On the initialization of statistical optimum filters with application to motion estimation

Laurent Kneip, Davide Scaramuzza and Roland Siegwart
Autonomous Systems Lab, ETH Zurich

***Abstract*— The present paper is focusing on the initialization of statistical optimum filters for motion estimation in robotics. It shows that if certain conditions concerning the stability of a system are fulfilled, and some knowledge about the mean of the state is given, an initial error covariance matrix that is optimal with regard to the convergence behavior of the filter estimate might be analytically obtained. Easy algorithms for the n-dimensional continuous and discrete cases are presented. The applicability to non-linear systems is also pointed out. The convergence of a normal Kalman filter is analyzed in simulation using the discrete model of a theoretical example.**

I. Introduction

Statistical optimum filtering techniques certainly belong to the class of the more powerful developments of the past century. The foundation has been layed down around 1960 by Swerling[1], Thiele[2], [3], Stratonovich[4], [5] and Kalman[6], and found its first applications in spacial navigation and communication technologies. Today, it is mostly used in a discrete recursive - and thus efficient - form for solving demanding real-time filtering and sensor fusion problems[7], [8], [9]. It is commonly known as producing optimal results in case of being applied to linear systems, where optimal means generating online state estimates with minimized error covariance. Several extensions to the classical Kalman filter like the EKF (Extended Kalman Filter) and the UKF (Unscented Kalman Filter) then also address the problem of filtering signals at the output of non-linear systems. They however quickly engage a lot of processing resources and do no longer produce optimal results. The EKF filter is doing a linearization of the non-linear system around the current state estimate, and thus produces errors when propagating the error covariance estimate through the model. The UKF improves this by the so-called unscented transform, which generally spoken consists in propagating the error covariance estimate through a number of weighted sampling points in the neighbourhood of the current state estimate (the so-called sigma-points). It provides a more accurate value for the propagated covariance in case of highly non-linear systems.

Numerous internet tutorials and book chapters about the discrete Kalman filter and its application to a huge variety of different fields are evidence of its tremendous character. Even classical problems like for instance SLAM[10] in robotics (Simultaneous Localization And Mapping) could be reformulated in a statistical way[11]. On the other hand, the availability of compact and independent collections of the formulas as well as the filter's image of having a universal applicability causes people to easily apply these "cooking recipies" without questioning the fundamental theories behind it far too often. This is especially true for highly interdisciplinary fields like robotics and autonomous systems. A proof might be given by the related Wikipedia article that is, unlike other topics, not introducing the concept in a very objective way by exclusively focusing on the discrete implementation of the Kalman filter, certainly at the same time a proof for the complexity of the material.

The present paper addresses one specific problem that is common to the classical Kalman filter as well as its discrete derivates for linear and non-linear systems, namely the choice of the initial error covariance matrix. Only little literature can be found on this topic despite of a high importance for the convergence of the filter. The commonly applied way in robotics consists in an empirical choice of positive starting values along the diagonal of the covariance matrix, each one representing the error covariance of the corresponding initial state estimate at the moment we switch on the filter. Even though there is nothing wrong about this, it is indeed possible to derive the initial filter covariance matrix in a rigorous analytical manner.

The purpose of the present work is to state the exact conditions that have to be met in order to easily calculate an exact initial error covariance matrix, provide a comprehensive algorithm for doing so and show the impact on the convergence behavior of motion estimation filtering. Section II starts with introducing the generic system structure and terminology used in this paper, and reminds some important equations for the continuation of the work. Section III then proceeds with the derivation of an easy method for obtaining a valid initial covariance matrix for both the continuous and discrete n-dimensional case. The applicability to non-linear systems will also be depicted. Section IV concludes with evaluating the convergence by comparing it to an empirical choice of the initial covariance matrix. It shows the optimal character of the methods described in this paper.

II. Terminology

Figure 1 shows the general state-space representation of a continuous system used as a basis for describing the theoretical material in this paper. The terminology used is mostly taken from [12], an excellent german book about system theory for stochastic processes. The only input to the system is the process noise $\mathbf{z}$, no other deterministic signal is used. This does not involve loss of genericy since not having any impact on the evolution equations of the system state covariance. The system matrix $\mathbf{A}$, the input matrix $\mathbf{G}$

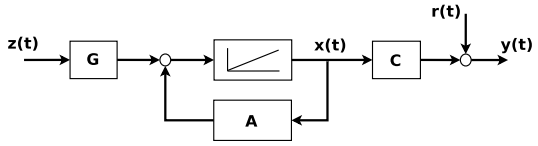

Fig. 1. The generic state-space representation of a continuous system with input process noise $\mathbf{z}(e;t)$ and output measurement noise $\mathbf{r}(e;t)$.

and the output matrix $\mathbf{C}$ maybe regarded as constant during time. The related process and measurement equations are given with

$$\dot{\mathbf{x}}(e;t) = \mathbf{A}\mathbf{x}(e;t) + \mathbf{G}\mathbf{z}(e;t) \tag{1}$$
$$\mathbf{y}(e;t) = \mathbf{C}\mathbf{x}(e;t) + \mathbf{r}(e;t). \tag{2}$$

The parameters $(e;t)$ express the stochastic nature of the signals as well as the dependency of time, respectively. Bold capitals indicate matrices, bold minuscules vectors. The noise in the system has the following properties:

- Input process $\mathbf{z}(e;t)$: non-stationary white gaussian noise with mean $\mu_\mathbf{z} = 0$ and auto-correlation $\mathbf{\Phi_z}(t,\tau) = \mathbf{\Psi_z}(t) \cdot \delta(\tau)$, $\mathbf{\Psi_z}$ being positive semi-definite and holding the squared covariance and cross-covariance terms of the different elements in $\mathbf{z}$.
- Measurement noise $\mathbf{r}(e;t)$: non-stationary white gaussian noise with mean $\mu_\mathbf{r} = 0$ and auto-correlation $\mathbf{\Phi_r}(t,\tau) = \mathbf{\Psi_r}(t) \cdot \delta(\tau)$, $\mathbf{\Psi_r}$ again being positive semi-definite.

The procedure presented in this paper is largely based on the evolution equation of the mean $\mu_\mathbf{x}$ and covariance matrix $\mathbf{V_x}$ of the system state $\mathbf{x}$ (not to confuse with the error covariance matrix $\mathbf{V}$ of the filter's state estimation $\hat{\mathbf{x}}$). The derivation of the equations is fairly simple. Since the following only presents a compact form of the derivation, the reader is invited to look up the detailed steps in the relevant literature[12]. Via building the expectation value on both sides of Eq. 1, we directly obtain

$$\dot{\mu}_\mathbf{x}(t) = \mathbf{A} \cdot \mu_\mathbf{x}(t). \tag{3}$$

Assuming unbiased system states, the covariance matrix may be computed via the equation

$$\mathbf{V_x}(t) = E\{\mathbf{x}(e;t) \cdot \mathbf{x^t}(e;t)\},$$

which represents nothing more than the vectorial time-variant squared mean of the state. Computing the derivation easily leads to

$$\dot{\mathbf{V}}_\mathbf{x}(t) = E\{\dot{\mathbf{x}}(e;t) \cdot \mathbf{x^t}(e;t)\} + E\{\mathbf{x}(e;t) \cdot \dot{\mathbf{x}}^\mathbf{t}(e;t)\},$$

and replacing with Eq. 1 leads to the following first form of the evolution equation

$$\dot{\mathbf{V}}_\mathbf{x}(t) = \mathbf{A}\mathbf{V_x}(t) + \mathbf{V_x}(t)\mathbf{A^t} + \mathbf{G}\mathbf{V_{zx}}(t) + \mathbf{V_{xz}}(t)\mathbf{G^t}.$$

Knowing that $\mathbf{V_{zx}}(t) = \frac{1}{2}\mathbf{\Psi_z}(t)\mathbf{G^t}$ and $\mathbf{V_{xz}}(t) = \mathbf{V_{zx}^t}(t)$ (again, see [12] for details), we finally obtain

$$\dot{\mathbf{V}}_\mathbf{x}(t) = \mathbf{A}\mathbf{V_x}(t) + \mathbf{V_x}(t)\mathbf{A^t} + \mathbf{G}\mathbf{\Psi_z}(t)\mathbf{G^t}. \tag{4}$$

This form of equation is also called a degenerated Riccati equation, since it has the same form except that the higher order term is missing.

A similar development is now done for discrete systems. Figure 2 indicates the modified system in state-space representation. The corresponding equations are

$$\mathbf{x}[k+1] = \mathbf{A}\mathbf{x}[k] + \mathbf{G}\mathbf{z}[k] \tag{5}$$
$$\mathbf{y}[k] = \mathbf{C}\mathbf{x}[k] + \mathbf{r}[k]. \tag{6}$$

Regarding the noise in the system, we now have

$$E\{\mathbf{z}[e;k]\} = \mathbf{0}, E\{\mathbf{r}[e;k]\} = \mathbf{0},$$
$$E\{\mathbf{z}[e;k] \cdot \mathbf{z^t}[e;l]\} = \delta_{kl} \cdot \mathbf{\Psi_z},$$
$$E\{\mathbf{r}[e;k] \cdot \mathbf{r^t}[e;l]\} = \delta_{kl} \cdot \mathbf{\Psi_r},$$
$$E\{\mathbf{z}[e;k] \cdot \mathbf{r^t}[e;l]\} = \mathbf{0}, \ \forall k,l$$
$$\text{with } \delta_{kl} = 1 \text{ if } k = l \text{ and else } 0.$$

Again, via taking the expectation value of Eq. 5, we directly obtain the evolution equation for the mean of the state

$$\mu_\mathbf{x}[k+1] = \mathbf{A} \cdot \mu_\mathbf{x}[k]. \tag{7}$$

The derivation of the evolution equation for the state covariance is significantly easier in the discrete case. We have

$$\begin{aligned}\mathbf{V_x}[k+1] &= E\{\mathbf{x}[e;k+1] \cdot \mathbf{x^t}[e;k+1]\} \\ &= E\{(\mathbf{A}\mathbf{x}[e;k] + \mathbf{G}\mathbf{z}[e;k]) \cdot (\mathbf{A}\mathbf{x}[e;k] + \mathbf{G}\mathbf{z}[e;k])^t\} \\ &= E\{(\mathbf{A}\mathbf{x}[e;k] + \mathbf{G}\mathbf{z}[e;k]) \cdot (\mathbf{x^t}[e;k]\mathbf{A^t} + \mathbf{z^t}[e;k]\mathbf{G^t})\} \\ &= \mathbf{A}E\{\mathbf{x}[e;k]\mathbf{x^t}[e;k]\}\mathbf{A^t} + \mathbf{G}E\{\mathbf{z}[e;k]\mathbf{x^t}[e;k]\}\mathbf{A^t} \\ &\quad + \mathbf{A}E\{\mathbf{x}[e;k]\mathbf{z^t}[e;k]\}\mathbf{G^t} + \mathbf{G}E\{\mathbf{z}[e;k]\mathbf{z^t}[e;k]\}\mathbf{G^t}\end{aligned}$$

The process noise $\mathbf{z}[e;k]$ and the "old" state $\mathbf{x}[e;k]$ are independent. Hence we have $E\{\mathbf{z}[e;k]\mathbf{x^t}[e;k]\} = E\{\mathbf{x}[e;k]\mathbf{z^t}[e;k]\} = 0$, and we finally obtain

$$\mathbf{V_x}[k+1] = \mathbf{A}\mathbf{V_x}[k]\mathbf{A^t} + \mathbf{G}\mathbf{\Psi_z}\mathbf{G^t} \tag{8}$$

as the evolution equation for the state covariance in the discrete case.

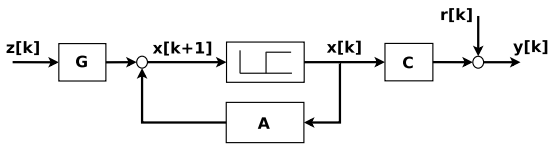

Fig. 2. The generic state-space representation of a discrete system with input process noise $\mathbf{z}[k]$ and output measurement noise $\mathbf{r}[k]$.

III. Obtaining the initial error covariance matrix

The most important thing to understand is that the initial error covariance is not depending on the measurement noise. By definition, the initial error covariance is expressing the error of the initial state estimate, a value that is set by hand before the filter is engaged. In the discrete case, this is for instance representing an a priori estimate used for correcting the first measurement, and hence it is obvious that it is neither taking any measurements into account and nor depending on the measurement noise. The only difference to a standard a priori estimate is that it has not been obtained through propagation through the model, which is again obvious since there is no previous state estimate.

The following development is based on the assumption that the system is in a statistically stationary mode when switching on the filter. First of all, there are some criteria that have to be met in order to even allow a stationary mode:

- The system has to be able to stabilize in a certain point, which means for instance being self-stable around a known location in state-space, despite of process noise acting on the true state, or being stably controlled around a known set-point for the state. In the latter case, a known deterministic setpoint input could additionally determine the system equations.
- The process noise has to enter a stationary mode, which means that the corresponding auto-correlation matrix should no longer be time dependent.

The reader might now ask why all this is needed. The point is that it provides means to get an idea about the expectation value of the system state and its covariance at the moment we switch on the filter. Doing so when the system is in a stationary mode then increases freedom in time, since it does no longer matter when we turn on the filter exactly. If, for whatever reason, some knowledge about the expectation value and its covariance is given for a certain instant in time, and if we are able to switch on the filter at exactly this moment, the principle depicted here might also be applied to non-stationary cases. In practice however, this is most probably not the case.

Once these conditions are fulfilled, the basic steps for obtaining the initial covariance matrix are rather easy. Since the different steps are depending on whether we work in continuous or discrete time space, the following is explained individually.

A. Continuous case

According to the initial conditions we mentioned beforehand, we may assume that we have a certain knowledge about the state of our system when engaging the filter, which helps to find a good starting value for the initial filter estimate. As always, $E(.)$ is taking the expectation value and $\hat{\mathbf{x}}$ represents the state estimate. We have $\hat{\mathbf{x}}(e;0) = E\{\mathbf{x}(e;0)\} = \mu_{\mathbf{x}}(0)$, and since we assume being in stationary mode, $\mu_{\mathbf{x}}(0)$ is known (in our case, it is $\mathbf{0}$ simply because we do not have any determined additional inputs). Please be aware that the time scale here is shifted with respect to Section II. Since being in the stationary mode, $t = 0$ for the filter actually corresponds to $t \to \infty$ for the observed system.

The next property we might derive in the mentioned mode is about the rate at which the covariance of the system state is changing, namely that we have $\dot{\mathbf{V}}_{\mathbf{x}}(0) = \mathbf{0}$. From Equation 4, we may then directly derive that

$$\mathbf{0} = \mathbf{A}\mathbf{V}_{\mathbf{x}}(0) + \mathbf{V}_{\mathbf{x}}(0)\mathbf{A}^{\mathbf{t}} + \mathbf{G}\boldsymbol{\Psi}_{\mathbf{z}}\mathbf{G}^{\mathbf{t}}. \tag{9}$$

Note that $\boldsymbol{\Psi}_{\mathbf{z}}$ is no longer a function of time. From the definition of the initial error covariance $\mathbf{V}(0)$, we then find that

$$\begin{aligned}
\mathbf{V}(0) &= E\{[\hat{\mathbf{x}}(e;0) - \mathbf{x}(e;0)] \cdot [\hat{\mathbf{x}}(e;0) - \mathbf{x}(e;0)]^t\} \\
&= E\{[\mu_{\mathbf{x}}(0) - \mathbf{x}(e;0)] \cdot [\mu_{\mathbf{x}}(0) - \mathbf{x}(e;0)]^t\} \\
&= \mu_{\mathbf{x}}^{\mathbf{2}}(0) - 2\mu_{\mathbf{x}}(0) \cdot E\{\mathbf{x}(e;0)\} + E\{\mathbf{x}(e;0)\mathbf{x}^t(e;0)\} \\
&= \mu_{\mathbf{x}}^{\mathbf{2}}(0) - 2\mu_{\mathbf{x}}^{\mathbf{2}}(0) + E\{\mathbf{x}(e;0)\mathbf{x}^t(e;0)\} \\
&= E\{\mathbf{x}(e;0)\mathbf{x}^t(e;0)\} - \mu_{\mathbf{x}}^{\mathbf{2}}(0) \\
&= \mathbf{V}_{\mathbf{x}}(0)
\end{aligned}$$

This equation expresses the fact that the initial error covariance of our filter is equal to the state covariance of the model in the stationary mode, which is actually quite meaningful. By replacement in Equation 9, we finally obtain a formula to determine the initial error covariance matrix, namely

$$\mathbf{A}\mathbf{V}(0) + \mathbf{V}(0)\mathbf{A}^{\mathbf{t}} = -\mathbf{G}\boldsymbol{\Psi}_{\mathbf{z}}\mathbf{G}^{\mathbf{t}}. \tag{10}$$

The challenge now consists in solving this equation for $\mathbf{V}(0)$. For matters of simplicity, the right side of the equation is replaced with $\mathbf{M} = -\mathbf{G}\boldsymbol{\Psi}_{\mathbf{z}}\mathbf{G}^{\mathbf{t}}$. Assuming that the dimension of the problem is n, the equation is equivalent to ($r(.,.)$ is taking a certain row from a matrix)

$$\begin{gathered}
r(\mathbf{A},1)r^t(\mathbf{V},1) + r(\mathbf{A},1)r^t(\mathbf{V},1) = M_{11} \\
r(\mathbf{A},1)r^t(\mathbf{V},2) + r(\mathbf{A},2)r^t(\mathbf{V},1) = M_{12} \\
\dots \\
r(\mathbf{A},1)r^t(\mathbf{V},n) + r(\mathbf{A},n)r^t(\mathbf{V},1) = M_{1n} \\
r(\mathbf{A},2)r^t(\mathbf{V},2) + r(\mathbf{A},2)r^t(\mathbf{V},2) = M_{22} \\
\dots \\
r(\mathbf{A},2)r^t(\mathbf{V},n) + r(\mathbf{A},n)r^t(\mathbf{V},2) = M_{2n} \\
\dots \\
r(\mathbf{A},n)r^t(\mathbf{V},n) + r(\mathbf{A},n)r^t(\mathbf{V},n) = M_{nn}
\end{gathered}$$

Note that only the equations corresponding to the upper triangular part of $\mathbf{M}$ are being considered. This is sufficient since $\mathbf{V}(0)$ is a symmetric matrix just as $\mathbf{M}$ and hence the number of unknowns corresponds to the number of equations. It is also straight-forward to see that each of the remaining equations would actually be equivalent to one of the listed ones. The terms $r(\mathbf{A},i) \cdot r^t(\mathbf{V}(0),j)$ always create a sum of n products where each time one element from $\mathbf{A}$ is multiplied with one element from $\mathbf{V}(0)$. We have

$$r(\mathbf{A},i) \cdot r^t(\mathbf{V}(0),j) = \Sigma_{k=1}^{n} A_{ik}V_{jk}.$$

Let's put the unknowns of $\mathbf{V}(0)$ into a vector $\mathbf{v}$ according to the following principle

$$\mathbf{v} = \begin{pmatrix} V_{11} \\ V_{12} \\ ... \\ V_{1n} \\ V_{22} \\ ... \\ V_{2n} \\ ... \\ V_{nn} \end{pmatrix}.$$

Again, only the elements of the upper triangular part are considered. The same may be done for the right side of the equation leading to a vector $\mathbf{m}$ that contains the elements of the upper triangular part of $\mathbf{M}$. Both vectors contain $q = \frac{n(n+1)}{2}$ elements. The essential step now consists in introducing $\mathbf{v}$ into our row-multiplications. This is done using the function $f(i,j)$, which is producing a row-vector of size q with all elements being zero except the one at index p being equal to one, with

$$p(i,j) = \begin{cases} (\Sigma_{l=1}^{i-1} n - l + 1) + j - (i-1) \text{ if } i \leq j \\ (\Sigma_{l=1}^{j-1} n - l + 1) + i - (j-1) \text{ if } i > j \end{cases}.$$

The function $f(i,j)$ actually does nothing else then a mapping between a certain element in the matrix $\mathbf{V}(0)$ and the corresponding element in the vector $\mathbf{v}$, all while respecting the symetrical structure of $\mathbf{V}(0)$. This finally leads to

$$r(\mathbf{A}, i) \cdot r^t(\mathbf{V}(0), j) = (\Sigma_{k=1}^{n} A_{ik} f(j,k)) \cdot \mathbf{v},$$

and by replacing in our initial list of equations, our problem may be reformulated as

$$\begin{pmatrix} \Sigma_{k=1}^{n} A_{1k} f(1,k) + \Sigma_{k=1}^{n} A_{1k} f(1,k) \\ \Sigma_{k=1}^{n} A_{1k} f(2,k) + \Sigma_{k=1}^{n} A_{2k} f(1,k) \\ ... \\ \Sigma_{k=1}^{n} A_{1k} f(n,k) + \Sigma_{k=1}^{n} A_{nk} f(1,k) \\ \Sigma_{k=1}^{n} A_{2k} f(2,k) + \Sigma_{k=1}^{n} A_{2k} f(2,k) \\ ... \\ \Sigma_{k=1}^{n} A_{2k} f(n,k) + \Sigma_{k=1}^{n} A_{nk} f(2,k) \\ ... \\ \Sigma_{k=1}^{n} A_{nk} f(n,k) + \Sigma_{k=1}^{n} A_{nk} f(n,k) \end{pmatrix} \cdot \mathbf{v} = \mathbf{m}.$$

Let's define $\mathbf{S}$ such that $\mathbf{S} \cdot \mathbf{v} = \mathbf{m}$. The solution to obtain the elements of $\mathbf{V}(0)$ thus finally results to

$$\mathbf{v} = \mathbf{S}^{-1} \cdot \mathbf{m}. \tag{11}$$

The pseudocode for creating the matrix $\mathbf{S}$ and generally solving for $\mathbf{v}$/$\mathbf{V}(0)$ is given with Algorithm 1.

Algorithm 1 Procedure for computing V(0) in function of A, G and Z in the continuous case.

```
M = −G · Z · G^t
n = (dimension of A)
q = n(n+1)/2                                   ▷ number of unknowns
H = (identity matrix of dimension q)
                                   ▷ H helps for f to get row-vectors
index = 1                          ▷ create matrix S and vector m
for i = 1 to n do
    for j = i to n do
        for k = 1 to n do
            p = 0
            if j ≤ k then
                for l = 1 to j − 1 do
                    p = p + n − l + 1
                end for
                p = p + k − (j − 1)
            else
                for l = 1 to k − 1 do
                    p = p + n − l + 1
                end for
                p = p + j − (k − 1)
            end if
            (row index of S) =
            (row index of S) + A_{i,k} · (row p of H)
        end for
        for k = 1 to n do
            p = 0
            if i ≤ k then
                for l = 1 to i − 1 do
                    p = p + n − l + 1
                end for
                p = p + k − (i − 1)
            else
                for l = 1 to k − 1 do
                    p = p + n − l + 1
                end for
                p = p + i − (k − 1)
            end if
            (row index of S) =
            (row index of S) + A_{j,k} · (row p of H)
        end for
        (element index of m) = M_{i,j}
        index = index + 1
    end for
end for

v = (inverse of S) · m                             ▷ solve for v
index = 1                                          ▷ create V(0)
for i = 1 to n do
    for j = i to n do
        V_{i,j} = v_index
        V_{j,i} = v_index
        index = index + 1
    end for
end for
```

B. Discrete case

The derivations in the discrete case are a bit different. We may again assume that we know $\mu_{\mathbf{x}}[0]$, the mean of the system state during the stationary mode. The conclusions about the state covariance in this mode result to $\mathbf{V}_{\mathbf{x}}[k] = \mathbf{V}_{\mathbf{x}}[k-1] = \mathbf{V}_{\mathbf{x}}[0]$. In analogy to the derivations under Section III-A, we may also state that $\mathbf{V}[0] = \mathbf{V}_{\mathbf{x}}[0]$, which means that the error covariance at the time we switch on the filter is equivalent to the state covariance in stationary mode. Replacing these relationsships into Equation 8, we finally obtain

$$\mathbf{AV}[0]\mathbf{A^t} - \mathbf{V}[0] = -\mathbf{G}\mathbf{\Psi_z}\mathbf{G^t} \tag{12}$$

as an equation for obtaining the initial error covariance matrix.

Again replacing the right side of the equation with $\mathbf{M}$, we may reformulate the problem as

$$\begin{array}{c}
[r(\mathbf{A},1)r^t(\mathbf{V},1)\,..\,r(\mathbf{A},1)r^t(\mathbf{V},n)]\cdot r^t(\mathbf{A},1) - V_{11} = M_{11}\\
[r(\mathbf{A},1)r^t(\mathbf{V},1)\,..\,r(\mathbf{A},1)r^t(\mathbf{V},n)]\cdot r^t(\mathbf{A},2) - V_{12} = M_{12}\\
...\\
[r(\mathbf{A},1)r^t(\mathbf{V},1)\,..\,r(\mathbf{A},1)r^t(\mathbf{V},n)]\cdot r^t(\mathbf{A},n) - V_{1n} = M_{1n}\\
[r(\mathbf{A},2)r^t(\mathbf{V},1)\,..\,r(\mathbf{A},2)r^t(\mathbf{V},n)]\cdot r^t(\mathbf{A},2) - V_{22} = M_{22}\\
...\\
[r(\mathbf{A},2)r^t(\mathbf{V},1)\,..\,r(\mathbf{A},2)r^t(\mathbf{V},n)]\cdot r^t(\mathbf{A},n) - V_{2n} = M_{2n}\\
...\\
[r(\mathbf{A},n)r^t(\mathbf{V},1)\,..\,r(\mathbf{A},n)r^t(\mathbf{V},n)]\cdot r^t(\mathbf{A},n) - V_{nn} = M_{nn}
\end{array}$$

Using the same terminology as under Section III-A, we may change the equations to

$$\begin{array}{c}
[\Sigma_{k=1}^n(\Sigma_{z=1}^n A_{iz}V_{kz})\cdot A_{jk}] - V_{ij} = M_{ij}\\
\Rightarrow \{[\Sigma_{k=1}^n(\Sigma_{z=1}^n A_{iz}f(k,z))\cdot A_{jk}] - f(i,j)\}\cdot\mathbf{v} = M_{ij}.
\end{array}$$

This finally leads to

$$\left[\begin{pmatrix}
\Sigma_{k=1}^n(\Sigma_{z=1}^n A_{1z}f(k,z))\cdot A_{1k}\\
\Sigma_{k=1}^n(\Sigma_{z=1}^n A_{1z}f(k,z))\cdot A_{2k}\\
...\\
\Sigma_{k=1}^n(\Sigma_{z=1}^n A_{1z}f(k,z))\cdot A_{nk}\\
\Sigma_{k=1}^n(\Sigma_{z=1}^n A_{2z}f(k,z))\cdot A_{2k}\\
...\\
\Sigma_{k=1}^n(\Sigma_{z=1}^n A_{2z}f(k,z))\cdot A_{nk}\\
...\\
\Sigma_{k=1}^n(\Sigma_{z=1}^n A_{nz}f(k,z))\cdot A_{nk}
\end{pmatrix} - \mathbf{I}_q\right]\cdot\mathbf{v} = \mathbf{m},$$

which is again a problem of the form $\mathbf{S}\cdot\mathbf{v} = \mathbf{m}$ and may be solved according to Equation 11. Another pseudocode for obtaining the exact initial error covariance in the discrete case is given with Algorithm 2.

Algorithm 2 Procedure for computing V(0) in function of A, G and Z in the discrete case.

$M = -G\cdot Z\cdot G^t$
$n =$ (dimension of A)
$q = \frac{n(n+1)}{2}$ ▷ number of unknowns
$H =$ (identity matrix of dimension q) ▷ H helps for f to get row-vectors
$index = 1$ ▷ create matrix S and vector m
for $i = 1$ to n **do**
  for $j = i$ to n **do**
    for $k = 1$ to n **do**
      $temp =$ (zero row vector of size q)
      for $z = 1$ to n **do**
        $p = 0$
        if $z \leq k$ **then**
          for $l = 1$ to $z-1$ **do**
            $p = p + n - l + 1$
          end for
          $p = p + k - (z-1)$
        else
          for $l = 1$ to $k-1$ **do**
            $p = p + n - l + 1$
          end for
          $p = p + z - (k-1)$
        end if
        $temp = temp + A_{i,z}\cdot$ (row p of H)
      end for
      (row $index$ of S) = (row $index$ of S) $+ A_{j,k}\cdot temp$
    end for
    $m_{index} = M_{i,j}$
    $index = index + 1$
  end for
end for
$S = S -$ (identity matrix of size q)

$v =$ (inverse of S) $\cdot\, m$ ▷ solve for v
$index = 1$ ▷ create $V(0)$
for $i = 1$ to n **do**
  for $j = i$ to n **do**
    $V_{i,j} = v_{\text{index}}$
    $V_{j,i} = v_{\text{index}}$
    $index = index + 1$
  end for
end for

C. Extension to non-linear systems

The extension to non-linear systems is strongly inspired from the EKF. The basic idea consists in simply replacing the constant system matrix $\mathbf{A}$ by a state depending matrix $\mathbf{A}(\mathbf{x}(e;t))$ that implements a linearization around the current system state. In the continuous case, Equation 10 then for instance changes to

$$\mathbf{A}(\mathbf{x}(0;e))\mathbf{V}(0) + \mathbf{V}(0)\mathbf{A^t}(\mathbf{x}(0;e)) = -\mathbf{G}\mathbf{\Psi_z}\mathbf{G^t}.$$

This however introduces a small problem regarding the determination of the initial error covariance matrix $\mathbf{V}(0)$ (or the covariance matrix at $t\to\infty$ for the real system), namely that we don't know the exact value of $\mathbf{x}(0;e)$ and are thus not able to compute the initial system matrix. The best we can do consists in replacing $\mathbf{A}(\mathbf{x}(0;e))$ by its expectation value. We then obtain

$$E\{\mathbf{A}(\mathbf{x}(0;e))\} = \mathbf{A}(E\{\mathbf{x}(0;e)\}) = \mathbf{A}(\mu_{\mathbf{x}}(0))$$

Using this system matrix provides a way to use all the previously established formulas in the non-linear case, too.

IV. Application to motion estimation

One problem arising when trying to evaluate the established algorithms is that its effects on the convergence of the filter can only be investigated in the statistical mean. This means that if we want to analyze the error of the state estimate, we do not only need to run experiments with ground-truth data, but also need to run a theoretically endless number of experiments with the same statistical starting conditions in order to verify the general convergence behavior of the filter. Therefore, the example presented in this paper consists of a simulation that might be repeated many times.

The example we are going to investigate is shown in Figure 3, a block between springs moving on a horizontal track. $x[k]$ designates the position, $a[k]$ the process noise (acceleration), and the system is assumed to be self-stable since losing energy due to friction between block and track. The friction force is proportional to the speed $v[k]$ (coefficient

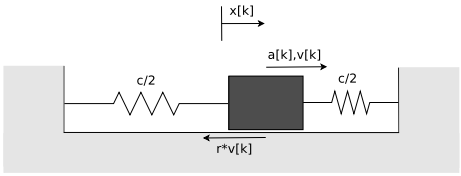

Fig. 3. Block between springs on horizontal track with friction.

r). The total spring constant is c. The state of our system is $\mathbf{x}[k] = (x[k] \quad v[k])^t$. T is the discrete time step. Hence the model of our system results to

$$\mathbf{x}[k] = \begin{pmatrix} 1 - \frac{T^2}{2}c & T - \frac{T^2}{2}r \\ -Tc & 1 - Tr \end{pmatrix} \cdot \mathbf{x}[k-1] + \begin{pmatrix} \frac{T^2}{2} \\ T \end{pmatrix} a[k]$$

$$\mathbf{y}[k] = \begin{pmatrix} 1 & 0 \end{pmatrix} \mathbf{x}[k] + \mathbf{r}[k]$$

For the experiment, $T = 0.01s$, $c = 0.2$ and $r = 0.8$. The process noise is defined with $\sigma_a = 0.8$ and $\Psi_a = \sigma_a^2$. Figure 4 shows one example for the evolution of the position during time. The simulation time is 20 s and the filter is enabled at $t = 10s$, which leaves enough time to the system to enter the stationary mode. The Kalman filter is able to safely estimate the state of our system despite of a noise variance of $\sigma_r = 0.03$.

With regard to the evaluation of the convergence behavior, it only makes sense to analyze the squared average error of position and speed since the initial error might have different sign depending on the random noise. The initial error covariance matrix found by our algorithm is $\begin{pmatrix} 0.02 & 0 \\ 0 & 0.004 \end{pmatrix}$. Different error covariances are obtained by multiplying the elements along the diagonal with a certain factor. For each initial error covariance matrix, the average is taken over 500 runs. Figure 5 shows the resulting estimation error. It shows clearly that the initial error covariance matrix computed by our formula results in best filter convergence performance

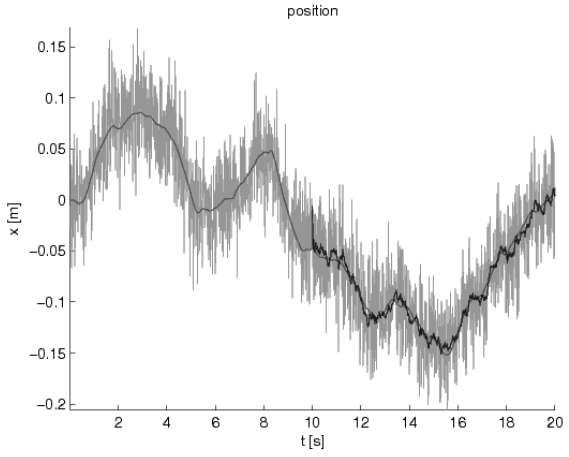

Fig. 4. Example for the position of the block (gray=groundtruth, brightgray=measurement, black=filter estimate).

(factor 1), both in position and speed. Setting the initial error covariances too high, even by orders of magnitude, does not have a huge influence on the convergence of the position. The influence on the speed is however noteable. Setting the initial error covariance only a bit too low directly results in slower convergence. It can be concluded that the convergence of the not directly measurable states is more critical. The inaccurate choice of the initial error covariance matrix like for instance $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ certainly has a strong negative influence on the convergence speed of the filter.

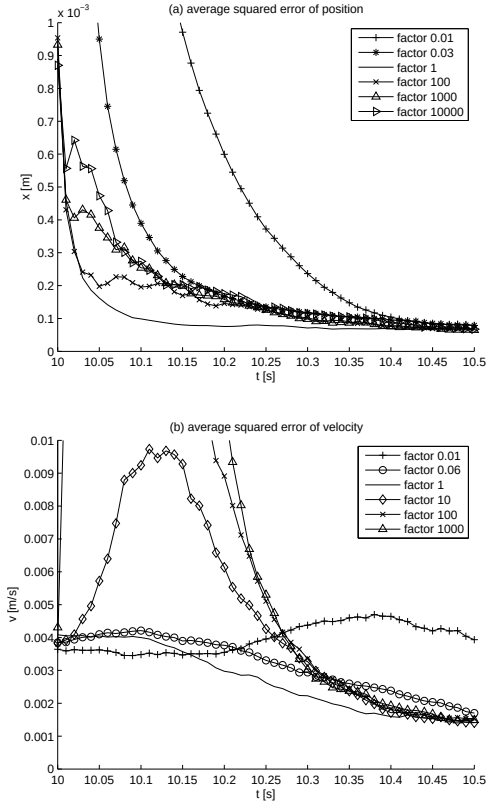

Fig. 5. Results of example 1 for different initial error covariance matrices each time averaged over 500 runs. The green curves show the convergence when the initial error covariance has been set too high, the red ones when it has been set too low and the blue one the one obtained using the principle depicted in this paper.

V. Conclusions and Outlook

This paper has shown the feasibility of deriving a correct initial error covariance matrix for statistical motion estimation filters. It could be shown that the specific case of a stationary system leads to analytic ways for computing the corresponding state covariance matrix, which is indeed equal to the error covariance matrix under the condition that some knowledge about the mean of the stationary state is given. The potential effects on the convergence of the filter estimate could be demonstrated in simulation, and it has been shown that if the intial assumptions are valid, applying the procedure presented in this paper basically leads to optimal convergence.

Future work now consists in evaluating the proposed procedures using real data with valid ground truth. The goal is to validate the application to an EKF fusion filter for MAV motion estimation. The challenge of generating statistical means might be overcome via logging longer datasets and engaging the filter offline at many different instants. Another certainly demanding aspect will be the extraction of all the model parameters for a given log, which obviously depends on several factors like for instance pilot skills and external conditions.

ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n. 231855 (sFly) and from the Swiss National Science Foundation under grant agreement n. 200021 125017/1.

REFERENCES

[1] E. Reich and P. Swerling. The detection of a sine wave in gaussian noise. *Journal of Applied Physics*, 1953.

[2] Steffen L. L. Time series analysis in 1880. a discussion of contributions made by T.N. Thiele. *International Statistical Review*, 49:319–333, 1981.

[3] Steffen L. L. *Thiele: Pioneer in Statistics*. Oxford University Press, 2002.

[4] Stratonovich R. L. Optimum nonlinear systems which bring about a separation of a signal with constant parameters from noise. *Radiofizika*, 2(6):892–901, 1959.

[5] Stratonovich R. L. Application of the markov processes theory to optimal filtering. *Radio Engineering and Electronic Physics*, 5(11):1–19, 1960.

[6] Kalman R. E. A new approach to linear filtering and prediction problems. *Transaction of the ASME, Journal of Basic Engineering*, pages 35–45, 1960.

[7] L. Armesto, J. Tornero, and M. Vincze. Fast ego-motion estimation with multi-rate fusion of inertial and vision. *Int. J. Rob. Res.*, 26(6):577–589, 2007.

[8] L. Armesto, S. Chroust, M. Vincze, and J. Tornero. Multi-rate fusion with vision and inertial sensors. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, volume 1, pages 193–199, April-1 May 2004.

[9] G. Nützi, S. Weiss, D. Scaramuzza, and R. Siegwart. Fusion of imu and vision for absolute scale estimation in monocular slam. In *International Conference on Unmanned Aerial Vehicles*, Dubai, 2010. Accepted for publication.

[10] J.J. Leonard and H.F. Durrant-Whyte. Simultaneous map building and localization for an autonomous mobile robot. pages 1442 –1447 vol.3, Nov 1991.

[11] A. Davison, D. Reid, D. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):1052–1067, 2007.

[12] Schlitt H. *Systhemtheorie für stochastische Prozesse*. Springer Verlag, 1992.