Update - Kalman Filter

João Böger

Sprace

Outline

Semester summary

- QFT I - Prof. Gustavo Alberto Burdman at USP as non-degree student UNDERWAY

Research

- Particle Dark matter
  - Escola de Matéria Escura - Farinaldo Queiroz DONE
  - Intro to Particle Dark Matter - Stefano Profumo UNDERWAY
- Standard Model
  - Intro to SM - S. Novaes UNDERWAY
- Tracking sequences
  - HLTIterativeTrackingIter02 DONE
  - Kalman Filter UNDERWAY
  - Add to HLT-Phase2's wiki on GitHub PLANNED

Kalman Filter - I

To better understand how the Kalman filter works and future application on my research I worked a tutorial following the principles of the algorithm [1].

Kalman filter effectively combines several sources of uncertainty to provide a more accurate estimate of the state of the system than any measurement by it's own. It can be divide in two steps:

- **Prediction step**: The system's state and uncertainties are determined by a **dynamic model**.
- **Update step**: Uses **measurements** with system's **prediction** to obtain a final state through a **weighted average**.

[1] The Kalman Filter in 1D using Python: Example - 1D Localization

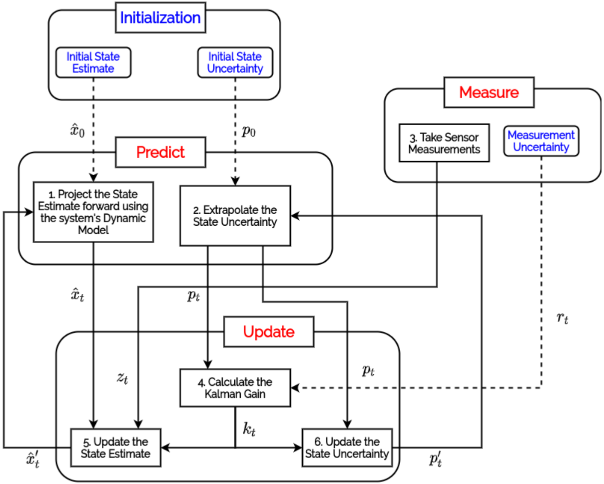


Figure: Kalman Filter 1D Flowchart.pdf

Recipe for a simple Kalman Filter

In order to implement a first approach to Kalman filter we can proceed as follows:

- **Define the state** Initial position of the robot.
- **Define the Motion Model** Linear motion with constant velocity.
- **Define how the measurements will be made** In order only to understand the algorithm I used random values.
- **Define the uncertainties q and r** Same thing, used random values.
- **Implement and test the filter** I used different values of q and r in order to verify if the final state was following the Kalman Gain k_t.

Kalman Filter Variables Definition

- x_t Predicted state estimate
- v_t Velocity
- Δt Time interval between two steps
- h_t Uncertainty of the predicted state estimate
- h_t^v Uncertainty of the velocity estimate
- q_t Process Noise Variance
- z_t Observation/measurement of the true state of x_t
- r_t Measurement uncertainty
- k_t Kalman gain
- x_t' Updated state estimate
- h_t' Updated variance of the state estimate

Kalman Filter Equations

Thus our Kalman filter will work as follows:

Prediction

$$x_t = x_{t-1} + v\Delta t \qquad \text{(Predicted state estimate)}$$

$$h_t = h_{t-1} + \Delta t^2 \cdot h_t^v + q_t \qquad \text{(Uncertainty associated)}$$

Update

$$k_t = \frac{h_t}{h_t + r_t} \qquad \text{(Kalman gain)}$$

$$x'_t = x_t + k_t(z_t - x_t) \qquad \text{(Updated state estimate)}$$

$$q'_t = p_t(1 - k_t) \qquad \text{(Updated uncertainty of the state estimate)}$$

Kalman Gain k_t

The Kalman filter estimates the state of the system through a **weigthed average** of the system's prediction and measurements. The **weights** are important once they codify if the filter trusts more on the measurements or the model

- $k_t \approx 1$ Means **less uncertainty** in the **measurements**. Therefore the filter trusts more the measurements than the model.
- $k_t \approx 0$ Means **less uncertainty** in the model **prediction**. Thus the filter follows the predictions more closely than the measurements.

The resulting **weighted average** places the updated state **somewhere between the prediction and the measurement** generally providing a more accurate estimate of the state of the system.

Kalman Filter for 2D particle travelling through 5 concentric detectors

The simulation implemented here has the following setup:

- The particle starts from the origin $(x, y) = (0, 0)$.
- It's initial velocity is unity and forming $45^{\underline{o}}$ with the x-axis.
- Time steps were set to $\Delta t = 1$.
- Random values were used for measurements (z_t), measurement uncertainty(r_t),Process Noise Variance q and uncertainty of the velocity estimate (h_t^v).

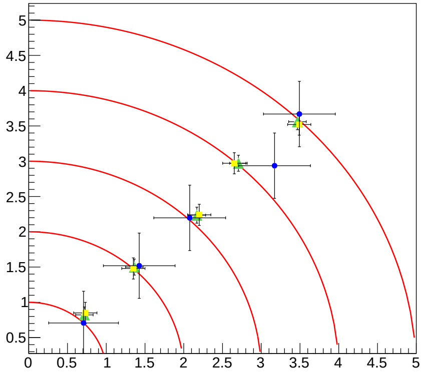


Figure: 2D particle travelling through 5 concentric detectors.

Kalman Filter for 2D particle travelling through 10 concentric detectors

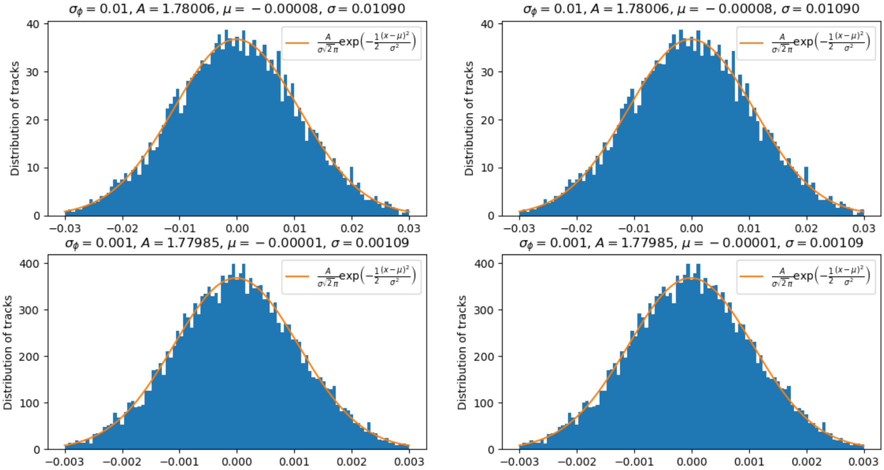
The next step of our study was to analyze the efficiency at which our Filter is reconstructing the particle tracks. In order to test it we used **made-up data** as measurements

- First we define a random value of $\phi_{real} \in \left[0, \frac{\pi}{4}\right]$
- Then we propagate 10 measurements at the concentric detectors at angles $\phi_{real} \pm \Delta\phi$

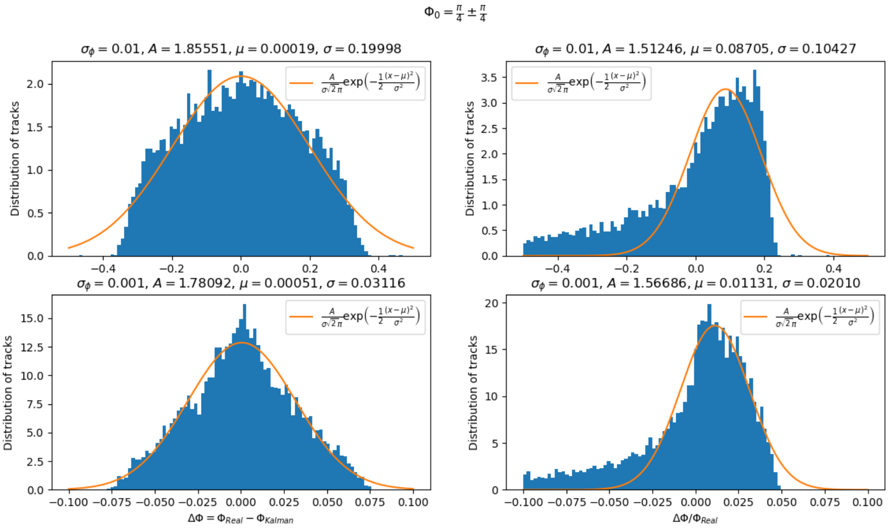
then when we run the Kalman Filter it'll only have access to 10 made-up measurements. Once we have the Kalman data, we can fit a line representing the Kalman prediction of our track, which will have an angle ϕ_{Kalman} with the x-axis. Then we compare our prediction ϕ_{Kalman} with the ϕ_{real} in order to analyze the precision of our Filter.

Kalman Filter for 2D particle travelling through 10 concentric detectors

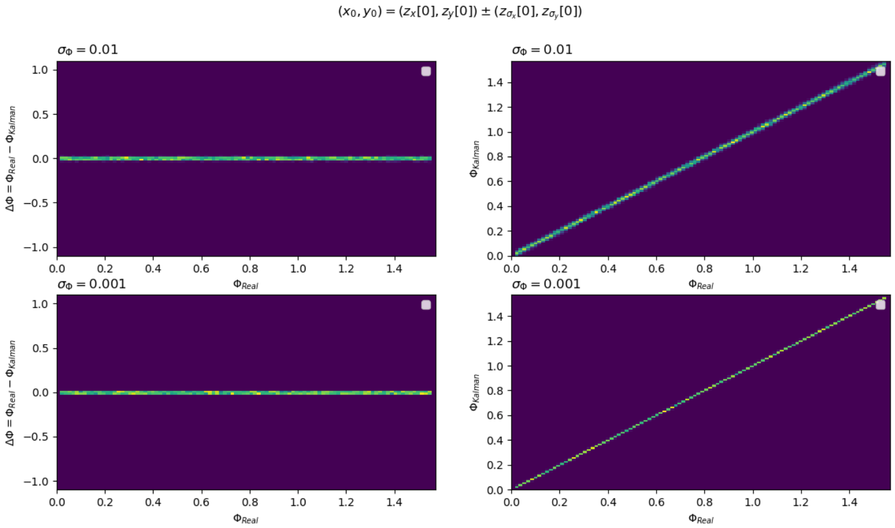
$\Phi_0 = \Phi_{z[0]} \pm \sigma_{\Phi_{z[0]}}$



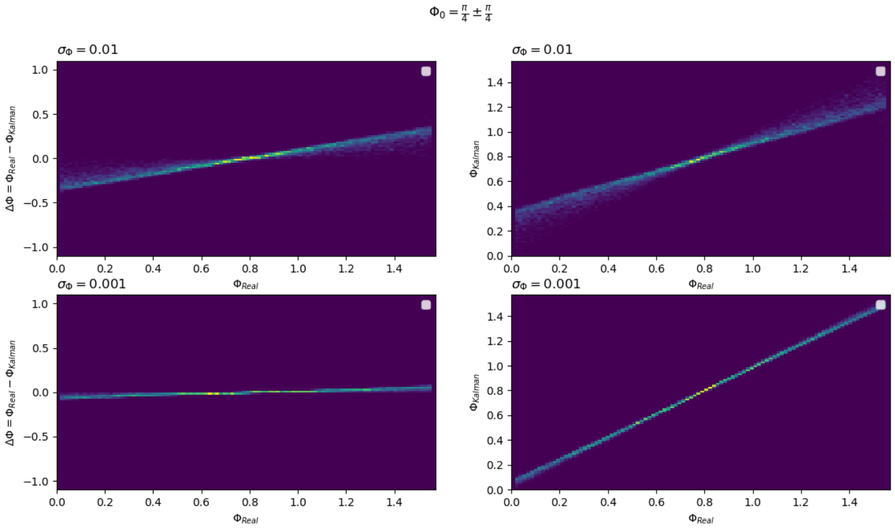
Kalman Filter for 2D particle travelling through 10 concentric detectors



Kalman Filter for 2D particle travelling through 10 concentric detectors



Kalman Filter for 2D particle travelling through 10 concentric detectors



Next steps

Our interest studying this problem is to better understand how the Track Reconstruction is made at the High Level Trigger. Some of the next steps planned are

- ☐ Implement the dynamics of a charge travelling in a magnetic field region as our propagation model
- ☐ Construct the fitting of a circle which intersects the origin
- ☐ Implement the combinatorial track filtering