The Unscented Kalman Filter (UKF)

Author
Giuseppe Coppola. mat: 263624

Università della Calabria, Department of Ingegneria Informatica, Modellistica, Elettronica e Sistemistica (DIMES)

Introduction

- It is used for estimating the state of *non-linear systems*
- It is a *derivative-free* alternative:
  1. Overcomes the limitation by using a deterministic sampling approach
  2. State distribution is approximated using a minimal set of choosen **sigma points** (capturing **mean** and **covariance** of the state)
  3. Mean and covariance are propagated through a non-linear transformation
- It consists of 2 main steps: **model prediction** and **data assimilation**

The Unscented Transformation (UT)

UT is a method for calculating the statistics of a random variable which undergoes a non-linear transformation.

$$y = f(x) + w$$

The objective is to calculate mean and covariance of the transformation.

ASSUMPTIONS: x and w are uncorrelated and x or w need not be Gaussian.

Non-Augmented UKF

- Noises are not included in the state.
- Random vector x is approximated by $2n+1$ symmetric sigma points

---

Algorithm 1 Unscented Kalman Filter (Non-Augmented Case)

Initialize with:

$\hat{x}_0 = E[x_0] = \mu_0$

$P_0 = E[(x_0 - \mu_0)(x_0 - \mu_0)^T]$

for $k = 1$ to ∞ **do**

*Calculate **Sigma Points**:*

$\chi_{k-1} = \left[\hat{x}_{k-1} \quad \hat{x}_{k-1} + \sqrt{(n+\lambda)P_{k-1}} \quad \hat{x}_{k-1} - \sqrt{(n+\lambda)P_{k-1}}\right]$

***Time Update**:*

$\chi_{k|k-1} = f(\chi_{k-1}, u_{k-1})$

$\hat{x}_{\bar{k}} = \sum_{i=0}^{2n} W_i^{(m)} \chi_{i,k|k-1}$

$P_{\bar{k}} = \sum_{i=0}^{2n)} = W_i^{(c)} \left(\chi_{i,k|k-1} - \hat{x}_{\bar{k}}\right)\left(\chi_{i,k|k-1} - \hat{x}_{\bar{k}}\right)^T + Q$

$\gamma_{k|k-1} = h(\chi_{k|k-1})$

$\hat{y}_{\bar{k}} = \sum_{i=0}^{2n} W_i^{(m)} \gamma_{i,k|k-1}$

***Measurement update equation**:*

$P_{y_k y_k} = \sum_{i=0}^{2n} W_i^{(c)} \left(\gamma_{i,k|k-1} - \hat{y}_{\bar{k}}\right)\left(\gamma_{i,k|k-1} - \hat{y}_{\bar{k}}\right)^T + V$

$P_{x_k y_k} = \sum_{i=0}^{2n} W_i^{(c)} \left(\chi_{i,k|k-1} - \hat{x}_{\bar{k}}\right)\left(\gamma_{i,k|k-1} - \hat{y}_{\bar{k}}\right)^T$

$K_k = P_{x_k y_k} P_{y_k y_k}^{-1}$

$\hat{x}_k = \hat{x}_{\bar{k}} + K_k(y_k - \hat{y}_{\bar{k}})$

$P_k = P_{\bar{k}} - K_k P_{y_{\bar{k}} y_{\bar{k}}} K_k^T$

end for

- **Mean** is the weighted average of transformed points
- **Covariance** is the weighted outer product of the trasformed points plus the process noise covariance
- There are 2 types of **weights**:
  1. Mean Weights:

$$W_0^{(m)} = \lambda/(n+\lambda)$$
$$W_i^{(m)} = 1/[2(n+\lambda)]$$
$$i = 1, \ldots, 2n$$

  2. Covariance Weights:

$$W_0^{(c)} = \lambda/(n+\lambda) + (1 - \alpha^2 + \beta)$$
$$W_i^{(c)} = W_i^{(m)}$$
$$i = 1, \ldots, 2n$$

Augmented UKF

- The state is extended to explicity include noise
- Random vector x is approximated by $2(n+m)+1$ symmetric sigma points

Algorithm 2 Unscented Kalman Filter (Augmented Case)

Initialize with:

$\hat{x}_0 = E[x_0] = \mu_0$

$P_0 = E[(x_0 - \mu_0)(x_0 - \mu_0)^T]$

$\hat{x}_0^a = [\hat{x}_0^T \ 0 \ 0]^T$

$P_0^a = E[(x_0^a - \hat{x}_0^a)(x_0^a - \hat{x}_0^a)^T] = \begin{bmatrix} P_0 & 0 & 0 \\ 0 & Q & 0 \\ 0 & 0 & V \end{bmatrix}$

for $k = 1$ to ∞ **do**

*Calculate **Sigma Points**:*

$\chi_{k-1}^a = \left[\hat{x}_{k-1}^a \quad \hat{x}_{k-1}^a + \sqrt{(n+m+\lambda)P_{k-1}^a} \quad \hat{x}_{k-1}^a - \sqrt{(n+m+\lambda)P_{k-1}^a}\right]$

Define:

$\chi^a = \left[(\chi^x)^T \ (\chi^w)^T \ (\chi^v)^T\right]$

Time Update:

$\chi_{k|k-1}^x = f(\chi_{k-1}^x, u_{k-1}, \chi_{k-1}^w)$

$\hat{x}_{\bar{k}} = \sum_{i=0}^{2(n+m)} W_i^{(m)} \chi_{i,k|k-1}^x$

$P_{\bar{k}} = \sum_{i=0}^{2(n+m)} = W_i^{(c)} \left(\chi_{i,k|k-1}^x - \hat{x}_k\right) \left(\chi_{i,k|k-1}^x - \hat{x}_k\right)^T$

$\gamma_{k|k-1} = h(\chi_{k|k-1}^x, \chi_{k-1}^w)$

$\hat{y}_{\bar{k}} = \sum_{i=0}^{2(n+m)} W_i^{(m)} \gamma_{i,k|k-1}$

Measurement update equation:

$P_{y_{\bar{k}} y_{\bar{k}}} = \sum_{i=0}^{2(n+m)} W_i^{(c)} \left(\gamma_{i,k|k-1} - \hat{y}_{\bar{k}}\right) \left(\gamma_{i,k|k-1} - \hat{y}_{\bar{k}}\right)^T$

$P_{x_k y_k} = \sum_{i=0}^{2(n+m)} W_i^{(c)} \left(\chi_{i,k|k-1} - \hat{x}_{\bar{k}}\right) \left(\gamma_{i,k|k-1} - \hat{y}_{\bar{k}}\right)^T$

$K_k = P_{x_k y_k} P_{y_{\bar{k}} y_{\bar{k}}}^{-1}$

$\hat{x}_k = \hat{x}_{\bar{k}} + K_k(y_k - \hat{y}_{\bar{k}})$

$P_k = P_{\bar{k}} - K_k P_{y_{\bar{k}} y_{\bar{k}}} K_k^T$

end for

- **Mean** and **Covariance** are calculated as in the case of Non-Augmented UKF
- Higher computational complexity (more sigma points need to be calculated)
- There are, also in this case, *Mean Weights* and *Covariance Weights*.
- The scaling parameter λ:

$$\lambda = \alpha^2(L + \kappa) - L$$

where:

1. α determine the spread of sigma point around the mean value ($1e-3 \leq \alpha \leq 1$)
2. κ is a secondary scaling parameter ($\kappa = 0$)
3. β is used to incorporate prior knowledge of the distribution of x ($\beta = 2$, for assumption of Gaussian)

Square-Root UKF

- More efficient version of the standard UKF.
- It propagates directly the square root matrix S ($P = SS^T \in \mathbb{R}^{n \times n}$).

Algorithm 3 Square-Root Unscented Kalman Filter

Initialize with:

$\hat{x}_0 = E[x_0] = \mu_0$

$S_0 = chol\left\{ E[(x_0 - \mu_0)(x_0 - \mu_0)^T] \right\}$

for $k = 1$ to ∞ **do**

*Calculate **Sigma Points**:*

$\chi_{k-1} = \left[\hat{x}_{k-1} \quad \hat{x}_{k-1} + \sqrt{(n+\lambda)} S_k \quad \hat{x}_{k-1} - \sqrt{(n+\lambda)} S_k \right]$

***Time Update**:*

$\chi_{k|k-1} = f(\chi_{k-1}, u_{k-1})$

$\hat{x}_{\bar{k}} = \sum_{i=0}^{2n} W_i^{(m)} \chi_{i,k|k-1}$

$S_{\bar{k}} = qr\left\{ \left[\sqrt{W_1^{(c)}} \left(\chi_{1:2n,k|k-1} - \hat{x}_{\bar{k}} \right) \quad \sqrt{Q} \right] \right\}$

$S_{\bar{k}} = cholupdate\left\{ S_{\bar{k}} \ , \ \chi_{0,k} - \hat{x}_{\bar{k}} \ , \ W_0^{(c)} \right\}$

$\gamma_{k|k-1} = h(\chi_{k|k-1})$

$\hat{y}_{\bar{k}} = \sum_{i=0}^{2n} W_i^{(m)} \gamma_{i,k|k-1}$

***Measurement update equation**:*

$S_{\tilde{y}_k} = qr\left\{ \left[\sqrt{W_1^{(c)}} \left(\gamma_{1:2n,k} - \hat{y}_k \right) \quad \sqrt{V} \right] \right\}$

$S_{\tilde{y}_k} = cholupdate\left\{ S_{\tilde{y}_k} \ , \ \gamma_{0,k} - \hat{y}_k \ , \ W_0^{(c)} \right\}$

$P_{x_k y_k} = \sum_{i=0}^{2n} W_i^{(c)} \left(\chi_{i,k|k-1} - \hat{x}_{\bar{k}} \right) \left(\gamma_{i,k|k-1} - \hat{y}_{\bar{k}} \right)^T$

$K_k = \left(P_{x_k y_k} / S_{\tilde{y}_k}^T \right) / S_{\tilde{y}_k}$

$\hat{x}_k = \hat{x}_{\bar{k}} + K_k (y_k - \hat{y}_{\bar{k}})$

$U = K_k S_{\tilde{y}_k}$

$S_k = cholupdate\{ S_{\bar{k}} \ , \ U \ , \ -1 \}$

end for

This algorithm uses 3 linear algebra techniques:

1. **QR Decomposition** ($O(N \cdot L^2)$):
   Given $A \in \mathbb{R}^{N \times L}$ ($N \geq L$)
   $$A = QR$$
   where $Q \in \mathbb{R}^{N \times N}$ (orthogonal) and $R \in \mathbb{R}^{N \times L}$ (upper triangular).
   Given $\bar{R}$ as the upper triangular part of R:
   $$\bar{R}^T \bar{R} = SS^T$$
2. **Cholesky Factor Updating** ($O(L^2)$ per update):
   It is used to update the matrix S so as to include the remaining part of the covariance;
   It's a rank-1 update $S \pm \sqrt{\nu} u u^T$
   $$S = cholupdate\{S, u, \pm\sqrt{\nu}\}$$
3. **Efficient Least Squares**:
   Instead of calculating $(AA^T)x = A^T b$, it solves
   $$Ax = b$$
   where $A = S^T$ and $b = P$

Simulations

Model (2D Motion)

$$\begin{bmatrix} x_{k+1}^{(1)} \\ x_{k+1}^{(2)} \\ x_{k+1}^{(3)} \\ x_{k+1}^{(4)} \end{bmatrix} = \begin{bmatrix} x_k^{(1)} + x_k^{(3)} \cdot \Delta t + 0.1 \cdot \sin(x_k^{(2)}) \\ x_k^{(2)} + x_k^{(4)} \cdot \Delta t \cdot \cos(x_k^{(1)}) \\ x_k^{(3)} \\ x_k^{(4)} \end{bmatrix}$$

The dynamic is affected by a **process Gaussian noise** with 0 mean and covariance Q.

Observational Model

$$h(x) = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

They are affected by a **measurement Gaussian noise** with 0 mean and covariance V.

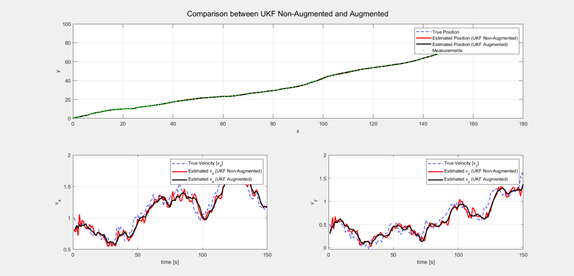
Accurate Sensors

Covariance Matrices

$$Q = \begin{bmatrix} 0.01 & 0 & 0 & 0 \\ 0 & 0.01 & 0 & 0 \\ 0 & 0 & 0.005 & 0 \\ 0 & 0 & 0 & 0.005 \end{bmatrix} \qquad V = \begin{bmatrix} 0.05 & 0 \\ 0 & 0.05 \end{bmatrix}$$

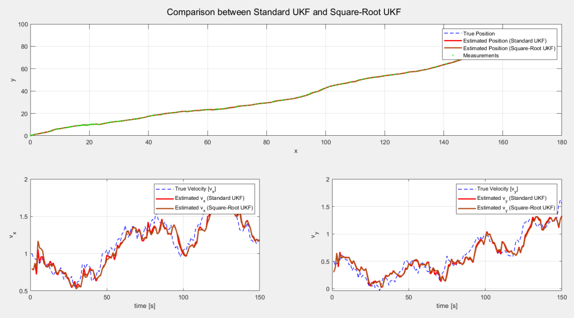
- Positions and velocities are expected to evolve with little pertubation
- Sensors that measure position are precise (with little random deviation in observation)

Accurate Sensors



- In both cases the position estimate is practically identical to the real trajectory.
- In **Non-Augmented UKF**, velocity estimates show small oscillations (especially at the "curvature"points).
- In **Augmented UKF** velocity estimates are smoother (because of the incorporation of process noise into the state vector) than those from UKF-NA.

Accurate Sensors



- In both cases the position estimate is practically identical to the real trajectory.
- **Square-Root UKF** shows slightly smoother estimates with fewer deviations (it absorbs more noise).
- **Square-Root UKF** is more numerically stable due to Cholesky decomposition.

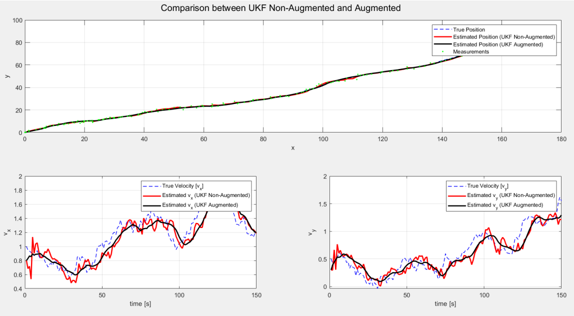
Not Accurate Sensors

Covariance Matrices

$$Q = \begin{bmatrix} 0.01 & 0 & 0 & 0 \\ 0 & 0.01 & 0 & 0 \\ 0 & 0 & 0.005 & 0 \\ 0 & 0 & 0 & 0.005 \end{bmatrix} \qquad V = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$$

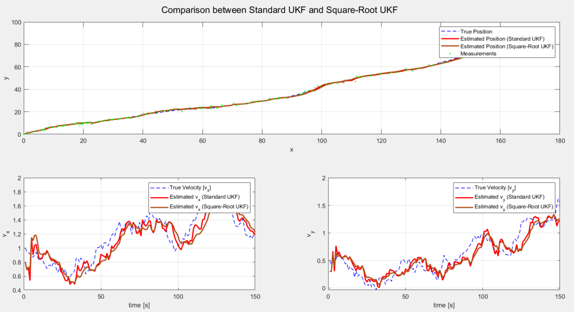
- Positions and velocities are expected to evolve fairly predictably.
- Sensors that measure position are much noisier (there is a significant uncertainty).

Not Accurate Sensors



- In both cases, it has a good estimate in terms of position.
- In **Augmented UKF**, velocity estimates is better than velocity estimates in **Non-Augmented UKF**
- **Augmented UKF** has the best numerical stability but it calculates more sigma points (more computational cost).

Not Accurate Sensors



- **Square-Root UKF** estimates are smoother.
- At peaks and valley there are slight deviations (because of measurements noise and system non-linearity).
- **Square-Root UKF** is more numerically stable.

Stress Test

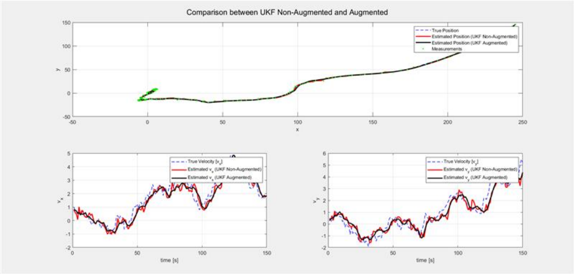
Covariance Matrices

$$Q = \begin{bmatrix} 0.2 & 0 & 0 & 0 \\ 0 & 0.2 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0.1 \end{bmatrix} \qquad V = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

- The dynamic is highly uncertain
- Sensors that measure position are extremely noisy

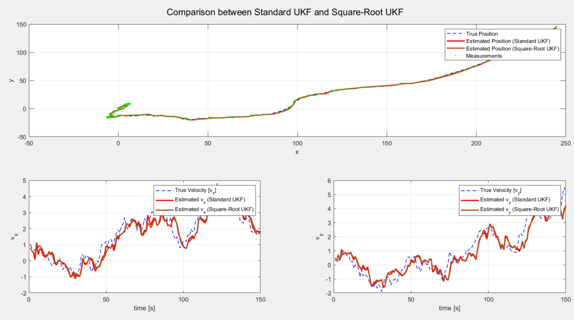
This scenario is used for test scopes !

Stress Test



- **Non-Augmented UKF** handles noise inefficiently (does not account for process noise directly in the sigma points).
- **Augmented UKF** tracks position very well even under stress.
- In **Augmented UKF** there are some visible deviation in velocity estimates, especially in rapid-change zones.

Stress Test



- Both filters track the position identically.
- **Square-Root UKF** tracks velocity very well (showing smoother tracking curves).
- **Square-Root UKF** is more numerically stable.

Conclusions

- The difference between **Non-Augmented UKF** and **Augmented UKF** is how noise is managed.
- **Augmented UKF** (with noise inclusion in the state) produces more accurate estimates but with more computational complexity.
- With low noise, both **A-UKF** and **SR-UKF** offering smoother and more stable estimates.
- With noisy sensors, **NA-UKF** is outperformed by other versions thanks to better noise modeling and numerical stability.
- Under stress conditions, the best is **SR-UKF** because showed robustness and best noise suppression (efficient matrix handling).