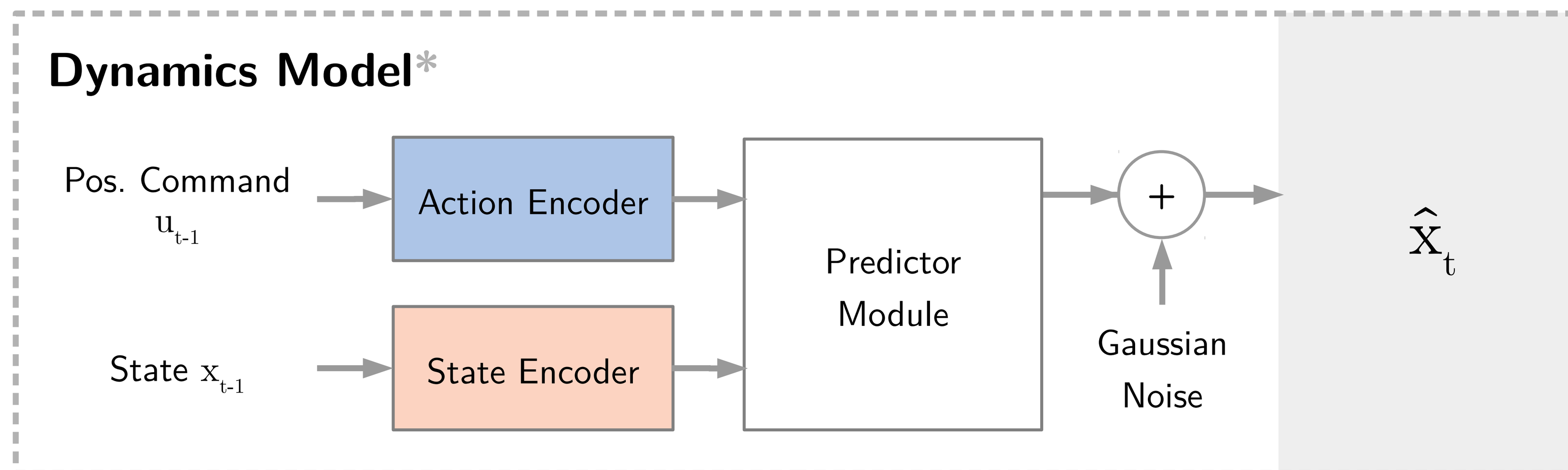
- **"On-handle"**: the robot gripper is enclosed around the door handle, and the robot uses it to move the door. Trajectories were randomly generated based on a human demonstration.

- "**Off-handle**": the robot directly pushes or pulls part of the door, without using the handle.

(3) Particle Filter Models

These models are used as components of a standard particle filter algorithm.

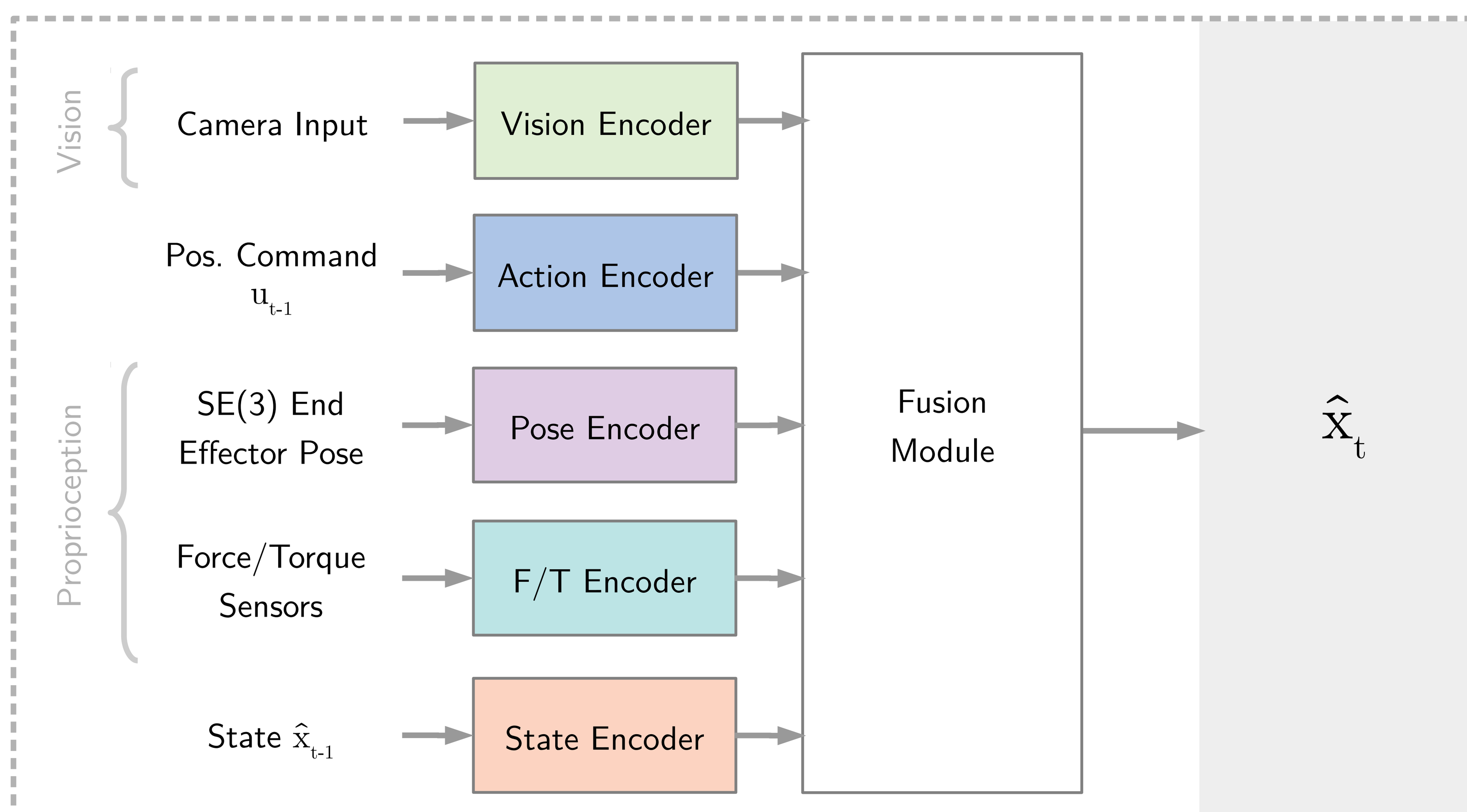




*For our task, which has a 1D state, a learned transition model performed almost the same as just noise. This was disabled for final experiments.

(4) Regression Model

This baseline directly estimates state from the previous state estimate, observation, & control.

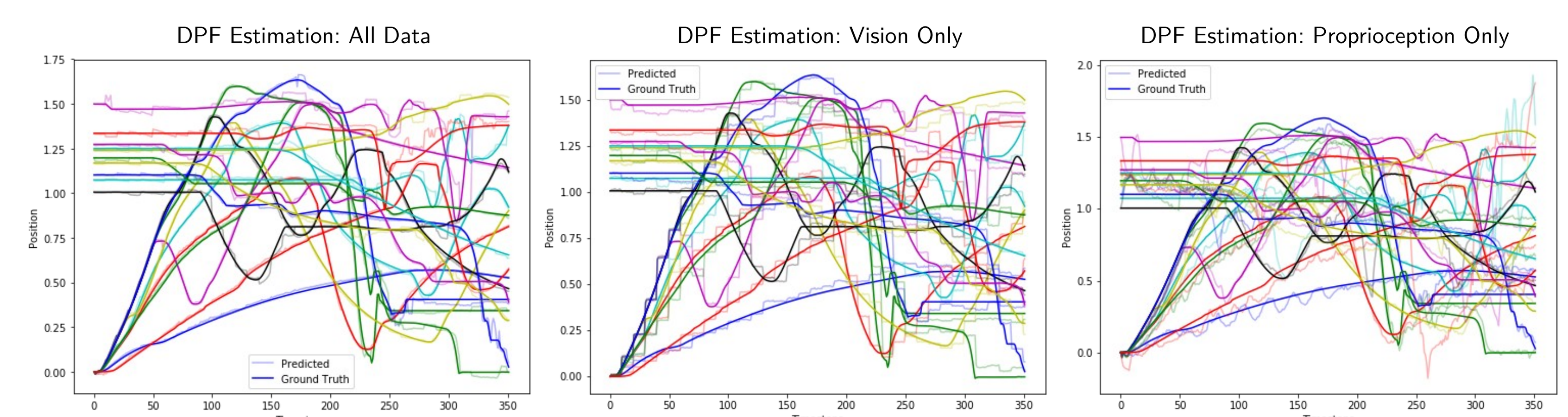


(4) Experiments

Three state estimation models: **(1)** baseline regression model, **(2)** particle filter baseline with components learned in isolation, and **(3)** differentiable particle filter (DPF) trained end-to-end by backpropagating estimation errors though the particle filter algorithm.

A version of each model was trained with: **(a)** all observations, **(b)** proprioception observations only, and **(c)** vision observations only.

	Training Set Position MSE			Validation Set Position MSE		
	On-handle	Off-handle	All	On-handle	Off-handle	All
Regression Baseline	0.00118	0.00118	0.00118	0.00096	0.001681	0.00131
Regression propr. only	0.00556	0.24904	0.12496	0.00951	0.03056	0.01682
Regression vision only	0.00217	0.00099	0.00136	0.00218	0.00201	0.00210
PF Baseline	0.00063	0.00103	0.00075	0.00067	0.00179	0.00121
PF proprioception only	0.00940	0.18695	0.09971	0.01931	0.03732	0.02039
PF vision only	0.00246	0.00150	0.00168	0.00265	0.00201	0.00233
DPF	0.00023	0.00044	0.00031	0.00030	0.00102	0.00063
DPF proprioception only	0.00564	0.06528	0.03075	0.01074	0.03437	0.01850
DPF vision only	0.00250	0.00141	0.00175	0.00259	0.00193	0.00234



Finally, we integrated our state estimates into a control system. We hand-designed a controller to move the door to a random goal position. Average final errors over 10 runs:

State Estimator	**Ground-truth**	**Regression**	**PF Baseline**	**DPF**
Controller MSE	$8.62*10^{-12}$	$7.1*10^{-4}$	$2.4*10^{-4}$	$4.8*10^{-5}$

(5) Initial Conclusions

End-to-end training was key for effective sensor fusion: DPF validation MSE was within 10% of both baselines when only one sensing modality was available, but the MSE was ~2x lower with both modalities.

Algorithmic prior & computation power alone weren't enough: despite the additional complexity of our particle filter baseline model and its worse runtime, MSE was barely lower than our regression baseline.