A general filter for measurements with any probability distribution

Yoav Rosenberg
Institute of Computer Science
The Hebrew University
Jerusalem Israel 91904
yoavr@cs.huji.ac.il

Michael Werman
Institute of Computer Science
The Hebrew University
Jerusalem Israel 91904
werman@cs.huji.ac.il

Abstract

The Kalman filter is a very efficient optimal filter, however, it has the precondition that the noises of the process and of the measurement are Gaussian. In this paper we introduce 'The General Distribution Filter' which is an optimal filter that can be used even where the distributions are not Gaussian. An efficient practical implementation of the filter is possible where the distributions are discrete and compact or can be approximated as such.

1. Definition of the problem

The notion of filtering is connected with that of a **process**. The process **state** at time t is described by a vector, which is unknown and must be computed. An example of such a process is a moving vehicle, where the state is the vehicle's position and speed. The information about the process comes from **measurements**, where the connection between the process and the measurement is known. Usually there is noise in the system, so that the measurement is described by its probability distribution.

In order to estimate the process's current state, the last measurement is not enough, because the measurement is not exact and the information that can be extracted from it is limited. If the process's evolution in time, that is, the connection between its past and current states is also known, the estimation of the past states can be used to improve the current one. This combining of information is *filtering*. The optimal filter uses all the data available from past measurements and information about the process's evolution in time to compute the best possible estimation of the process' current state.

The prevalent optimal filter is the Kalman filter which can be used when the measurement noise and the process state transition noise are Gaussian. The main advantage of this filter is its small computation and storage complexity. The problem is that when the measurement is not a Gaussian distribution, the Kalman filter cannot be used. This was the motivation to develop the general distribution filter which can be used for any distribution function. As we will see, although the filter is defined for any probability distribution of the measurement, an efficient computer implementation of this filter is possible when the probability distribution of the measurement is discrete or can be approximated as such.

We will demonstrate a tracking application where the distributions cannot be approximated as Gaussian, and the Kalman filter cannot be used. As a better alternative, the measurement's distribution is represented as a probability matrix, and the filtering is achieved using the general distribution filter.

In the appendix we show that the Kalman filter is a special case of the general distribution filter.

1.1. Definitions

- Given a **process** $\mathcal{P}$. $\mathbf{x}_t$ is a random vector that describes its **state** at a discrete time t. Let the probability distribution function of $\mathbf{x}_t$ be $P_{\mathbf{x}_t}()$.
- The **state transition** from state t to $t+1$ is given by the equation:

$$\mathbf{x}_{t+1} = \Phi_{(t,t+1)}(\mathbf{x}_t) + \mathbf{n}_t$$

where
$\Phi_{(t,t+1)}(\mathbf{x}_t)$ is the transition function. From now on, we will refer to it as just $\Phi(\mathbf{x}_t)$.
$\mathbf{n}_t$ is the process noise with probability distribution $P_{\mathbf{n}_t}()$,
where $\mathbf{n}_t$ is independent of all $\mathbf{n}_j, j < t$.

- $\mathbf{y}_t$ is the **measurement** vector observed at time t. The connection between the measurement and the state is given as:

$$P(\mathbf{y}_t \mid \mathbf{x}_t)$$

1063-6919/97 $10.00 © 1997 IEEE

Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)
1063-6919/97 $10.00 © 1997 IEEE

i.e. the probability of obtaining the observed measurement vector $\mathbf{y}_t$ for any possible $\mathbf{x}_t$. $P(\mathbf{y}_t \mid \mathbf{x}_t)$ should be independent of all $\mathbf{n}_j, j < t$ and of $\mathbf{y}_j, j < t$. $\mathbf{n}_t$ should be independent of $\mathbf{y}_j, j < t$.

1.2. The filter equations

State propagation: Propagation from state *t* to *t+1*: Ignoring the propagation noise, let

$$\mathbf{x}'_{t+1} = \Phi(\mathbf{x}_t) \tag{1}$$

Let $\Phi^{-1}(\mathbf{u})$ be the region of all the vectors that are mapped to $\mathbf{u}$ by Φ:

$$\Phi^{-1}(\mathbf{u}) = \{\mathbf{v} \mid \Phi(\mathbf{v}) = \mathbf{u}\}$$

The probability distribution of $\mathbf{x}'_{t+1}$ is:

$$P_{\mathbf{x}'_{t+1}}(\mathbf{u}) = P_{\mathbf{x}_t}(\Phi^{-1}\mathbf{u}) \tag{2}$$

where $P'_{\mathbf{x}_{t+1}}$ is the state's probability distribution at $t+1$, ignoring the process noise effect.

The process noise random vector is added to $\mathbf{x}'_{t+1}$ to get the process state at *t+1*:

$$\mathbf{x}^-_{t+1} = \mathbf{x}'_{t+1} + \mathbf{n}_t$$

and the probability distribution of this sum is:

$$P^-_{\mathbf{x}_{t+1}} = P_{\mathbf{x}'_{t+1}} \otimes P_{\mathbf{n}_t} \tag{3}$$

Here we use the fact that $\mathbf{x}'_{t+1}$ and $\mathbf{n}_t$ are independent. This is a consequence of the requirement that $\mathbf{n}_t$ is independant of $\mathbf{y}_j, j < t$ and of $\mathbf{n}_j, j < t$

Filtering: In the state propagation stage we computed the apriori process state probability distribution $P^-_{\mathbf{x}_{t+1}}(\mathbf{v})$. The next stage is to take the measurement and use it to update the process state estimation. This will give the aposteriori process state probability distribution.

Applying Bayes theorem gives:

$$P^+_{\mathbf{x}_{t+1}}(\mathbf{u}) = \frac{P(\mathbf{y}_{t+1} \mid \mathbf{x}_{t+1} = \mathbf{u})\, P^-_{\mathbf{x}_{t+1}}(\mathbf{u})}{P(\mathbf{y}_{t+1})} \tag{4}$$

where $\mathbf{P}^-_{\mathbf{x}_{t+1}}$ is the apriori state distribution, and $\mathbf{P}(\mathbf{y}_{t+1})$ is the probability of measuring $\mathbf{y}_{t+1}$:

$$P(\mathbf{y}_{t+1}) = \int_{\mathbf{v}} P(\mathbf{y}_{t+1} \mid \mathbf{x}_{t+1} = \mathbf{v})\, P^-_{\mathbf{x}_{t+1}}(\mathbf{v})\, d\mathbf{v}$$

1.3. The working point

The equations for the filter gives the probability distribution $P_{\mathbf{x}_t}(\mathbf{u})$ of the process state $\mathbf{x}_t$ at time *t*. In many practical cases it is necessary to choose one value from the probability distribution that will be the 'best' representive for $\mathbf{x}_t$. We call this chosen value 'the working point'.

There are several options to define what is the 'best' working point for $\mathbf{x}_t$. Here we describe three methods for choosing the working point. The first two methods are commonly used in estimation theory [1], while the third is new. We will first define the three working points and then discuss the meaning of each working point, choosing the best depending on our needs.

- **The maximum likelihood criterion**
  Choose the point with the highest probability:

  $$\mathbf{x}_{ml} = \{\mathbf{u} \mid P_{\mathbf{x}_t}(\mathbf{u}) = maximum\}$$

  This gives the maximum likelihood choice for $\mathbf{x}_t$.

- **The least squared error criterion**
  Choose the point that minimizes the expected squared error:

  $$\mathbf{x}_{ls} = \{\mathbf{u} \mid Z(\mathbf{x}) = minimum\}$$

  where

  $$Z(\mathbf{x}) = \int_{\mathbf{v}} P(\mathbf{v})\, \|\mathbf{v} - \mathbf{x}\|^2 d\mathbf{v}$$

  $Z(\mathbf{x})$ is the least square error expected, i.e. the error variance.

- **Optimization criteria**
  We define the region S:

  $$S = \{s \mid P_{\mathbf{x}_t}(\mathbf{s})/P_{\mathbf{x}_t}(\mathbf{x}_{ml}) \geq \epsilon\}$$

  where $0 \leq \epsilon \leq 1$.
  The region S is the set of all the points with probability (relative to the best probability) above a threshold ϵ.
  This region represents the probable choices for $\mathbf{x}_t$. The working point is found by imposing an external optimization scheme on this region. In the tracking system (Section 3), we minimize $|s - v|$ which is the distance of s from the vector $\mathbf{v}$, so that:

  $$\mathbf{x}_{opt} = min_{s \in S}\ |\mathbf{s} - \mathbf{v}|$$

  If $\epsilon = 1$ then the region S will be only one point, which is the maximum likelihood point. Therefore $\mathbf{x}_{opt} \rightarrow \mathbf{x}_{ml}$ when $\epsilon \rightarrow 1$. On the other hand, if $\epsilon \rightarrow 0$, then $\mathbf{x}_{opt}$ is more affected by the optimization. When $\epsilon \rightarrow 0$, the valid region is all the space including the origin, and $\mathbf{x}_{opt} \rightarrow \mathbf{v}$.

  The optimization factor described here is the one that fits our needs. In other cases different optimization schemes can be defined and used to choose the working point.

Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)
1063-6919/97 $10.00 © 1997 IEEE

1.4. Comparing the three working points

The maximum likelihood working point $\mathbf{x}_{ml}$ is easy to find, but is vulnerable to noise. When the probability distribution is relatively smooth, the maximum likelihood point is not much larger than the other points, and any small noise can cause the $\mathbf{x}_{ml}$ location to change. So even if the real process state is constant over time, we can get a totally different value of $\mathbf{x}_{ml}$ for every $\mathbf{t}$.

The least squared error working point $\mathbf{x}_{ls}$ tends to be more stable, since it depends on the whole distribution rather than just a specific point like $\mathbf{x}_{ml}$. For this reason the error influence tends to be smoothed. The problem is that this point usually does not give the true value of the process, and it is sensitive to outliers. This working point is meaningful only in symmetrical distributions where $\mathbf{x}_{ml} = \mathbf{x}_{ls}$.

$\mathbf{x}_{opt}$ is the most practical value to choose in the general distribution case. In this method, we first look at the subspace of the probability distribution with all the possible choices for $\mathbf{x}$. Then an external optimization scheme is applied, and the working point is chosen. $\mathbf{x}_{opt}$ is much more stable than $\mathbf{x}_{ls}$, and if a good optimization scheme can be imposed, there is a good chance that $\mathbf{x}_{opt}$ will be the correct process state. The proper value for ϵ depends mainly on the noise level. If it is small, we choose a value close to 1, so that $\mathbf{x}_{opt}$ will be more affected by the maximum likelihood location. On the other hand, if the noise level is higher, we choose for ϵ a value close to 0. In this case, $\mathbf{x}_{opt}$ will be more affected by the optimization.

1.5. Adaptive filtering

In order to use the filter, the state transition function $\Phi_{t,t+1}(\mathbf{x})$ must be known. In many cases, it is impossible to know the exact function. Instead, $\Phi_{t,t+1}$ can be approximated as a simple function for short time intervals. For example, consider tracking a car on a road. We cannot know the exact function that describes the car motion. But we can assume that the car travels with a constant velocity for short time intervals. So the function $\Phi_{t,t+1}(\mathbf{x})$ is constant, where the process $\mathbf{x}$ is the velocity. In these cases, the dynamics of the filter is not valid for long time intervals, because of the changes in the process. These changes in the process dynamics are usually slow, so that for a short time, the filter is still useful. In order for the filter to be used for longer periods, it must be adaptive. The essential meaning of the adaptivity is that less weight is given to old measurements than new ones or in other words, weakening the filtering effect. This way, the filter adapts itself to the changes in the process.

We will now describe two implementations for the filter's adaptivity:

- Control over the filter adaptivity can be achieved by choosing the process noise probability distribution function $P_{\mathbf{n}_t}$. A function like $P_{\mathbf{n}_t}(\mathbf{u}) = \delta(\mathbf{u})$ means that the process has no noise, and will cause the filter to be not adaptive at all. On the other hand, choosing a constant function like $P_{\mathbf{n}_t}(\mathbf{u}) = K$ will cause the filter to forget all the old measurements, and we will totally lose the filtering effect. What is needed is a function whose shape is between these two extremes. As the function becomes broader, the filter is more adaptive and the filtering effect weaker. The exact filter shape should be fitted for every case individually.

- Another possibility is determining the adaptivity rate directly. This can be done by updating the process probability distribution $P_{\mathbf{x}_{t+1}}(\mathbf{v})$ before taking the new measurement:

$$P'_{\mathbf{x}_t}(\mathbf{v}) = P_{\mathbf{x}_t}(\mathbf{v})^{1/k}$$

for $k > 1$. The factor k directly controls the adaptivity of the filter by giving exponentially less weight to previous measurements. For example, letting k=2 gives a weight of 1 to the current measurement, 1/2 to the last one, 1/4, 1/8, 1/16... to the preceding ones, and so on. [1]

After updating the probability distribution, it should be normalized so that the sum over the probability space is 1.

1.6. Resetting the filter

Another common situation is an abrupt change in the process dynamics. For example, a body is falling with a constant acceleration and stops abruptly on the floor. Usually the filter adaptivity is not enough to overcome this. For such quick changes, the needed adaptivity factor would have to be so large that the filter would lose its effect. A better approach is a detector that checks if each new measurement is feasible under the current process state. If it is, the measurement is accepted, if not, we reset the filter, because the process is not valid any more. The description of this detector follows: [2]

In order to determine if the measurement is feasible, we compare the expected probability to see the measurement when $P(\mathbf{x}_t^-)$ is known, to the probability to see the measurement where no apriori information is available (i.e. assuming equal distribution of $\mathbf{x}_t^-$):

$$\gamma = \frac{\mathcal{E}\{P(\mathbf{y_t}|\mathbf{x}_t^-)\}}{P(\mathbf{y}_t)} = \frac{\int P(\mathbf{y}_t|\mathbf{u})\, P(\mathbf{x}_t^- = \mathbf{u})\, d\mathbf{u}}{\int P(\mathbf{y}_t|\mathbf{u})\, K\, d\mathbf{u}} \quad (5)$$

[1] This adaptivity method is known and used in Kalman filter and Autoregressive filters as in [2] and is adapted and expanded here for our general filter.

[2] This general idea is usually used in Kalman filter to reject wrong measurements (see [3]).

Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)
1063-6919/97 $10.00 © 1997 IEEE

where K is such that $\int_{\mathbf{u}\in\mathcal{U}} K\, d\mathbf{u} = 1$
if $\gamma > 1$ the probability to see the current measurement is higher, given the current state distribution $P(\mathbf{x}_t^-)$, and so we accept the process as valid. On the other hand, if $\gamma < 1$, the possibility to see the current measurement is less probable given the state distribution matrix $P(\mathbf{x}_t^-)$, so we assume that the process is not valid any more and the process history should be reset. In cases where no changes in the process's dynamic are expected, instead of resetting the filter, the current measurement should be ignored.

2. The practical implementation of the general distribution filter

The filter was derived for the case of general probability distribution function. A computer implementations of the filter are usually practical only for discrete distributions. However, even a continuous distribution can be approximated by a discrete distribution. In many times, the discretization can be a much better approximation than the Gaussian. The problem is that for a real-time implementation, the matrix should be relatively small. When the state vector and the measurement are 2-dimensional, the probability distribution can be described as matrices. But for higher-order cases the matrix become cubes or a hypercubes. The memory size needed for the implementation grows and the process is more time consuming. This is the main drawback of this filter, as compared with the Kalman filter, which is more compact and practical for larger state vectors. In the next chapter we demonstrate the implementation of the filter through a practical example from image processing, where this filter is quite practical and efficient.

3. A practical example: Tracking a moving object in a video sequence

A moving object is inspected by a video camera and its location on the image should be tracked. A set of n points on the object are given in the first frame, and it is known that the displacement $\mathbf{d} = (u, v)$ of the points between consecutive frames is within the range: $U_{min}..U_{max}$, $V_{min}..V_{max}$. The object is tracked by tracking the mean motion of the n points. A complete description of the tracking system can be found in [4] and [5], we show here only the outline. The first stage in the tracking system is to find the displacement of a single point. It can be shown that the point's displacement can be found confidentially only when the point is a sharp corner. In other cases, the displacement cannot be found precisely, and it should be described as a probability distribution. Fig. 1 shows such a probability matrix and its Gaussian approximation. It can be seen how the Gaussian approximation does not preserve the information held in the probability matrix because its peak fall in an area where the real probability is almost zero.

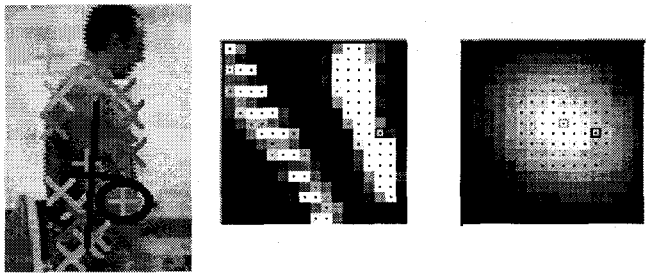


Figure 1. Center: the probability matrix of the point inside the circle. Right: Gaussian approximation.

Filtering The velocity of the tracked point is assumed to be roughly constant in a short sequence. Therefore, filtering can be used to improve the point's tracking, where the process is the velocity of the tracked point and each frame is a measurement. In many papers, the probability distribution of the displacement was approximated as a Gaussian, and the Kalman filter was used. However, this is unjustified, as the probability matrix can have an arbitrary shape, and approximating it as a Gaussian causes severe information loss. However, with the filter introduced here these problem are avoided. Besides, the size of the probability matrix is relatively small, so an efficient implementation is possible. Fig. 2 demonstrate the filtering effect.

Using the tracked points to compute the object's motion Given the displacement probability matrices of the N points, the object's mean motion is computed as follows: Let $\mathbf{P}_i$ be the distribution matrix of the i-th tracked point. Let

$$\mathbf{P}_{sum}(x, y) = \sum_{i=1}^{N} \mathbf{P}_i(x, y)$$

For each value (u, v), the expected squared error will be:

$$MSE(u, v) = \frac{\sum_x \sum_y P_{sum}(x, y)[(x-u)^2 + (y-v)^2]}{\sum_x \sum_y P_{sum}(x, y)}$$

$(\bar{u}, \bar{v})$ minimizes MSE, where:

$$\bar{u} = \frac{\sum_x \sum_y P_{sum}(x, y)\, x}{\sum_x \sum_y P_{sum}(x, y)}$$

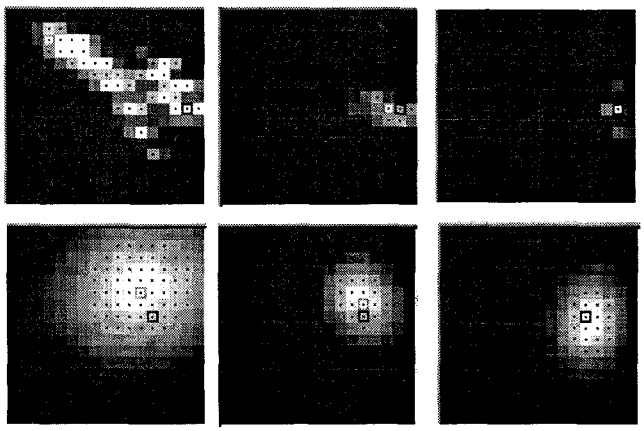
$$\bar{v} = \frac{\sum_x \sum_y P_{sum}(x, y)\, y}{\sum_x \sum_y P_{sum}(x, y)}$$

Choose $(\bar{u}, \bar{v})$ as the object's motion.

The right matrix in Fig. 2 shows $\mathbf{P}_{sum}$ at one frame.

Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)
1063-6919/97 $10.00 © 1997 IEEE

Figure 2. First row: The probability matrix of one point (left), after filtering (middle) and the all points sum matrix (right). Second row: Approximating all probabilities as Gaussians.



Updating the tracked point's location in order to keep the tracked point on the same image feature each tracked point must choose a single value for the displacement, and update its location accordingly in the next frame.

As discussed in Section 1.3, choosing the maximum likelihood entry from the matrix yields a very noisy result. A better choice for the working point is $\mathbf{x}_{opt}$. The optimization criteria used is, that the point moves as close as possible to the object's mean motion.

More details and experimental results can be found in [4] and [5].

4. Summary

We have introduced the general distribution filter as an alternative to the Kalman filter in cases where the measurement's distribution cannot be approximated by a Gaussian. An effective computer implementation of this filter is possible when the measurement has a discrete distribution, or when it can be approximated as discrete. The filter was implemented in a tracking application, but it can have various applications in other fields as well.

A. Derivation of the Kalman filter as a special case of the global distribution filter

In this appendix we show how the Kalman filter can be derived as a special case of the global distribution filter when imposing the Kalman filter constraints.

Transition in time of the state from *t* to *t+1*. In the Kalman filter case, the transition function Φ is a square matrix, and the state vector at t+1 is:

$$\mathbf{x}_{t+1} = \Phi\, \mathbf{x}_t$$

We will consider the case that Φ is invertible, so that Φ^{-1} exists [3]. For this case, Equation 2 turns to:

$$P_{\mathbf{x}_{t+1}} = P_{\mathbf{x}_t}(\Phi^{-1}\, \mathbf{u}) \tag{6}$$

$\mathbf{x}_t$ is a Gaussian:

$$P_{\mathbf{x}_t}(\mathbf{u}) = K e^{(\mathbf{u}-\mathbf{m}_t)^T \Sigma_t^{-1} (\mathbf{u}-\mathbf{m}_t)/2} = \mathcal{N}(\mathbf{m}_t, \Sigma_t) \tag{7}$$

Using Equations 6 and 7, it can be shown that the probability distribution of $P_{\mathbf{x}_{t+1}}$ is also a Gaussian:

$$P_{\mathbf{x}_{t+1}}(\mathbf{u}) = K e^{(\mathbf{u}-\mathbf{m}_{t+1})^T \Sigma_{t+1}^{-1} (\mathbf{u}-\mathbf{m}_{t+1})/2}$$

where:

$$\mathbf{m}_{t+1} = \Phi \mathbf{m}_t,\ \Sigma_{t+1} = \Phi \Sigma_t \Phi^T$$

Therefore the state vector after the transition is:

$$\mathbf{x}'_{t+1} = \mathcal{N}(\Phi \mathbf{m}_t,\ \Phi \Sigma_t \Phi^T) \tag{8}$$

where $\mathbf{x}'_{t+1}$ is the transition of state *t* to state *t+1* without the process noise. We see that if $\mathbf{x}_t$ is Gaussian, then $\mathbf{x}_{t+1}$ is also Gaussian. To get the complete state transition equation we have to add the noise effect.

The process noise

We assume that the transition noise $\mathbf{n}_t(\mathbf{u})$ is a multivariate Gaussian distribution with covariance matrix Σ_n and a mean of $\mathbf{0}$. Thus:

$$\mathbf{n}_t(\mathbf{u}) = K\, e^{-\mathbf{u}^T \Sigma_n\, \mathbf{u}/2} = \mathcal{N}(\mathbf{0},\ \Sigma_n)$$

As we have seen, the sum of two uncorrelated random vectors is also a random vector. The probability distribution of the sum is the convolution:

$$P_{\mathbf{x}}(\mathbf{u}) = P_{\mathbf{x}_1}(\mathbf{u}) \otimes P_{\mathbf{x}_2}(\mathbf{u})$$

If $\mathbf{x}_1$ and $\mathbf{x}_2$ are uncorrelated Gaussians, where $\mathbf{x}_1 = \mathcal{N}(\mathbf{u}_1,\ \Sigma_1)$ and $\mathbf{x}_2 = \mathcal{N}(\mathbf{u}_2,\ \Sigma_2)$, the convolution turns into a simpler form:

$$\bar{\mathbf{x}} = \mathbf{x}_1 + \mathbf{x}_2 = \mathcal{N}(\mathbf{u}_1 + \mathbf{u}_2,\ \Sigma_1 + \Sigma_2)$$

which is also Gaussian.

In our case, the two random vectors are the process and the process noise:

$$\mathbf{n}_t(\mathbf{u}) = \mathcal{N}(0,\ \Sigma_n) \tag{9}$$

$$\mathbf{x}'_{t+1} = \mathcal{N}(\Phi \mathbf{m}_t,\ \Phi \Sigma_t \Phi^T) \tag{10}$$

[3] We do not derive here the case where Φ is not invertible. In this case, the definition of Φ^{-1} can be extended using the singular value decomposition for Φ.

Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)
1063-6919/97 $10.00 © 1997 IEEE

From the convolution in Equation 3 we get:

$$\mathbf{x}_{t+1}^{-} = \mathcal{N}(\Phi \mathbf{m}_t,\ \Phi \Sigma_t \Phi^T + \Sigma_n)$$

where $\mathbf{x}_{t+1}^{-}$ is the a priori state vector at $t+1$.

Taking the measurement

So far we have found the apriori probability distribution. After taking the measurement we get the aposteriori distribution.

In the Kalman filter, the measurement probability distribution is:

$$P_{\mathbf{y}|\mathbf{x}} = \mathbf{C}\,\mathbf{x} + \mathcal{N}(0,\ \mathbf{\Sigma_y}) = \mathcal{N}(\mathbf{C}\,\mathbf{x},\ \mathbf{\Sigma_y})$$

and the process is

$$P_{\mathbf{x}} = \mathcal{N}(\bar{\mathbf{x}},\ \mathbf{\Sigma_x})$$

In the Kalman filter it is assumed that the state vector $\mathbf{x}$ and the measurement noise $\mathcal{N}(0,\ \mathbf{\Sigma_y})$ are uncorrelated.
The aposteriori distribution:

$$P_{\mathbf{x}}^{+} = P(\mathbf{x} \mid \mathbf{y}) = \frac{P(\mathbf{y} \mid \mathbf{x})P(\mathbf{x})}{P(\mathbf{y})} = \frac{P(\mathbf{y} \mid \mathbf{x})P(\mathbf{x})}{\int P(\mathbf{y} \mid \mathbf{x})P(\mathbf{x})dx}$$

It can be shown that $P_{\mathbf{x}}^{+}$ is a Gaussian with:

$$\bar{\mathbf{x}}^{+} = \bar{\mathbf{x}} + \mathbf{G_x}\,(\mathbf{y} - \mathbf{C}\,\bar{\mathbf{x}})$$

$$\mathbf{\Sigma_x}^{+} = (\mathbf{I} - \mathbf{G_x}\,\mathbf{C})\bar{\mathbf{x}}$$

where

$$\mathbf{G_x} = \mathbf{\Sigma_x}\,\mathbf{C}^T\,(\mathbf{C}\,\mathbf{\Sigma_x}\,\mathbf{C}^T + \mathbf{\Sigma_y})^{-1}$$

References

[1] R.O. Duda and P. E. Hart. *Pattern classification and scene analysis*. Wiley, New-York, 1973.

[2] M. S. Grewal. *Kalman filtering: Theory and practice*. Prentice-Hall, 1993.

[3] S. Haykin. *Adaptive Filter Theory*. Prentice-Hall, 1986.

[4] Y. Rosenberg and M. Werman. Representing local motion as a probability distribution matrix and object tracking. *IUW 97 DARPA*.

[5] Y. Rosenberg and M. Werman. Representing local motion as a probability distribution matrix for object tracking and other applications. *http://www.cs.huji.ac.il/papers/IP/CVPR97/index.html*.

Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)
1063-6919/97 $10.00 © 1997 IEEE