Cubature-based Kalman Filters for Positioning

H. Pesonen, Tampere University of Technology
R. Piché, Tampere University of Technology

Abstract—We review a family of nonlinear filtering methods that includes unscented filters and cubature Kalman filters. These methods approximate the integrals occurring in the Bayesian formulation of the filtering problem by a sum of weighted integrand evaluations calculated at prescribed nodes. In addition to methods from the literature we introduce a new spherical-radial integration rule based filter. The filters are compared using an extensive set of positioning benchmarks including real and simulated data from GPS and mobile phone base stations. It is found that in tested scenarios no particular filter in this family is clearly superior.

Index Terms—Bayesian filtering, unscented filtering, Gaussian filters, cubature filters

I. INTRODUCTION

The problem of determining a position from a time series of measurements is often formulated as a filtering problem, where the position is estimated recursively by updating the estimate of the past position with the newest available measurements. In positioning, the measurements are often related nonlinearly to the unknown position, so to obtain an estimate of the position we have to solve a nonlinear filtering problem [6]. A theoretical solution to the nonlinear filtering problem can be obtained in the Bayesian filtering framework [3], which finds the posterior distribution of the position coordinates, and possibly other variables such as velocity, given all the available data. However, in practice the posterior distribution and its summary statistics (mean, covariance, etc.) have to be solved using approximations.

One general approach to approximating the Bayesian solution is the following, which we call cubature-based Kalman filtering. At each time step, the true posterior distribution is approximated with a Gaussian distribution and its summary statistics (mean, covariance) as the posterior. These moments are found in the Bayesian filtering framework [3], which is a vector of measurements at time \( t_k \) of the state given all the available data in two steps. The basic nonlinear discrete-time state-space model we are considering for the positioning problem is

\[
\begin{align*}
\mathbf{x}_{k+1} &= F_k \mathbf{x}_k + \mathbf{w}_k \\
\mathbf{y}_k &= \mathbf{h}_k(\mathbf{x}_k) + \mathbf{v}_k, 
\end{align*}
\]

where \( \mathbf{x}_k \in \mathbb{R}^{n_x} \) is the state of the system and \( \mathbf{y}_k \in \mathbb{R}^{n_y} \) is a vector of measurements at time \( t_k \). The linear state-transition matrix \( F_k \) describes the evolution of the state (e.g. position, velocity, clock bias, clock drift, etc.) in time and \( \mathbf{h}_k(\cdot) \) links the state and the observations. The random noises \( \mathbf{w}_k \) and \( \mathbf{v}_k \) are assumed white, mutually independent and independent of the initial state \( \mathbf{x}_0 \).

In addition, they are often modeled as Gaussian, an assumption that we are also making in this work, and

\[
\begin{align*}
\mathbb{E}[\mathbf{w}_k\mathbf{w}_k^T] &= Q_k, \\
\mathbb{E}[\mathbf{v}_k\mathbf{v}_k^T] &= R_k, \\
\mathbf{x}_0 &\sim \mathcal{N}(\bar{\mathbf{x}}_0, P_0|0). 
\end{align*}
\]

In Bayesian filtering we solve the posterior distribution of the state given all the available data in two steps. The first step (prediction) is

\[
p(\mathbf{x}_k | \mathbf{y}_{1:k-1}) = \int p(\mathbf{x}_k | \mathbf{x}_{k-1}) p(\mathbf{x}_{k-1} | \mathbf{y}_{1:k-1}) \, d\mathbf{x}_{k-1}
\]
and the second step (update) is

\[
p(x_k|y_{1:k}) = \frac{p(y_k|x_k)p(x_{k-1}|y_{1:k-1})}{\int p(y_k|x_k)p(x_{k-1}|y_{1:k-1})dx_k},
\]

(3)

where we have used the notation \(y_{1:n} = \{y_1, \ldots, y_n\}\).

If we approximate the posterior as a Gaussian so that

\[
p(x_{k-1}|y_{1:k-1}) = \mathcal{N}_{P_{x|k-1}}(x_{k-1}),
\]

where \(\mathcal{N}\) is the normal density with mean \(\mu\) and covariance \(\Sigma\), and we have the system (1), then the predictive distribution is also a Gaussian with mean

\[
E[x_k|y_{1:k-1}] = E[F_{k-1}x_{k-1} + w_{k-1}|y_{1:k-1}]
\]

and covariance

\[
E[(x_k - \bar{x}_{k-1})(x_k - \bar{x}_{k-1})^T|y_{1:k-1}]
\]

\(= F_{k-1}E[(x_{k-1} - \bar{x}_{k-1})(x_{k-1} - \bar{x}_{k-1})^T|y_{1:k-1}]F_{k-1}^T
\]

\(+ E[w_{k-1}w_{k-1}^T|y_{1:k-1}]
\]

\(= F_{k-1}P_{k-1}|k-1|F_{k-1}^T + Q_{k-1} = P_{k}|k-1|
\)

(4)

The distribution of the predicted measurement is also approximated as a Gaussian with mean

\[
E[y_k|y_{1:k-1}] = E[h_k(x_k) + v_k|y_{1:k-1}]
\]

\(= \int h_k(x_k)p(x_k|y_{1:k-1})dx_k = \bar{y}_{k-1}
\)

(6)

and covariance

\[
E[(y_k - \bar{y}_{k-1})(y_k - \bar{y}_{k-1})^T|y_{1:k-1}]
\]

\(= \int h_k(x_k)h_k(x_k)^Tp(x_k|y_{1:k-1})dx_k - \bar{y}_{k-1}\bar{y}_{k-1}^T + R_k = P_{k}|k-1|
\)

(7)

For computing the posterior distribution, we need the crosscovariance

\[
E[(x_k - \bar{x}_{k-1})(y_k - \bar{y}_{k-1})^T|y_{1:k-1}]
\]

\(= \int x_kh_k(x_k)^Tp(x_k|y_{1:k-1})dx_k - \bar{x}_{k-1}\bar{y}_{k-1}^T
\]

\(= P_{k}|k-1|
\)

and covariance

\[
P_{k}|k = P_{k}|k-1 - K_kP_{k}|k-1K_k^T,
\]

(10)

where

\[
K_k = P_{k}|k-1(P_{k}|k-1)^{-1}
\]

and

\[
P_{k}|k = P_{k}|k-1
\]

The cubature rules for the computation of the integrals (6) – (8) are discussed in the next section.

III. Cubature rules

As the computation of means and covariances are carried out componentwise, the integrals discussed in the last section are all of the form

\[
\int g(x)\mathcal{N}_{\Sigma}(x)dx = \frac{1}{\sqrt{\pi^n}} \int g(\sqrt{2\Sigma}x + \mu)e^{-x^T\Sigma x}dx,
\]

(12)

i.e. integral of a nonlinear function multiplied by a Gaussian weight. Next we discuss two different numerical integration approaches to approximately compute (12) as

\[
I(f) = \int f(x)w(x)dx \approx \sum_{i=1}^{n} w_i f(x_i),
\]

(13)

where \(w(\cdot)\) is a standard normal density. The unscented transformation [7] can also be interpreted as an approximate integral of the form (13); we omit the details.

A. Gauss-Hermite quadrature

Gauss-Hermite quadrature is a Gaussian quadrature over \((-\infty, \infty)\) with an exponential weight function \(e^{-x^2}\). An m-point Gauss-Hermite quadrature rule

\[
\int_{-\infty}^{\infty} P(x)e^{-x^2}dx \approx \sum_{i=1}^{m} \omega_i P(\xi_i)
\]

(14)

is exact for all polynomials \(P(x)\) up to a degree \(2m - 1\). To compute the \(n\)-dimensional integral (12), the one-dimensional quadrature rule has to be applied as a product rule

\[
\int \mathbb{R}^n g(x_1, \ldots, x_n)e^{-x^T x}dx
\]

\(\approx \sum_{i_1=1}^{m} \cdots \sum_{i_n=1}^{m} g(\xi_{i_1}, \ldots, \xi_{i_n})\omega_{i_1} \cdots \omega_{i_n},
\]

(15)

which is exact for all monomials

\[
P(x) = x_1^{i_1} \cdots x_n^{i_n},
\]

(16)

where \(0 \leq i_j \leq 2m - 1\). A problem with product rules is that the computational complexity quickly becomes enormous as the number of dimensions grows, because the product rules requires \(m^n\) function evaluations. For example in satellite positioning the dimension of the state can often be 8 or more, and a three-point quadrature rule needs \(3^8 = 6561\) or more function evaluations. For two-point Gauss-Hermite formula the nodes and weights are \(\xi_{1,2} = \pm \sqrt{2}\) and \(\omega_{1,2} = \frac{\sqrt{2}}{2}\), and the number of function evaluations required is \(2^5 = 256\) or more, which is still excessive.
B. Spherical-radial cubature

The spherical-radial cubature rule for integral (12) is based on the spherical-radial transformation [1, 8]. Denoting \( x = r y \), where \( y^T y = 1 \) and \( r \in [0, \infty) \), we can write (12) as

\[
I(f) = \int_0^\infty \int_{U_n} f(r y)r^{n-1}e^{-r^2} d\sigma(y) dr,
\]

where \( U_n \) is the surface of an \( n \)-dimensional unit sphere and \( \sigma(\cdot) \) is the spherical surface measure. The integral is split into spherical integral

\[
S(r) = \int_{U_n} f(r y) d\sigma(y)
\]

and radial integral

\[
R = \int_0^\infty S(r)r^{n-1}e^{-r^2} dr.
\]

A fully symmetric spherical cubature rule

\[
S(r) \approx \sum_{i=1}^{2n} b_i f(r s_i)
\]

is described in [1], for which \( 2n \) nodes \( s_i \) are located at the intersection of \( n \)-dimensional unit sphere and its axes. The nodes have equal weights \( b_i = \frac{2\sqrt{n\pi}}{\Gamma(n/2)} \), where \( \Gamma(\cdot) \) is the gamma function.

The radial integral can be transformed as

\[
R = \frac{1}{2} \int_0^\infty S(\sqrt{r})r^{\frac{n}{2}-1}e^{-r} dr
\]

which can be approximated using generalized Gauss-Laguerre quadrature [4]

\[
R \approx \sum_{j=1}^{m} \frac{1}{2} a_j S(\sqrt{r}),
\]

where \( r_j \) are the zeros of the generalized Laguerre polynomial \( L_m^{(\frac{n}{2})-1}(r) \) and the weights are

\[
a_j = \frac{\Gamma(m + \frac{n}{2} - 1) r_j}{m!(m + \frac{n}{2} - 1)(L_m^{(\frac{n}{2})-1}(r_j))^2}
\]

Arasaratnam and Haykin [1] use only the one-point formula \( m = 1 \). We employ also the two-point formula where the nodes are the roots of the second order generalized Laguerre polynomial \( m = 2 \) and weights can be found from the moments equations as the rule has to be exact for certain degree polynomials.

The spherical-radial rule can be obtained from (20) and (22) as a product rule [1]

\[
I(f) \approx \sum_{i=1}^{2n} \sum_{j=1}^{m} \frac{a_j}{2} b_i f(r_j s_i).
\]

The total number of function evaluations required to compute an integral is then \( 2nm \). GPS positioning \( (n = 8) \) thus requires 16 or 32 function evaluations, when using discussed cubature rules. The nodes and weights for \( m = 1, 2 \) are listed in Table I.

IV. Cubature based Gaussian filters

The cubature based Gaussian filter algorithms use cubature rules of the form

\[
I(f) \approx \sum_{i=1}^{m} \omega_i f(\xi_i)
\]

to approximate the integrals in (6) – (8). We assume that

\[
p(x_{k-1}|y_{1:k-1}) \approx \mathcal{N}_{F_k|y_{1:k-1}}(x_{k-1}),
\]

Then the predictive distribution is

\[
p(x_k|y_{1:k-1}) \approx \mathcal{N}_{F_k|y_{1:k-1}}(x_k),
\]

where

\[
\bar{x}_{k|k-1} = F_k|y_{1:k-1}\bar{x}_{k-1|k-1}
\]

\[
P_k|y_{1:k-1} = F_k|y_{1:k-1}P_{k-1|y_{1:k-1}}F_k|y_{1:k-1} + Q_k.
\]

The cubature nodes are transformed according to the predictive distribution

\[
\zeta_i = \sqrt{P_{k|y_{1:k-1}}}\xi_i + \bar{x}_{k|k-1},
\]

where \( P_k = \sqrt{P_{k|y_{1:k-1}}} \). We approximate

\[
\bar{y}_{k|k-1} \approx \sum_{i=1}^{m} \omega_i h_k(\zeta_i)
\]

\[
P_{yy_k|y_{1:k-1}} \approx \sum_{i=1}^{m} \omega_i h_k(\zeta_i)h_k(\zeta_i)^T - \bar{y}_{k|y_{1:k-1}}\bar{y}_{k|y_{1:k-1}}^T
\]

and use the approximations to compute the Kalman gain

\[
K_k = P_{yy_k|y_{1:k-1}}(P_{yy_k|y_{1:k-1}})^T
\]

and posterior mean and covariance

\[
\bar{x}_{k|k} = \bar{x}_{k|k-1} + K_k(y_k - \bar{y}_{k|k-1})
\]

\[
P_k = P_{k|y_{1:k-1}} - K_kP_{yy_k|y_{1:k-1}}K_k^T.
\]

Notice that EKF can also be interpreted as a cubature-based filter as it first approximates the integrand with Taylor series and then integrates it exactly. In some sense EKF uses one node \( x_i = 0 \) with a weight \( \omega_i = 1 \) and is exact for first degree polynomials.
V. TESTS

We test the filters in two positioning scenarios. In the first scenario we use real GPS pseudo and delta range measurements along with simulated range measurements from mobile phone network. As the distance between base stations and the receiver is much shorter than the distance to satellites, the mobile network observations are locally more nonlinear than the pseudo range measurements. In the second scenario, we use only data from the mobile phone network. The tests are carried out using Personal Navigation Filter Framework (PNaFF) [9]. We test unscented Kalman filter (UKF), extended Kalman filter (EKF), cubature Kalman filter (CKF) [1], Gauss-Hermite quadrature filter with two-point quadrature (GHQF) and cubature Kalman filter with two-point radial quadrature (CKF2). The points used by different filters to evaluate the integrands in two-dimensional case are illustrated in Figure 1, the number of nodes employed by different cubature filters are listed in Table I, where nodes are given as generators, with associated weights in parenthesis. Generators are a shorthand notation for a set of symmetric points. Generator \([a_1, a_2, \ldots, a_r] \in \mathbb{R}^n\) represents all the nodes that can be obtained from the node \((a_1, \ldots, a_r, 0, \ldots 0) \in \mathbb{R}^n\) with permutations and sign changes.

![Figure 1. Nodes of cubature rules in two-dimensional problem](image)

**TABLE I**  
NUMBER OF REQUIRED NODES, GENERATORS AND THE ASSOCIATED WEIGHTS USED BY DIFFERENT CUBATURE FILTERS.

<table>
<thead>
<tr>
<th>Filter (# nodes)</th>
<th>(<a href="w_i">a_i</a>)</th>
<th>(<a href="w_i">a_i, \ldots, a_i</a>)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EKF (1)</td>
<td>([1])</td>
<td>([1]a_1)</td>
</tr>
<tr>
<td>UKF (2n + 1)</td>
<td>([w_1])</td>
<td>(<a href="w_2">a_2, \ldots, a_r</a>)</td>
</tr>
<tr>
<td>CKF (2n)</td>
<td>([w_3])</td>
<td>(<a href="w_4">a_4, \ldots, a_r</a>)</td>
</tr>
<tr>
<td>CKF2 (4n)</td>
<td>([w_5])</td>
<td>(<a href="w_6">a_6, \ldots, a_r</a>)</td>
</tr>
<tr>
<td>GHQF (2n)</td>
<td>(<a href="w_6">a_i, \ldots, a_r</a>)</td>
<td></td>
</tr>
</tbody>
</table>

The first test scenario is a hybrid positioning application, in which we use real GPS pseudorange data along with simulated range measurements from mobile phone base stations. The state consists of position, velocity and receiver clock bias and drift, so that the dimension of the state \(n = 8\). The filters were tested with 41 tracks consisting of 42576 time steps in total. The results are listed in Table III. As would be assumed, due to the locally almost linear nature of pseudorange observations, EKF performs exceptionally well and its performance is hard to improve on. No filter performs better than EKF, CKF coming close in terms of accuracy but is naturally computationally more expensive. UKF and CKF2 suffer from the high-dimensionality of the problem, and run into some numerical problems, made much worse by ‘too accurate’ measurements, therefore diverging in a few test cases causing the average performance to be poor. Very accurate observations cause the likelihood function to be concentrated in a very small region of the space, and depending where the cubature nodes evaluate the function, severe numerical problem can occur. GHQF is more stable, but is computationally very expensive and does not compete with EKF in terms of accuracy.

**TABLE II**  
NUMERICAL VALUES RELATED TO THE CUBATURE NODES.  
\(n = \) DIMENSION, \(\kappa = n - 3\) \(\) \(=\) PARAMETER OF UKF, \(\alpha = \frac{n}{2} - 1\)

<table>
<thead>
<tr>
<th>Filter</th>
<th>CPU time (m)</th>
<th>mean err (%)</th>
<th>err &lt; 50 m (%)</th>
<th>err &lt; 150 m (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EKF</td>
<td>1</td>
<td>31</td>
<td>89</td>
<td>98</td>
</tr>
<tr>
<td>UKF</td>
<td>3</td>
<td>221</td>
<td>80</td>
<td>91</td>
</tr>
<tr>
<td>CKF</td>
<td>3</td>
<td>31</td>
<td>89</td>
<td>98</td>
</tr>
<tr>
<td>CKF2</td>
<td>5</td>
<td>345</td>
<td>69</td>
<td>85</td>
</tr>
<tr>
<td>GHQF</td>
<td>37</td>
<td>40</td>
<td>90</td>
<td>98</td>
</tr>
</tbody>
</table>

**TABLE III**  
HYBRID NAVIGATION TEST RESULTS WITH REAL GPS DATA.

The second scenario is a positioning application using simulated range measurement data to pseudolites which in this test case are mobile phone base stations. The base stations are located much closer to the receiver than the satellites, causing the range measurements to be more
nonlinear locally than the pseudo range observations. A total of 500 tracks with 120 time steps long each is simulated. The state consists of position and velocity parameters, so that the dimension \( n = 6 \). The results are listed in Table IV. Now all filters have similar accuracy. The mean error is within 5 m for all the filters and the number of estimates having error less than 50 m and 150 m is within 2% for all the filters. The only major difference between the methods is in the required computation time. Not even range measurements to nearby base stations have nonlinearity that would be too much for EKF to perform worse compared to other filters based on Gaussian approximations. Furthermore, it can be noted that the filters did not run into numerical problems as they did with satellite data, due to the lower dimensionality of the problem and the larger noisiness of measurements.

<table>
<thead>
<tr>
<th>Filter</th>
<th>CPU time</th>
<th>mean err</th>
<th>err &lt; 50 m</th>
<th>err &lt; 150 m</th>
</tr>
</thead>
<tbody>
<tr>
<td>EKF</td>
<td>1</td>
<td>45</td>
<td>80</td>
<td>94</td>
</tr>
<tr>
<td>UKF</td>
<td>6</td>
<td>48</td>
<td>79</td>
<td>93</td>
</tr>
<tr>
<td>CKF</td>
<td>6</td>
<td>46</td>
<td>79</td>
<td>94</td>
</tr>
<tr>
<td>CKF2</td>
<td>10</td>
<td>46</td>
<td>79</td>
<td>93</td>
</tr>
<tr>
<td>GHQF</td>
<td>25</td>
<td>43</td>
<td>81</td>
<td>94</td>
</tr>
</tbody>
</table>

**TABLE IV**
Test results with simulated mobile phone range measurements.

VI. CONCLUSIONS

Filters belonging to a family of nonlinear filtering methods that includes unscented filters and cubature Kalman filters were reviewed. The algorithms were presented and tested using real and simulated data. The most notable differences between the filters in tested scenarios were caused by numerical inaccuracies, caused by very accurate pseudo range observations and high dimensionality of the problem. These problems were not present in the tests with simulated data, where the dimension of the problem is lower and range measurements were noisier. In both test scenarios it was found that the performance of EKF is very robust and efficient compared to cubature-based Kalman filters. However, in problems with more severe nonlinearities, such as GPS/INS, cubature-based Kalman filters could be employed and should perform better than EKF.

REFERENCES