16-833-SLAM: Final Project Midterm Report

Russell Schwartz, Ian Krause

November 10th, 2023

Midterm Report: “Extensions to the Unscented Kalman Filter”

Abstract

The UKF achieves better performance over the EKF in terms of the quality of the posterior gaussian approximation. In part, it does this by consulting the dynamics model on a larger portion of the state space through its selection of so-called sigma-points. In this project we propose to explore extensions to the usual sigma-point selection scheme, including schemes that select more than $2n + 1$ points, use more than one shell of points, and/or make use of higher-order information. We plan on comparing these novel methods to baseline EKF and UKF implementations in simulated environments, including a SLAM task.

In this midterm report we summarize our progress on this project thus far, including a discussion of our baseline KF, EKF, and UKF implementations, as well as our plans for the immediate future.

1 Background

The EKF extends the Kalman Filter by approximating the system dynamics (and measurement model) via 1st-order taylor expansion at the current state estimate. The EKF is non-optimal, in the sense that the resulting gaussian is (in general) not the best-fitting gaussian for the underlying posterior.

One way to compute this best-fitting gaussian (or at least a very close approximation to it) would be to use a histogram-based approach, where we discretize the state space into a set of bins, pass each bin through our non-linear dynamics, and then fit a gaussian to the resulting histogram-distribution. However, this approach is impractical (and it defeats the purpose of using a parametrized belief in the first place).

The UKF provides a better approach, which is more akin to a particle filter. A set of so-called ”sigma points” are sampled from the n-dimensional state space in accordance with the prior:

- 1 point at the mean
- $2n$ points distributed symmetrically about the mean, at a fixed distance (mahalanobis distance)

We pass each of these points through our non-linear dynamics, and then fit a gaussian to the resulting particle-distribution, taking care to weight the points relative to their probability density in the prior. Impressively, the resulting estimate captures the posterior mean and covariance accurately to the 3rd order.

We aim to extend the existing UKF implementation to multiple sigma point shells in the hope of improving filtering and performance in comparison to the standard EKF and UKF filters.

2 Scope Discussion

Our scope for this project has changed more so than it has increased. We still aim to implement additional UKF sigma point shells for our original 2D SLAM system. However, we have decided to test our filter implementation against a larger variety of systems. These systems with differ in terms of degrees complexity and non-linearity. We plan to evaluate if performance increases from our inclusion of additional sigma point shells is system dependent in comparison to performances from standard EKF and UKF filters.

3 Progress Update

We have made significant progress thus far on implementing the baselines and surrounding frameworks for our filtering experiments. A number of major items are complete (all in python):

1. Framework for defining general dynamics system models (including auto-differentiation)
2. Baseline implementations of Kalman Filter, EKF, and UKF
3. Example systems and EKF vs UKF comparison (including 2D non-linear SLAM)
4. Tools for computing and visualizing explicit uncertainty propagation

3.1 System Models Framework

We implement a family of classes for representing (very generically) dynamic systems which may be candidates for filtering. The base `SystemModel` class is a high-level class representing a dynamic system, including a dynamics model, measurement model, and corresponding noise models. The constructor requires the dimensions of the following spaces:

- `state_dim`: dimension of the state space (x)
- `control_dim`: dimension of the control input space (u)
- `measurement_dim`: dimension of the measurement space (z)
- `dynamics_noise_dim`: dimension of the dynamics noise vector space (w)
- `measurement_noise_dim`: dimension of the measurement noise vector space (v)

in addition to four callable functions:

- `dynamics_func`: $(x, u, w) \to x$
- `measurement_func`: $(x, v) \to z$
- `dynamics_noise_func`: $() \to w$
- `measurement_noise_func`: $() \to v$

which together define the system and its noises. In addition, we define four subclasses

- `GaussianSystemModel(SystemModel)`: both process noise and observation noise are (potentially non-additive) zero-mean gaussians
- `DifferentiableSystemModel(SystemModel)`: differentiable dynamics and measurement models (requires explicit jacobians)
- `AutoDiffSystemModel(GaussianSystemModel, DifferentiableSystemModel)`: automatically differentiable dynamics and measurement models (amenable to EKF, no need to provide explicit jacobians)
- `LinearSystemModel(GaussianSystemModel, DifferentiableSystemModel)`: linear dynamics, linear measurement model, and additive gaussian noises (amenable to KF)

Automatic differentiation is accomplished by the `JAX` library.

3.2 Baseline Implementations

We implement the Kalman Filter, Extended Kalman Filter, and Unscented Kalman Filter as baselines for comparison (as well to serve as starting points for novel-filter development). These implementations are built to operate on the generic system models defined above, and are effectively very modular. They each implement a `predict_step(x, u)` and `update_step(z)` method for filtering control inputs and measurements respectively. When instantiating a filter, the user specifies the system model that will be operated on.

The `ExtendedKalmanFilter` implementation requires a system which is both a `GaussianSystemModel` and a `DifferentiableSystemModel` as its target system. This allows it to call to the system's jacobian getter methods, which may be automatic.

The `UnscentedKalmanFilter` implementation is a bit more general as it only requires a `GaussianSystemModel`. The user has the opportunity to specify a custom `SigmaPointSelector`, whose job it is to select sigma points given a prior mean and covariance. Currently, only the standard sigma-point selection scheme is implemented, which includes tunable scaling parameters α, β, κ, and options for the matrix-square-root algorithm including cholesky- and eigen-decomposition based methods. See Figure 1.

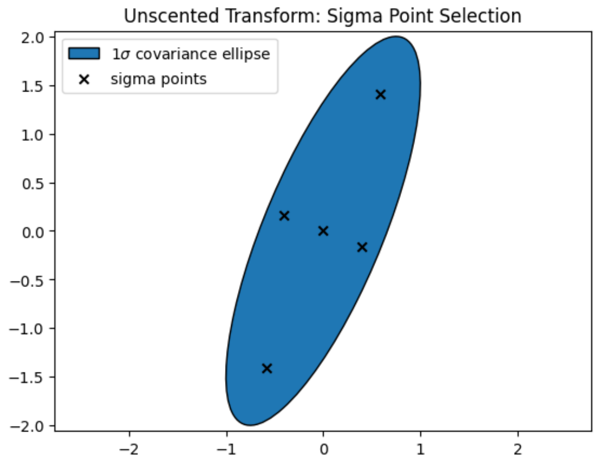

Figure 1: Example of sigma-point selection in 2D using $\alpha = 0.5$, $\beta = 2.0$, $\kappa = 0.0$, and eigendecomposition-based matrix square root

3.3 Example Systems

To verify our filter implementations, we define some example systems with varying levels of complexity and non-linearity.

- 1D Linear Kinematic Car Model: The 1D Kinematic car model is a quintessential linear system (a double-integrator). It describes the motion of a car in one dimensional space due to force (acceleration) based inputs. This system contains very few states and is therefore quite easy to visualize. See Figure 2.

- 1D Nonlinear Kinematic Car Model: A nonlinear variant of the 1D kinematic car model that adds a drag term with a nonlinear (quadratic) dependence on velocity. See Figure 3

- 2D SLAM: The 2D SLAM system is a slightly more complex system and is also non-linear. This system estimates the position and orientation of a robot as well as a certain number of landmark locations. Inputs to the system include a sequence of directional robot velocity commands as it moves. Landmark measurements consist of bearing and range. See Figure 4

- Mackey-Glass: The Mackey-Glass system can be derived from the Mackey-Glass equations that form the system dynamics. These equations are set of differential equations that can exhibit chaotic behavior.

This system is known for being particularly challenging apply a filter to for the purpose of state estimation. (Implementation ongoing)

- Double Inverted Pendulum: The double pendulum exhibits highly non-linear behavior. By putting the pendulum on a movable linear cart, we obtain a potentially controllable system, one that has received much attention. Different types of measurements can be modeled. (Implementation ongoing)

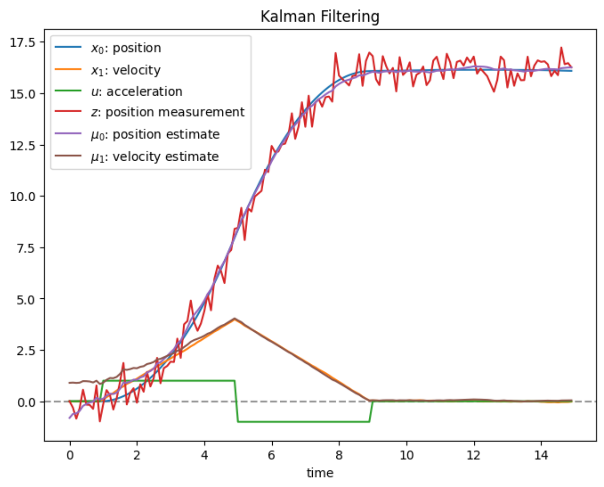

Figure 2: Kalman Filtering on the 1D Linear Kinematic Car example system

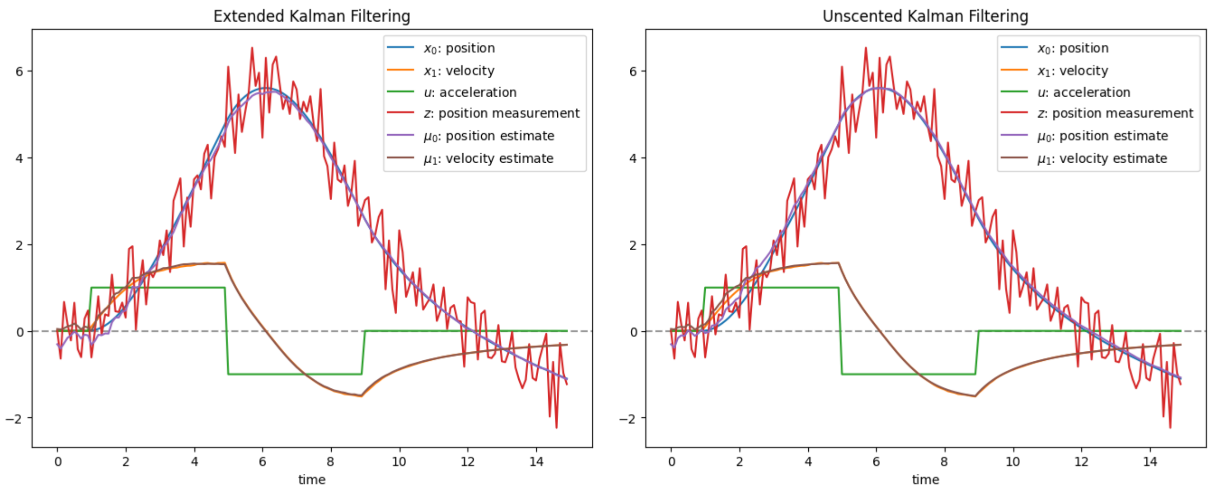

Figure 3: Extended and Unscented Kalman Filtering on the 1D Nonlinear Kinematic Car example system. The UKF performs marginally better on this system.

3.4 Uncertainty Propagation

In addition to these end-to-end filtering applications, we develop some tooling for directly computing and visualizing uncertainty propagation through arbitrary functions. These will be useful to understand the affect of nonlinearity and to evaluate our approximations. See Figure 5.

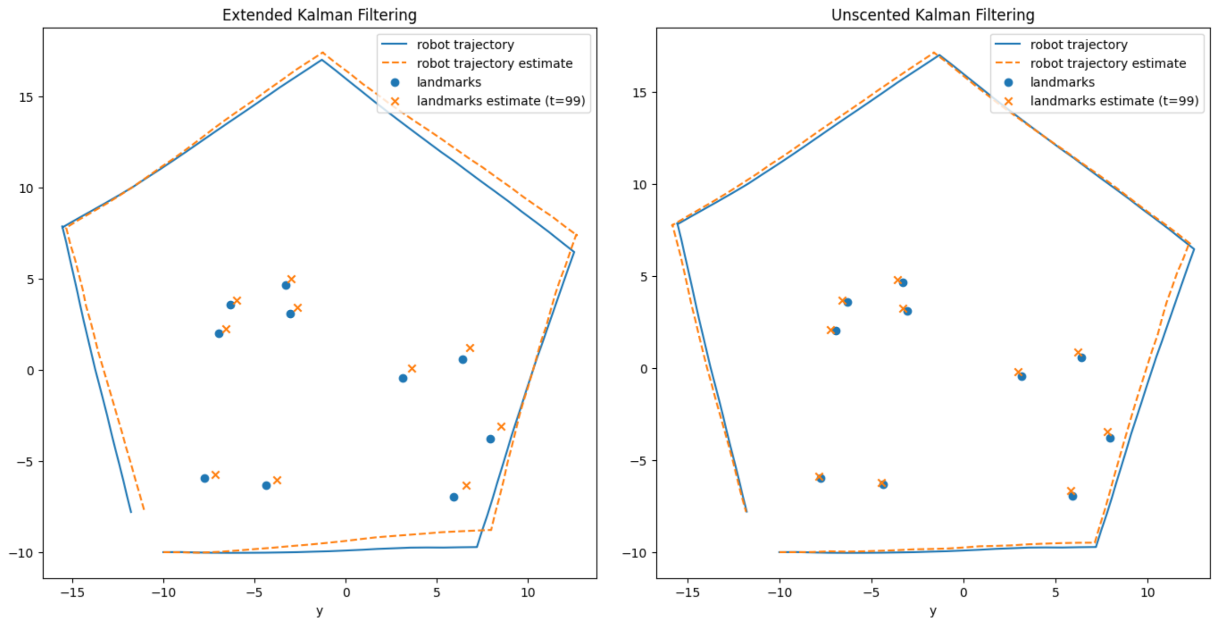

Figure 4: Extended and Unscented Kalman Filtering on the 2D Nonlinear SLAM system. The UKF performs marginally better on this system.

4 Updated Timeline

Our next steps mainly consist of implementing the proposed extensions to the UKF, and evaluating them on a variety of systems.

1. Polish existing code
2. Finish data parsing and reconstruction node with example
3. Finish Mackey-Glass system implementation
4. Run experiments and log data
5. Compare various filter implementations quantitatively
6. Iterate filter designs
7. Prepare and practice presentation (Before Dec 4th)
8. Write and submit final report (Before Dec 13th)

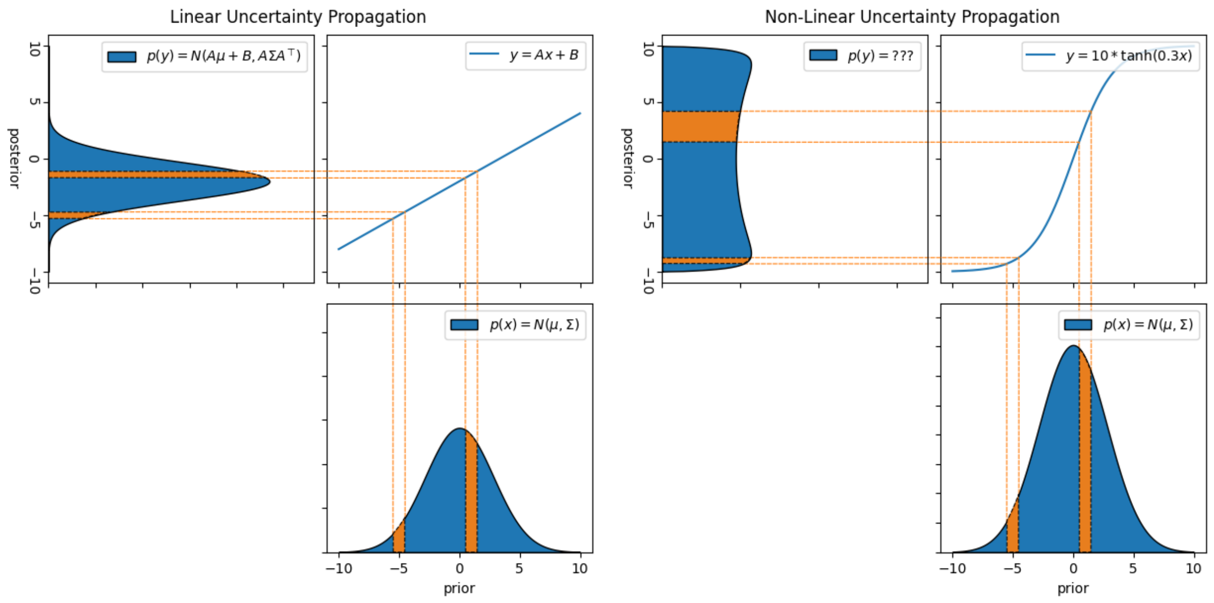

Figure 5: Uncertainty propagation of a gaussian prior through a linear function (left) and a nonlinear function (right). Select segments of the prior and their images in the posterior are highlighted to show examples of mass transport explicitly. The image under the linear function is an easily-parametrized gaussian, while the image under the nonlinear function is not. Indeed, the nonlinear image is generally not parametrizable by any common distribution family.

References

[1] Yiping Cheng and Ze Liu. Optimized selection of sigma points in the unscented kalman filter. In *2011 International Conference on Electrical and Control Engineering*, pages 3073–3075, 2011.

[2] Simon J. Julier and Jeffrey K. Uhlmann. New extension of the Kalman filter to nonlinear systems. In Ivan Kadar, editor, *Signal Processing, Sensor Fusion, and Target Recognition VI*, volume 3068, pages 182 – 193. International Society for Optics and Photonics, SPIE, 1997.

[3] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[4] Michael C. Mackey and Leon Glass. Oscillation and chaos in physiological control systems. *Science*, 197(4300):287–289, 1977.

[5] E.A. Wan and R. Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, pages 153–158, 2000.