Using pseudo Kalman-filters in the presence of constraints application to sensing behaviors

Thierry Viéville, Peter Sander

UNITÉ DE RECHERCHE
INRIA-SOPHIA ANTIPOLIS

Rapport de Recherche

N°1669

Programme 4
Robotique, Image et Vision

Using pseudo Kalman-Filters in the presence of constraints Application to Sensing Behaviors

Thierry VIÉVILLE, Peter SANDER

Avril 1992

Institut National
de Recherche
en Informatique
et en Automatique

Sophia Antipolis
B.P. 109
06561 Valbonne Cedex
France
Tél.: 93 65 77 77

Using pseudo Kalman-Filters in the presence of constraints Application to Sensing Behaviors

Thierry VIÉVILLE, Peter SANDER
INRIA Sophia, BP109 06561 Valbonne, France
thierry@sophia.inria.fr

Abstract

A new generalization of the linear Kalman-Filter to non-linear equations is introduced. The deterministic interpretation of this mechanism is discussed. The proposed algorithm is an alternative to the well known Extended Kalman Filer. A small experimental illustration is given. An application to sensory-motor behaviors is proposed.

Utilisation de pseudos Filtres de Kalman en présence de contraintes Application aux Comportements Perceptifs

...

Abstract

Une nouvelle généralisation du formalisme du Filtre de Kalman Linéaire est introduite dans le cas de d'équations non-linéaires. L'interprétation déterministe, de ce mécanisme est discutée. L'algorithme proposé est une alternative au Filtre de Kalman Etendu. Une petite illustration expérimentale est donnée. Une application de ce formalisme aux comportements sensori-moteurs est proposée.

Contents

1 Introduction

The aim of a artificial vision systems is to build and update a 3D representation of the environment from a set of visual cues [43, 18, 1, 3, 17]. Let us assume visual primitives have been identified across an image sequence [16]. Such primitives are described using geometry and uncertainty [17, 22]. In order to convert these picture primitives into three-dimensional dynamic structures, complex algorithms of identification and classification are needed. A set of techniques - extensions of the Kalman Filter - are well adapted to solving most of these problems [4].

However implementations of these techniques (typically a Multiple-Model Approach of the Iterated Extended Kalman Filter [5]) are very costly and need a lot of analytic derivations before obtaining the equations of the final algorithm. In addition, some equations yield numerical unstabilities and the usual approximations are not sufficient [38], implying that other implementations have to be worked out [42].

In order to simplify the design of such algorithms, the problem of automatic code generation of Kalman Filters has already been addressed in the past (see [25], for instance), while software tools aimed at providing symbolic derivations of robotics models and computer vision applications are also under development [11, 40]. All the calculations are rather trivial (symbolic derivation of Jacobians, simplification of matrix expressions, computation of the sign of an expression). But one cannot use a computer algebra system to generate the program code, because a lot of dedicated tests have to be performed within the algorithm, and manually introduced.

Therefore, it is worthwhile to attempt to follow another approach, dealing with non-linear equations. One very powerful tool is to express the related equations through *constraints*. Using constraints, several kind of information such as qualitative variables, unitary vectors, homogeneous coordinates, compact representation for rotations [41], relations between variables, inequalities, can be introduced, in an *implicit* way [23, 4, 36, 37, 12, 9]. But, in that case, the modified Kalman Filter, or *pseudo Kalman Filter*, must manage such constraints.

We would like, in this paper, to derive such a new approach, in the non-linear case, in order to improve the efficiency and the quality of the knowledge representation used in vision algorithms, and also obtain algorithms which can be derived automatically using a computer algebra system.

We limit our discussion to the deterministic interpretation of the Kalman Filter whereas an approximate probabilistic interpretation is only given in an informal way.

What is this paper about

In the first section we briefly review the use of linear Kalman filters in vision, and the problems encountered when non-linear equations are used.

We then describe an alternative approach, called the *Pseudo Kalman Filter*, and explain how to use it in different situations.

In the third section we demonstrate some technical results about the convergence and the stability of the algorithm, and discuss how to implement it.

In the final section we describe a simple experimental result, and show its implementation in a computer algebra system.

The notations used for the Kalman Filter and a very short review of this mechanism are given in appendix A.

The main theorems and technical developments are given in appendix B.

An application of this framework to the modelizing of sensory-motor tasks, or sensing behaviors, is proposed in appendix C.

2 Linear and Non-linear parameter estimates

2.1 Using the Linear Kalman Filter in Vision

Let us consider the problem of estimating a parameter $\mathbf{x} \in \mathcal{R}^n$ from a set of measurements $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, ...\mathbf{y}_N\}$ with $\mathbf{y}_i \in \mathcal{R}^{q_i}$, $\mathcal{R}$ being the set of real numbers. Each measurement is related to the unknown parameter by a linear relation of the form: $\mathbf{y}_i = M_i \cdot \mathbf{x}$, M_i being a $q_i \times n$ matrix. This is illustrated in Fig.1. The solution of such a problem is obtained by solving a set of linear equations with respect to the components of $\mathbf{x}$.

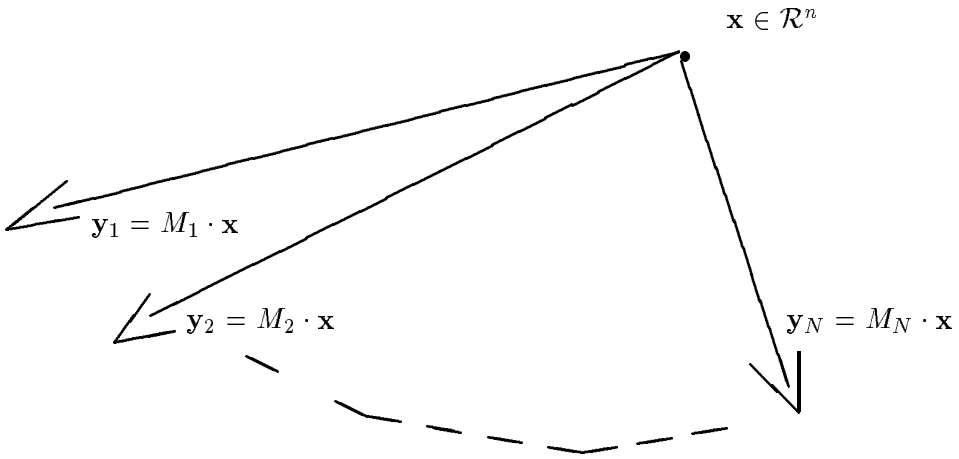

Figure 1: Estimating a parameter from a set of measures

In fact, this problem is -in practical situations- not that obvious and a lot of important aspects have to be taken into account:

- We might have **an uncertain initial guess** about the unknown parameter, with a "confidence neighbourhood" in which the parameter is supposed to be.
- Each **measurement is given with some uncertainty**, more precisely, we don't have a simple linear relation between measures and parameter, but an unknown quantity is added, thus perturbating the measure.
- The set of measurements does not necessarily provide a complete, unique estimate of the parameter, since **there might be redundancies** for some components, and no estimates for other components, the system being **not necessarily entirely observable**.
- We have to perform not only a *quantitative estimate* of the parameter, but also *qualitatively test hypotheses* during the process. Such tests are:
  - Is a given measure coherent with our knowledge about the parameter, or should this measurement be considered as erroneous and rejected (**data outlier**)?
  - Considering different relations between the measures and the parameter (**multiple-models**) , which one is to be chosen as the better model for this estimation?
  - Considering two set of measures related to a set of parameters, how to solve the correspondence problem (**data matching**)?
  - Considering two measures or two parameters, how to decide if they correspond to the same data, and in that case how to match them to obtain a better estimate (**data fusion**)?
- Parameters are not time-invariant but change with time and we have to estimate **the parameter evolution**, this internal model about the evolution being also approximate (**dynamic vision**).

- The system to be observed is not passive but at the sensor or preprocessing level, or at the mechanical level, there are some parameters to be tuned to obtain a better estimate or to perform a sensing behavior or perceptual task (**active vision**).

Kalman Filter based estimation techniques provide answers to these problems *in the linear case*. They are based on the notion of "uncertain knowledge" about a parameter, and weighted least-squares techniques. These techniques are not reviewed here, since they are so commonly described [5, 43, 17, 4]. We derive a set of equations in appendix A, in order to fix our notations.

Unfortunately, these methods are exact only in the linear case which is a too restrictive hypothesis. In addition their extension to the non-linear case leads to important problems as discussed in the next subsection.

However, in the non-linear case, all these considerations can be worked out using constraints, as already pointed out in several domains: Geometric Reasoning [23], Artificial Vision [4], Pattern Recognition [12], Control Theory [9], Robotics [36] or Biological modeling [37]. We try to implement this general idea in this paper.

2.2 The non-linear case : Indictments against the Extended Kalman Filter

In the non-linear case, the most common approach is to linearize the equations and to compute, locally, the estimates using the same kind of equations. This is called the Extended Kalman Filter, and we are now going to explain in detail why, *the Extended Kalman Filter approach is not to be used in every case in Vision*, cf [3]. The Extended Kalman Filter approach is, in fact, based on two ideas:

(1) The generalized Mahalanobis distance A reasonable criterion in order to estimate a vector $\mathbf{x}$ through a set of N vectorial measurement equations $\mathbf{f}(\mathbf{y}_i, \mathbf{x}) = 0, i = 1 \ldots N$ is to minimize the non-linear least-square criterion:

$$\Xi^2 = \frac{1}{2} \sum_{i=1}^{N} \mathbf{f}(\mathbf{y}_i, \mathbf{x})^t \Lambda_f(\mathbf{y}_i, \mathbf{x})^{-1} \mathbf{f}(\mathbf{y}_i, \mathbf{x})$$

which is a weighted least-square criterion, for which each measurement equation $\mathbf{f}(\mathbf{y}_i, \mathbf{x})$ is taken into account with a weight related to the covariance matrix $\Lambda_f(\mathbf{y}_i, \mathbf{x})$ of the given measurement.

Statistically this is related to an approximative *chi-square* test, and thus minimizing this criterion corresponds to minimizing the "statistical differences" between the different distributions.

If the errors are relatively small the covariance can be approximated using a first-order expansion, and we have:

$$\Lambda_f(\mathbf{y}_i, \mathbf{x}) \simeq \frac{d\mathbf{f}}{d\mathbf{y}_i} \Lambda_{\mathbf{y}_i} \frac{d\mathbf{f}}{d\mathbf{y}_i}^t$$

where $\Lambda_{\mathbf{y}_i}$ is the covariance of the ith measurement $\mathbf{y}_i$.

The choice of such a criterion is driven by statistical considerations, and is of common use in Vision. It is indeed possible to use it, even if the measurement equations are not linear.

(2) The Kalman recursive implementation Now, in the linear case, and *only in the linear case*, there exists an iterative way to compute the minimum of this criteria which corresponds to the *Mahalanobis* distance, between the distributions. This scheme is known as the Kalman filter and it processes each measurement once, one after another, the order of the measurements being without influence on the final result. This obviously dramatically reduces the number of operations needed in real-time implementations (however see Prop.2).

But what happens, in the non-linear case, if using such an implementation?

(a) Systematic errors in the recursive process The measurement equation is linearized around the original estimate of the parameter $\mathbf{x}$, and the Kalman equations are applied on the linearized equations. The process is repeated for each measurement. If this mechanism converges, the last measurement is linearized around the true solution while the first is not. In other words, the result depends on the way the measurements are put in sequence. And if there is a convergence, the criterion which has been minimized is not the original criterion Ξ^2 but a composite criterion where the error for each measurement is not estimated at the optimum, but around an intermediate value. Finally, we have determined experimentally that the modified criterion, based on the Extended Kalman Filter, might have a lot of local minimums related to minimizations based on only one measure, while the original criterion is convex [42].

(b) Numerical unstability Using linear techniques on non-linear equations [1], requires an approximate linearization of the system and an iterative use of the minimization techniques. Thus, since such equations are not well conditioned, any kind of approximation will introduce a bias which will not be cancelled by filtering, since this kind of error is systematic [42]. We then sometimes have to consider these equations without simplification, and avoid using approximate filters.

The worst case is when the linearization is very sensitive to errors, that is when the Jacobians are small. This is also quite a common situation. In addition, whereas in the case of tracking (steady-state), one can assume *a-priori* estimate to be close to the final solution since the variations might be relatively continuous, we cannot make the same assumption during the initialization of the estimate. All those algorithms have a *bootstrapping phase* at the beginning of the sequence. In the bootstrapping phase, there is no information about the parameter values, and the algorithm has to deal with initial estimates which are far from the final solution.

Problems with the manipulation of minimal representations In this formalism, visual object representation implies the choice of minimal regular geometric representations, the configuration space being a subset of a n-dimensional manifold without singular points. Hence, estimating the set of parameters defining an object is not a straightforward problem, whereas the following problems arise: (1) derivation of non-linear measurements equations, (2) computation of first-order approximations of quadratic forms over these equations in order to obtain an estimation of the covariances, (3) switch between the different maps of the manifold depending on the range of the parameters [3]. The derivation of the Kalman filter equations are thus very difficult to perform automatically, since a lot of tests have to be added to the algebraic equations, when switching from one map of the manifold to another.

Lack of theoretic results about convergence Moreover, there is no criterion concerning the convergence of such a method. In fact, in the linear case, the Kalman filter equations provide an explicit solution of the optimal stochastic estimator defined by the following recursive relationship on the conditional probability function (see for example [5], Chap. 3):

$$p\left[\mathbf{x}_{m+1} | \{\mathbf{y}_i\}_{i=1..m+1}\right] = \frac{1}{c} p\left[\mathbf{y}_{m+1} | \mathbf{x}_{m+1}\right] \int p\left[\mathbf{x}_{m+1} | \mathbf{x}_m\right] p\left[\mathbf{x}_m | \{\mathbf{y}_i\}_{i=1..m}\right] d\mathbf{x}_m$$

For a linear system, with Gaussian noise and initial state, the conditional probability function is completely determined by its conditional mean and covariance (sufficient statistic), and the previous equation reduces to the Kalman filter equations. For a linear system without the Gaussian noise hypothesis, the Kalman Filter is still somehow optimal, but among all linear operators only, while in the previous case optimality was obtained among all (linear and non-linear) operators.

[1] In Vision even with reasonable hypotheses such as considering small motions, and even if - in addition - the motion is assumed to be locally constant, some set of equations are not linear but at least quadratic [38]

In contrast with these precise results, the Extended Kalman Filter is not related to any result about convergence or optimality. This has been pointed out by several authors. Its experimental behavior might be bad, even in simple cases. More precisely, a recent study [27] - in the case of a simple structure from motion computation - showed that:
(1) the estimate of the conditional probability, as given above, requires an excessive amount of computation even in this simple case.
(2) the Extended Kalman Filter usually performs very badly and seriously underestimates covariances, thus assuming the data is much better estimated than it is, whereas the estimate is less well estimated that it is.
Thus, this is far from a good compromise.

Improvements of the Extended Kalman Filter

There indeed exists many heuristics to improve the original scheme: applying iteratively the Kalman filter equations on one measurement up to a local optimal, using second-order expansions instead of first-order ones, increasing covariances after each step in order to delay the convergence and to have a more homogeneous criterion, running twice the sequence of measurements in the direct and the reverse order, etc... But we think that there is a conceptual error here: the Extented Kalman Filter methods are dedicated to the treatment of an *on-line* causal sequence of temporal measurements, while we are in an *off-line* situation where the set of visual tokens to be analysed is not ordered by any causal or temporal order.

Anyway, if an automatic generation of algorithms is to be done, it is not so easy to introduce these heuristics and to manually adjust the algorithms to preserve convergence. It is then important to think about a new "simpler" method.

It has been also suggested to directly minimize the non-linear least-square criterion. There exists in fact very efficient algorithms dedicated to the minimization of such a non-linear least-square criterion [19]. Such numerical methods are now well known they are variations of the Gauss-Newton method, but take advantage of the particular nature of the criterion. The method ensures that steady progress is made whatever the starting point, and has the rapid ultimate convergence of Newton's method [19]. This has been implemented in vision [42], but it requires a lot of numeric calculations.

We finally need another definition of these estimators, clean enough to be studied and understood in theory, and for which criteria of convergence can be derived, but simple enough for the algorithms to be automatically produced by a computer algebra system.

This paper is an attempt to obtain such a compromise.

3 Pseudo Kalman Filter for time-invariant parameters

Let us assume we have a set of measurements $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, ...\mathbf{y}_N\}$ with $\mathbf{y}_i \in \mathcal{R}^{q_i}$, $\mathcal{R}$ being the set of real numbers. Each measurement $\mathbf{y}_i$ is given with a mean value $\bar{\mathbf{y}}_i$ and a covariance Λ_i.

Let us directly give the major ideas of this paper.

3.1 A new modification of the original Kalman Filter

Introducing constraints between parameters. For an object, the parameters are represented by a vector $\mathbf{x} \in \mathcal{R}^n$. Similarly to the linear case, $\mathbf{x}$ is given with a mean value or expectation $\bar{\mathbf{x}}$ and a covariance S. However, the key idea is the following: *we do not allow the vector $\bar{\mathbf{x}}$ to be everywhere in $\mathcal{R}^n$, but we consider state vectors belonging to a manifold, defined by p implicit equations $\mathbf{c}(\mathbf{x}) = 0_{\mathcal{R}^p}$.*

This definition is the result of a compromise: (α) On one hand, by having - *at first* - $\mathbf{x}$ belonging to $\mathcal{R}^n$, it is possible to preserve a simple statistical interpretation of the parameters as vectorial Gaussian random variables, with mean and covariance as sufficient statistic, and to

use linear estimators to compute $\mathbf{x}$. Of course any "abstract" manifold of dimension p, can be included in $\mathcal{R}^n$, with $n \geq 2p$ (Whitney Embedding Theorem) [35], in the very worst case, while in most cases $n = p + 1$ is sufficient. (β) By - *in a second step* - enforcing $\mathbf{x}$ to belong to a particular manifold, that is to satisfy a set of implicit equations, it is possible to restrict our estimate to a particular set of $\mathcal{R}^n$, corresponding to the parameter structure, for example simply considering algebraic manifolds (unary vectors, homogeneous coordinates, qualitative variables etc...).

Using linear measurement equations. Our second fundamental assumption is the following: *there is a linear relation between the measurements* $\mathbf{y}_i \in \mathcal{R}^{q_i}$ *and the state vector* $\mathbf{x} \in \mathcal{R}^n$*:* $\mathbf{y}_i = M_i \cdot \mathbf{x}$*,* M_i *being a* $q_i \times n$ *matrix.*

It is crucial to note that this assumption is **not** a restriction if $\mathbf{x}$ belongs to a manifold, while it was if $\mathbf{x}$ simply belongs to $\mathcal{R}^n$.

Proof Suppose this is not the case. Then, there is a non-linear relation between $\mathbf{y}_i$ and $\mathbf{x}$, let us say $f(\mathbf{x}, \mathbf{y}_i) = 0$, while $\mathbf{x}$ is given with constraints $\mathbf{c}(\mathbf{x}) = 0$. Now consider the new state vector $\tilde{\mathbf{x}}^T = (\mathbf{x}^T, v^T)$ with $\{f(\mathbf{x}, v) = 0, \mathbf{c}(\mathbf{x}) = 0\}$ as a constraint and $\mathbf{y}_i = v$ as the measurement equation. The vector $\tilde{\mathbf{x}}$ is, in fact, linearly related to the measure, and the non-linearity is now part of the constraints, as required **End Of Proof**

.

See also section 4.4 for a generalization.

Considering quadratic information for parameters Let us assume, we have an uncertain estimate, an "expectation" $\bar{\mathbf{x}}$ of the parameter $\mathbf{x}$. This means that we can provide an initial guess for $\mathbf{x}$, with a confidence neighbourhood $\mathcal{I}$.

In the linear case, the confidence neighbourhood is defined by a quadratic form in $\mathcal{R}^n$ written as:

$$\mathbf{x} \in \mathcal{I} \Leftrightarrow \Xi^2(\mathbf{x}) = (\mathbf{x} - \bar{\mathbf{x}})^T \cdot S^{-1} \cdot (\mathbf{x} - \bar{\mathbf{x}}) < \Xi_0^2$$

where S is a definite positive matrix [2].

In our case, this confidence neighbourhood we will consider the intersection between this ellipsoid centered on $\bar{\mathbf{x}}$, with $\mathbf{c}(\bar{\mathbf{x}}) = 0$, and the linear tangent space of the zero-manifold of the constraints. This is also a an ellipsoid.

We thus are now going to consider $\left(\bar{\mathbf{x}}, P_{\mathbf{x}} = S^{-1} = U^T \cdot D^{-1} \cdot U\right)$ as our *quadratic information* about $\mathbf{x}$ in $\mathcal{R}^n$, this information is of dimension $\frac{n(n+3)}{2}$, since the expectation $\bar{\mathbf{x}}$ is of dimension n, the quadratic errors on each eigen-axis in D are n values, while a rotation is a $n(n-1)/2$ manifold[3]. If nothing is known about $\mathbf{x}$, we simply have $P_{\mathbf{x}} = S^{-1} = 0$.

Minimizing a generalized Mahalanobis distance Following this, a reasonable estimate of $\mathbf{x}$, having a set of measurements $\mathbf{Y}$, and an *a priori* estimate $\mathbf{x}_0$ with covariance S_0 is:

$$\mathbf{x} = argmin_{\mathbf{x}} max_{\lambda} (\mathbf{x} - \mathbf{x}_0)^T S_0^{-1} (\mathbf{x} - \mathbf{x}_0) + \sum_{i=1}^{N} (\bar{\mathbf{y}}_i - M_i \cdot \mathbf{x})^T \cdot \Lambda_i^{-1} \cdot (\bar{\mathbf{y}}_i - M_i \cdot \mathbf{x}) + \lambda^T \cdot \mathbf{c}(\mathbf{x}) \quad (1)$$

where λ is a vector of Lagrange multipliers. The first term of this criterion measures the squared distance of $\mathbf{x}$ from an initial estimate, weighted by its covariance matrix, while the second term is nothing else but a weighted least-square criterion, and the last term is a set of constraints. With no constraint, this criterion is just the Mahalanobis distance between $\mathbf{x}$ and $\mathbf{Y}$. In that linear case, the Kalman filter, indeed, minimizes this criterion. In our case, when the criterion is given with constraints, we obtain a generalization of the original Kalman filter.

[2] **Interpreting the matrix** S

The matrix S has a *deterministic interpretation* as a "quadratic error matrix". In order to understand this point, let us consider the diagonal decomposition of S:

$$S = U^T \cdot D \cdot U = U^T \cdot \begin{pmatrix} V_1 & 0 & \dots & 0 \\ 0 & V_2 & \dots & 0 \\ & & \dots & \\ 0 & 0 & \dots & V_n \end{pmatrix} \cdot U$$

where U is an orthogonal matrix ($U^{-1} = U^T$), well defined since S is symmetric. Considering now the affine transformation $\mathbf{x}' = U \cdot (\mathbf{x} - \bar{\mathbf{x}})$ (precisely a translation $-\bar{\mathbf{x}}$ followed by a rotation U), we obtain: $\Xi^2(\mathbf{x}) = \sum_{i=1}^{n} \frac{(\mathbf{x}'_i)^2}{V_i}$, and it is now obvious that $\Xi^2(\mathbf{x})$ is the sum of the quadratic errors on each component of the parameter, weighted by a strictly positive factor. If $n = 1$, this corresponds to the notion of "confidence interval", since $\frac{(x-\bar{x})^2}{V} < 1$ if and only if $x \in [\bar{x} - \sqrt{V}, \bar{x} + \sqrt{V}]$.

This notion is now generalized for a vector and corresponds to the interior of an ellipsoid. The "size" of the neighbourhood is defined for each component with V_i, and the "geometry" of the neighbourhood is given by the directions of the n orthogonal axis of the ellipsoid, which correspond to the columns of the matrix U^{-1}.

Moreover there is a *probabilistic interpretation* of this quadratic form, $\bar{\mathbf{x}} = E[\mathbf{x}]$ being the mean or expectation of the quantity, and $S = E[(\mathbf{x} - \bar{\mathbf{x}})(\mathbf{x} - \bar{\mathbf{x}})^T]$ its covariance matrix. For instance, if $\mathbf{x}$ is considered as a Gaussian random variable, that is normally distributed with the density:

$$d(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^n det(S)}} e^{-\frac{1}{2}(\mathbf{x}-\bar{\mathbf{x}})^T \cdot S^{-1} \cdot (\mathbf{x}-\bar{\mathbf{x}})}$$

we precisely have: $\Xi^2(\mathbf{x}) = -2\log(d(\mathbf{x})) + constant$, such *minimizing* Ξ^2 *corresponds to take the parameter with the maximum probability*. In addition to this, considering the same diagonal decomposition as before, it is obvious that $\Xi^2(\mathbf{x})$ is the sum of independent, zero-mean, unity-variance Gaussian random variable, and it is said to have a *chi-square* distribution with n degrees of freedom.

Points outside the confidence neighbourhood ($\mathcal{H}_1$) are thus those which are significantly not equal to $\bar{\mathbf{x}}$, whereas points inside the confidence neighbourhood ($\mathcal{H}_0$), are -possibly- close to $\bar{\mathbf{x}}$, but either because they indeed are, or because the confidence neighbourhood is large.

[3] Remember that an orthogonal matrix is the exponential of an anti-symmetric matrix (Cayley formulas).

3.2 A first look at the main result

It will be demonstrated that, under weak assumptions, a simple iterative algorithm can be used to find the local minimum of the previous criterion. The equations for this algorithm are given now:

Proposition 1 *The solution of the least-square estimate given in equation (1) might be computed in two steps: First compute the solution of the unconstrained least-square estimate which is the solution of the following linear system:*

$$\begin{array}{rcl} S_1^{-1} & = & S_0^{-1} + \sum_{i=1}^{N} M_i^T \cdot \Lambda_i^{-1} \cdot M_i \\ \mathbf{x}_1 & = & S_1 \cdot (S_0^{-1} \cdot \mathbf{x}_0 + \sum_{i=1}^{N} M_i^T \cdot \Lambda_i^{-1} \cdot \bar{\mathbf{y}}_i) \end{array}$$

Second, compute the fixed point $\mathbf{x}^*$ *of the following series, which - if it converges - converges toward a solution of the problem:*

$$\mathbf{x}_{m+1} = \mathbf{x}_1 + S_1 \cdot \frac{\partial \mathbf{c}(\mathbf{x}_m)}{\partial \mathbf{x}}^T \cdot (\frac{\partial \mathbf{c}(\mathbf{x}_m)}{\partial \mathbf{x}} \cdot S_1 \cdot \frac{\partial \mathbf{c}(\mathbf{x}_m)}{\partial \mathbf{x}}^T)^{-1} \cdot (\frac{\partial \mathbf{c}(\mathbf{x}_m)}{\partial \mathbf{x}} \cdot (\mathbf{x}_m - \mathbf{x}_1) - \mathbf{c}(\mathbf{x}_m))$$

The covariance of the final estimate is:

$$S^* = S_1 - S_1 \cdot \frac{\partial \mathbf{c}(\mathbf{x}^*)}{\partial \mathbf{x}}^T \cdot \left(\frac{\partial \mathbf{c}(\mathbf{x}^*)}{\partial \mathbf{x}} \cdot S_1 \cdot \frac{\partial \mathbf{c}(\mathbf{x}^*)}{\partial \mathbf{x}}^T \right)^{-1} \cdot \frac{\partial \mathbf{c}(\mathbf{x}^*)}{\partial \mathbf{x}} S_1.$$

This series converges if and only if $\|\mathbf{c}(\mathbf{x}_m)\|$ *is strictly decreasing and the convergence is, under weak assumptions, quadratic.*
The proof will be given later.

There is a simple interpretation of the previous equations: $(\mathbf{x}_1, S_1)$ are the mean and covariance of the best estimate without constraint, while the recursive equations give the reprojection of $\mathbf{x}_1$ onto the manifold, defined by $\mathbf{c}()$. Thus, the basic idea is to first estimate the state in $\mathcal{R}^n$ using an exact linear Kalman filter and then to find the closest compatible estimate, with $\mathbf{c}(\mathbf{x}) = 0$.

A non-stationary state $\mathbf{x}$ is estimated as the solution of a Gauss-Markov state equation at each iteration, as in the standard Kalman Filter approach (see section 4.4).

Such an algorithm, even with this simple structure, is indeed rather costly to implement manually, since the underlying derivations might be long, and the related code is lengthy (typically a few thousands of operations), but is straightforward to program in a computer algebra system.

One can think about having a direct numerical implementation of the previous equations, with numerical estimation of the derivatives. However, such an implementation has two drawbacks: on one hand, because of numerical instabilities, an analytic form of the gradient $\frac{\partial \mathbf{c}(\mathbf{x})}{\partial \mathbf{x}}$ is required; on the other hand, this algorithm is convergent only if the manifold is not singular in a neighborhood of the solution, and except for special cases, this cannot be verified without the computation of the rank of $\frac{\partial \mathbf{c}(\mathbf{x})}{\partial \mathbf{x}}$, which is better done with symbolic methods [38].

3.3 Toward a theoretic justification for pseudo Kalman Filters

There is a theoretic construction related to the previous definitions. Although not always useful in practice, it is however important to understand which kind of constructions are related to our mechanism.

Let us work with $\mathcal{R}^n$ and the manifold $\mathcal{C}$, null-space of: $\mathbf{c}() : \mathcal{R}^n \to \mathcal{R}^p$, $p \leq n$, embedded in $\mathcal{R}^n$.

Consider a generic submersion $C^* : \mathcal{R}^n \to \mathcal{C}$ which performs a mapping of any vector $\mathbf{x}$ onto a vector of the manifold. This submersion is to be used to output a value on the manifold,

having an estimate in $\mathcal{R}^n$. It might be important, in real-time implementation to have such a function available[4].

The algorithm given in Prop.1 provides such a submersion from an optimal linear unconstrained estimate onto a point of the manifold. The first estimate is $\mathbf{x}_1$, and the second a vector $\mathbf{x}^*$, projected onto the manifold $\mathcal{C}$. This second vector is thus close to the first one, considering a particular metric, the covariance metric. In other words **we choose a vector on $\mathcal{C}$ for which the Euclidian** *Mahalanobis distance* **to the linear unconstrained estimate is minimum in $\mathcal{R}^n$.**

Consider now, the following relation between two vectors in $\mathcal{R}^n$:

$$\mathbf{x}' \leftrightarrow \mathbf{x}'' \Leftrightarrow C^*(\mathbf{x}') = C^*(\mathbf{x}'') = \mathbf{x}^*$$

This is an equivalence relation, and the parameters of an object or "state" is defined as an equivalence class for this relation. The final estimate of the algorithm is a representative of an equivalence class, and C^* being given, *any representative, including* $\mathbf{x}_1$ *might be used as internal representation of the state.* Statistical tests will always be valid for this linear estimate, since our computation is precisely the same as for a linear Kalman filter. In other words, we want to construct a σ-algebra for this quotient space for which the measurable subsets will be induced by the measurable subsets in $\mathcal{R}^n$.

More precisely, how to compute the covariance for the final estimate onto the manifold? We maintain two ideas. First, the covariance corresponds to a quadratic form, and in the case of a point onto a manifold, this tensorial object is to be defined in the tangent bundle of the manifold. Second, the quadratic form attached to $\mathbf{x}^*$ has to be related to the quadratic form attached to $\mathbf{x}_1$ by a projective operator related to C^*, since $C^*(\mathbf{x}_1) = \mathbf{x}^*$. This is given in the last formula of Prop.1. This two ideas bear out the fact we compute the covariance from a linear projector.

Where are, now, the theoretical drawbacks and approximations of such a method?

The principal drawback is that the dimension of the vector of parameters is higher than the real dimension of the state of the problem, inducing a increase in the size of the filter. Precisely, we give up the minimal state representation with all its efficiency and its complications. However embedding a manifold in a Euclidian space is usually not that complicated. Most of the time a p-dimensional manifold can be embedded in a $n = p + 1$ Euclidian space although general constructions requires at most $n = 2p$. Moreover this embedding has usually a "physical interpretation" since the Euclidian space $\mathcal{R}^n$ can be interpreted in the context of the problem.

The fact we use too many parameters has only one algorithmic limitation: while the Kalman Filter approach deal with partially observed systems (thus this is not a limitation during the linear filtering stage), we will see that during the reprojection, we theoretically need to have a complete estimate of the n-components of $\mathbf{x}$ with a finite precision. That is one need "more measures" or an initial guess in $\mathcal{R}^n$. This is not a real limitation, and in addition, one can also deal with such partially observed systems (see section 4.2).

The main approximation is, now, the following: *The covariance is computed in $\mathcal{R}^n$, and linearly reprojected onto the tangent space of the final estimate.*
This is an approximation, but rather natural: one expects the covariance to be decreased since the true set of parameters are "better known" (they are not everywhere in $\mathcal{R}^n$ but belongs to the manifold), by the introduction of constraints. By projecting we take only into account the covariance "parallel" to the manifold. Better approximations can be derived for special cases such as directional quantities [32, 29].

Finally, we preserved a simple statistical interpretation[5], not for the original non-linear parameters, but for a linear overdimensioned vector, representing the non-linear state of the system. In that case statistical tests as describe in the appendix A can be derived.

[4]For instance consider the constraint:$||\mathbf{x}|| = 1$. A natural submersion is then $C^*(\mathbf{x}) = \frac{\mathbf{x}}{||\mathbf{x}||}$ which relates any vector to a normalized vector, thus belonging to $\mathcal{C}$. It is possible for any estimate to output an estimate belonging to $\mathcal{C}$, even if the algorithm is still converging. This estimate is not the best one, but still usable.

[5]There is a way to construct a rigorous Stochastic Calculus in Manifolds [13] but such a formalism is far from being usable in applications.

4 Implementation of pseudo Kalman Filters

4.1 Implementation of the algorithm

We now would like to study how the algorithm given in Prop.1 can be established. We first are going to demonstrate and discuss the first step of this algorithm which is simply a re-implementation of the linear Kalman filter equations, we then are going to worked out the second step which is related to a quadratic optimization problem, as developed in the appendix B.

4.1.1 Linear estimate without constraint

Proposition 2 *The solution of equation (1) without constraints ($\lambda = 0$), is precisely given by the two first equations of Prop.1.*

The direct minimization of this criterion is faster than using the original Kalman filter equations, in a recursive manner.

Proof

The problem is:

$$\mathbf{x} = argmin_{\mathbf{x}} \left\{ J = (\mathbf{x} - \mathbf{x}_0)^T S_0^{-1} (\mathbf{x} - \mathbf{x}_0) + \sum_{i=1}^{N} (\bar{\mathbf{y}}_i - M_i \cdot \mathbf{x})^T \cdot \Lambda_i^{-1} \cdot (\bar{\mathbf{y}}_i - M_i \cdot \mathbf{x}) \right\}$$

This criterion is a quadratic positive criterion as the sum of quadratic positive criteria, because S_0 and Λ_i^{-1} are positive matrix. This criterion is a quadratic positive definite criterion because S_0 is quadratic positive definite. Thus, the solution of this problem is unique and given by the normal equations of this optimization problem.

Computing the derivative of J with respect to $\mathbf{x}$ yield the two first equations of Prop.1, as expected.

Now remind that multiplying a $(a \times b)$ matrix with a $(b \times c)$ matrix requires $o(abc)$ operations while inverting a $(a \times a)$ symmetric matrix requires about $o(1/6a^3)$ operations (QL decomposition).

Let us then solve the previous set of equation using the following algorithm:

$$\begin{array}{l} X = S_0^{-1} \\ y = S_0^{-1} \cdot \mathbf{x}_0 \\ for\ i = 1..N\ do\{ \\ \quad A = M_i^T \cdot \Lambda_i^{-1} \\ \quad X+ = X + A \cdot M_i \\ \quad y+ = A \cdot \bar{\mathbf{y}}_i \\ \} \\ \mathbf{x} = X^{-1} \cdot y \\ end \end{array}$$

which requires $o(n^2 + n + N(n + 2n^2(q+1)) + 1/3n^3) \simeq o(Nn^2q) + o(n^3)$ operations if the inverses of the covariances matrix are used as variables, while the original Kalman filter algorithm:

$$\begin{array}{l} S = S_0^{-1} \\ x = \mathbf{x}_0 \\ for\ i = 1..N\ do\{ \\ \quad K = S \cdot M_i^T \cdot (\Lambda_i + M_i \cdot S \cdot M_i^T)^{-1} \\ \quad S = (I - K \cdot M_i)S \\ \quad x = x + K \cdot (\bar{\mathbf{y}}_i - M_i \cdot x) \\ \} \\ end \end{array}$$

requires $o(N(q + n^2q + 3nq^2 + 1/6q^3 + 1/3n^3 + 2nq))$ for the best implementation.

It is easy to see the former is much faster (as soon as $N >> n$), and, in addition it should be less sensitive to rounding errors. **End Of Proof**

It might be surprising to have better performances with this simple method. Why such a situation? Because the original Kalman filter equations not only gives the best estimate of $\mathbf{x}$

after N measures but also the best estimate of $\mathbf{x}$ after $1, 2, ..N-1$ measures which is not to be used in our case.

4.1.2 Projection of the linear estimate

We want to find the point onto the state manifold, which is as close as possible to the linear estimate, considering the covariance metric. This corresponds to the following minimization problem: minimizing a quadratic form under regular constraints. A specific algorithm has been worked out and is given in details in the appendix B.

The link between both methods is the following: the unconstrained linear estimate being founded, the criterion given in equation (1) is now equivalent to:

$$\mathbf{x} = argmin_{\mathbf{x}} max_{\lambda} (\mathbf{x} - \mathbf{x}_1)^T S_1^{-1} (\mathbf{x} - \mathbf{x}_0) + \lambda^T \cdot \mathbf{c}(\mathbf{x})$$

This transformation is obvious since, in that case $\mathbf{x}_1$, indeed minimizes the previous criterion for $\lambda = 0$, S_1 being a definite positive matrix. This is pointed out at the beginning of the appendix A.

4.1.3 Projection of the covariance matrix

We finally have to compute the projection of the covariance, using the fact that the algorithm is a series of projection as shown by Prop.7. This is given by the next proposition:

Proposition 3 *The final estimate $\mathbf{x}^*$ is related to the unconstrained estimate by a linear projector. Computing the covariance of $\mathbf{x}^*$ from this linear projector yields the result of Prop.1.*

Proof Let us note $P^* = P(\mathbf{x}^*) = S_1 \cdot C_{\mathbf{X}}(\mathbf{x}^*)^T \cdot (C_{\mathbf{X}}(\mathbf{x}^*) \cdot S_1 \cdot C_{\mathbf{X}}(\mathbf{x}^*)^T)^{-1} \cdot C_{\mathbf{X}}(\mathbf{x}^*)$. Obviously $P^* \cdot P^* = P^*$ it is thus a projector.

Considering the equation $F(\mathbf{x}^*) = \mathbf{x}^*$ we obtain: $(I - P^*) \cdot (\mathbf{x}^* - \mathbf{x}_1) = \mathbf{z} = 0$, that is $(\mathbf{x}^* - \mathbf{x}_1)$ is in the image space of this projector.

Let us now write that the covariance of $\mathbf{z}$ is 0. We have expanding

$$E\left[((I - P^*) \cdot (\mathbf{x}^* - \mathbf{x}_1)) \cdot ((I - P^*) \cdot (\mathbf{x}^* - \mathbf{x}_1))^T\right] = 0$$

and performing a few algebra: $S^* - (I - P^*) \cdot S_1 = 0$, which is the expected result. **End Of Proof**

4.1.4 Conclusion

Combining results of Prop.2, Prop.4 and Prop.3 we obtain Prop.1. We have no evidence that the second step of the algorithm always converges, but it does in reasonable useful situations (Prop.12, Prop.13, Prop.14) and the convergences is, under week assumptions, quadratic and easy to verify (Prop.8).

4.2 Generalization to semi-observable states

If S_1^{-1} is a symmetric positive but not definite matrix, that is not invertible, the previous algorithm cannot be directly used. This corresponds to the fact that the state is not entirely observable.

It is however possible to overcome this difficulty, by the following method: let us note $\mathcal{Z}_{S_1^{-1}}$ the linear sub-space of $\mathcal{R}^n$, corresponding to the null-space of S_1^{-1}, and $\mathcal{Z}^{\perp}_{S_1^{-1}}$ the corresponding perpendicular subspace. Let us note z the dimension of $\mathcal{Z}_{S_1^{-1}}$, $0 \leq z \leq n$.

We decompose the vector $\mathbf{x}$ as $\mathbf{x} = \mathbf{x}_z + \mathbf{x}_{\perp}$, with $\mathbf{x}_z \in \mathcal{Z}_{S_1^{-1}}$ and $\mathbf{x}_{\perp} \in \mathcal{Z}^{\perp}_{S_1^{-1}}$.

Moreover the p constraints are to be written in the form $\mathbf{c}(\mathbf{x}_z, \mathbf{x}_{\perp})$ and from the implicit function theorem to be reduced into to set of $p-z$ and z constraints $\mathbf{c}_{p-z}(\mathbf{x}_z, \mathbf{x}_{\perp})$ and $\mathbf{c}_z(\mathbf{x}_z, \mathbf{x}_{\perp})$,

the latter being non-singular. In that case we can locally solve the second set of equations with respect to $\mathbf{x}_z = \mathbf{c}_z^{-1}(\mathbf{x}_\perp)$, and use as new constraint: $\mathbf{c}_{p-z}(\mathbf{c}_z^{-1}(\mathbf{x}_\perp), \mathbf{x}_\perp)$.

The quadratic form S_1^{-1} is definite on $\mathcal{Z}^\perp_{S_1^{-1}}$, and the algorithm can be applied on the modified problem.

4.3 Generalization to constraints with regular singularities

If $\mathbf{c}(\mathbf{x})$ is singular the proposed algorithm does not converge. This means that constraints are not independent, but that one equation depends upon another. In that case, compute the Jacobian of $\mathbf{c}(\mathbf{x})$, and its null space $\mathcal{Z}_\mathbf{c}$, will tell us in which subspace the constraints are singular, while they are regular in $\mathcal{Z}^\perp_\mathbf{c}$. Since this algorithm does not depend upon a linear transformation on the set of constraints, it is then possible to project the constraints in the subspace orthogonal to the null-space of the Jacobian, and obtain a new set of constraints, non singular. This mechanism is only valid for regular singularities.

If $\mathbf{c}(\mathbf{x})$ is only singular at the point corresponding to the solution, as for qualitative variables studied in Prop. 14, the algorithm still converges but has ti be stopped near the final solution.

4.4 Generalization to time-dependent problems

Let us now show that we can manage non-linear time-dependent problems with the same formalism.

Reduction of non-linear time-dependent problems Let us consider the following general non-linear model ($\mathbf{x}(t) \in \mathcal{R}_n$ and $\mathbf{y}(t) \in \mathcal{R}_q$):

$$\begin{array}{rcll} \mathbf{f}(\dot{\mathbf{x}}(t), \mathbf{x}(t), t) & = & 0_{\mathcal{R}^{n1}} & (evolution \quad equations) \\ \mathbf{g}(\mathbf{y}(t), \mathbf{x}(t), t) & = & 0_{\mathcal{R}^{n2}} & (measurement \quad equations) \\ \mathbf{h}(\mathbf{x}(t), t) & = & 0_{\mathcal{R}^{n3}} & (parameter \quad constraints) \end{array}$$

this is in fact the most general model in non-linear filtering [6]. It is possible to reduce this problem to a pseudo Kalman filter problem, in the following way:

Let us define a new state vector, with all implicit parameters:

$$\mathbf{X}(t) = \left| \begin{array}{c} \mathbf{x}(t) \\ \mathbf{u}(t) = \left| \begin{array}{c} \dot{\mathbf{x}}(t) \\ \mathbf{y}(t) \end{array} \right. \end{array} \right.$$

Let us consider the two linear constant relations:

$$\dot{\mathbf{x}}(t) = P \cdot \mathbf{u}(t) = \begin{pmatrix} I_{n\times n} & 0_{n\times q} \\ 0_{n\times n} & 0_{q\times q} \end{pmatrix} \cdot \mathbf{u}(t)$$

and:

$$\mathbf{y}(t) = M \cdot \mathbf{u}(t) = \begin{pmatrix} 0_{n\times n} & 0_{n\times q} \\ 0_{q\times n} & I_{q\times q} \end{pmatrix} \cdot \mathbf{u}(t)$$

which simply state the relations between the new state vector and its derivatives and the new state vector and the measures.

Let us define a new constraint which will be equivalent to the $\mathbf{f}$, $\mathbf{g}$ and $\mathbf{h}$ $n_1 + n_2 + n_3 = p$ equations:

$$\mathbf{c}(\mathbf{u}(t), \mathbf{x}(t), t) = \left| \begin{array}{c} f(P \cdot \mathbf{u}(t), \mathbf{x}(t), t) \\ g(M \cdot \mathbf{u}(t), \mathbf{x}(t), t) \\ \mathbf{h}(\mathbf{x}(t), t) \end{array} \right.$$

[6] We assume that $|\frac{d\mathbf{f}()}{d\dot{\mathbf{x}}}| \neq 0$, the implicit differential equation being a diffeomorphism for $\dot{\mathbf{x}}$, thus locally solvable.

We now have transformed the original non-linear problem on an almost linear problem: evolution equations and measurement equations are linear, while all non-linearities are related to the constraint.

Generalization to the continuous independent case We can distinguish two cases when considering time-dependent:

- The independent case: In that case the measurement equation, and the evolution equation are not to be considered at the same time. The system first predicts the evolution of the state vector, then corrects this estimate using the information from a new set of measures. This is the case when the measurement equation is not continuous but sampled, that is we measure a discrete sequence of values $\mathbf{y}_i = \mathbf{y}(t_i)$ but not a continuous process $\mathbf{y}(t)$, the evolution equation being either continuous or sampled [33].
- The inter-dependent case: In that case, both equations are to be considered at the same time.

See the appendix A for a discussion in the case of the linear Kalman Filter.

The first case is much simpler and the generalization of our formalism is straightforward. Assuming we have a discrete sequence of measures $\mathbf{y}_i = \mathbf{y}(t_i)$. Since we can separate the two problems, we obtain:

$$\left\{ \begin{array}{rcl} \dot{\mathbf{x}}(t) & = & P \cdot \mathbf{u}(t) \\ \mathbf{c}(\mathbf{u}(t), \mathbf{x}(t), t) & = & 0 \end{array} \right._{t \neq t_i} \quad and \quad \left\{ \begin{array}{rcl} \mathbf{y}(t_i) & = & M \cdot \mathbf{u}(t_i) \\ \mathbf{c}(\mathbf{u}(t_i), \mathbf{x}(t_i), t_i) & = & 0 \end{array} \right._{t = t_i}$$

the first set of equations being a differential system with constraints, the second set of equations corresponding to our problem. Between two measurement samples, the estimation is done using the first set of equations, at the occurrence of one sample, an *instantaneous* correction is made using the second set of equations.

Covariances are computed in the usual way, using the equations of reprojection, as given in Prop 1, and their evolution corresponds to the same differential equation as given in the appendix A. The structure of such an algorithm is very similar to the original Kalman Filter.

This situation is the most common, in vision: a continuous physical system under observation, using sampled measures.

Direct resolution in the continuous inter-dependent case What happens now if we can make the previous assumptions?

The estimation problem is now equivalent to the following minimization problem:

$$\begin{array}{l} \mathbf{x} = argmin_{\mathbf{x}} max_{\lambda,\mu} \Big\{ J = (\mathbf{x}(0) - \mathbf{x}_0)^T S_0^{-1} (\mathbf{x}(0) - \mathbf{x}_0) + \\ \quad \int \Big[(\bar{\mathbf{y}}(t) - M \cdot \mathbf{u}(t))^T \cdot \Lambda(t)^{-1} \cdot (\bar{\mathbf{y}}(t) - M \cdot \mathbf{u}(t)) \\ \quad + \mu^T \cdot \mathbf{c}(\mathbf{u}(t), \mathbf{x}(t), t) + \lambda^T \cdot (P \cdot \mathbf{u}(t) - \dot{\mathbf{x}}(t)) \Big] \, dt \Big\} \end{array}$$

We can write the related Hamiltonian for this dynamic system with constraints [10], which simply corresponds to use the following notations:

$$\begin{array}{rcl} H & = & L + \lambda^T \cdot P \cdot \mathbf{u}(t) + \mu^T \cdot \mathbf{c}(\mathbf{u}(t), \mathbf{x}(t), t) \\ & with & \\ L & = & (\bar{\mathbf{y}}(t) - M \cdot \mathbf{u}(t))^T \cdot \Lambda(t)^{-1} \cdot (\bar{\mathbf{y}}(t) - M \cdot \mathbf{u}(t)) \\ & and & \\ J & = & (\mathbf{x}(0) - \mathbf{x}_0)^T S_0^{-1} (\mathbf{x}(0) - \mathbf{x}_0) + \int \Big[H - \lambda^T \cdot \dot{\mathbf{x}}(t) \Big] \, dt \end{array}$$

while, integrating by part (see [10] for a complete development), we obtain:

$$J = (\mathbf{x}(0) - \mathbf{x}_0)^T S_0^{-1} (\mathbf{x}(0) - \mathbf{x}_0) + \int \left[H + \dot{\lambda}^T \cdot \mathbf{x}(t) \right] dt$$

The normal equations of this system are obtained computing $J_{\mathbf{u}} = 0$ and $J_{\mathbf{x}} = 0$, thus:

$$\begin{array}{rcccl} \dot{\lambda}^T & = & -H_{\mathbf{x}} & = & -\mu^T \cdot \mathbf{c}_{\mathbf{x}}(\mathbf{u}(t), \mathbf{x}(t), t) \\ 0 & = & H_{\mathbf{u}} & = & M^T \cdot \Lambda^{-1} \cdot (\bar{\mathbf{y}}(t) - M \cdot \mathbf{u}(t)) + \lambda^T \cdot P + \mu^T \cdot \mathbf{c}_{\mathbf{u}}(\mathbf{u}(t), \mathbf{x}(t), t) \end{array}$$

and the solution is obtained from two differential equations in $(\mathbf{x}, \lambda)$:

$$\begin{array}{rcl} \dot{\mathbf{x}}(t) & = & P \cdot \mathbf{u} \\ \dot{\lambda} & = & -\mathbf{c}_{\mathbf{x}}(\mathbf{u}(t), \mathbf{x}(t), t)^T \cdot \mu \end{array}$$

and two set of $n + q$ and p implicit equations in $(\mathbf{u}, \mu)$:

$$\begin{array}{rcl} M^T \cdot \Lambda^{-1} \cdot (\bar{\mathbf{y}}(t) - M \cdot \mathbf{u}(t)) + \lambda^T \cdot P + \mu^T \cdot \mathbf{c}_{\mathbf{u}}(\mathbf{u}(t), \mathbf{x}(t), t) & = & 0 \\ \mathbf{c}(\mathbf{u}(t), \mathbf{x}(t), t) & = & 0 \end{array}$$

This straightforward derivation is the standard way to solve such problem. It allows us to generalize the previous formalism to a time-dependent general Pseudo Kalman Filter. It has however some drawbacks: it is rather heavy (we have $3n + p + q$ parameters), the equation in λ has to be solved in the backward direction [10], and the implementation is not possible in real time.

But in most situation the method given in the previous paragraph is applicable.

5 A short experimental result

We now illustrate the previous discussion showing a single simple example, while most sophisticated related applications have been reported elsewhere [42], [41], [39]. This simple case has been already used to compare different estimators related to the Kalman Filter approach [27].

We assume having a set of 2D points $p_i = (x_i, y_i)$ in rotation in the plane, and a set of measurement $m_i = x_i / y_i$ of their perspective projections onto the x axis. Let us take $R = \begin{pmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{pmatrix}$ as the rotation matrix. We want to estimate θ.

This trivial measurement equation is indeed not linear in term of θ, but in term of $\mathbf{x}^T = (cos(\theta), sin(\theta))$. We then are conducted to estimate not θ but $\mathbf{x}$ with the obvious quadratic constraint $c = \mathbf{x}^T \cdot \mathbf{x} - 1 = 0$.

This algorithm has been automatically generated and results of the execution are given in Fig.2. The θ parameter has been estimated as $\theta = arctan(\mathbf{x}_2 / \mathbf{x}_1)$ and its variance computed using first order expansions.

Simulation has been done in the presence of an important additive Gaussian noise (standard deviation is twice the level of the measure), in order to observe the behavior of the algorithm.

The convergence of the algorithm is similar to what is observed for standard Kalman filters (for instance, in the absence of noise we obtain the convergence after two iterations, as expected). The constraint precision depends upon the variance of the parameter error in a coherent way : the less the variance, the more the constraint precision.

The algorithm is implemented in Maple in the following way: a Maple routine generates a C-code file in which a numeric procedure corresponds to one step of the algorithm. Then a C-program is to be made to call this routine recursively.

The derivation is in fact not straightforward and is done in several steps:

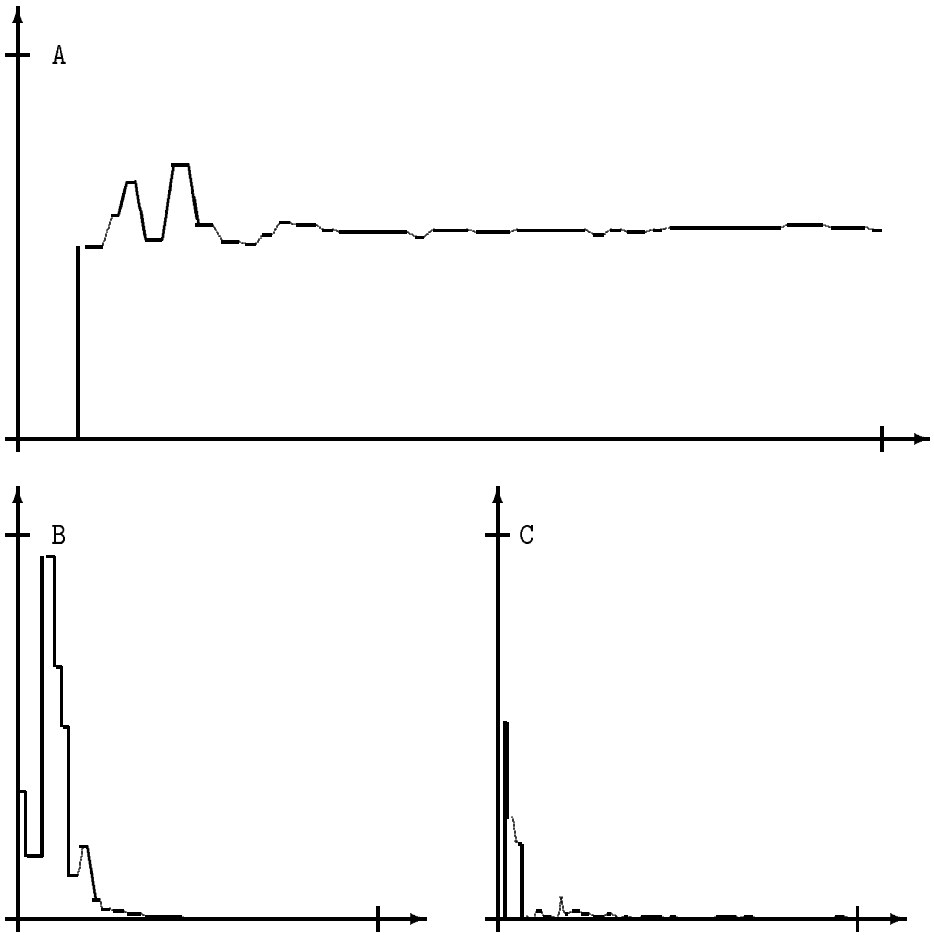

Figure 2: The behavior of a pseudo Kalman Filter for one parameter. A: estimation of θ, B: variance on the estimation of θ, C: evaluation of constraint precision.

1. The set of constraints is put in normal form and simplified. In fact complex set of constraints might be introduced, containing inequalities, boolean expressions of constraints:
   - Inequalities are converted into equalities using the formal sign function Sg() [40].
   - Boolean expressions of constraints are converted into algebraic expressions using standard methods.
   - Linear constraints are eliminated using change of coordinates.
   - Tautologies are eliminated.
   - Final expressions are simplified.

   .
2. The Jacobian $C_{\mathbf{x}}$ is calculated, simplified and its determinant computed to check singularities. If the expression involves symbolic derivation on matrix, the operations are carried out without using the components.
3. The true function $\mathbf{F}()$ is calculated, simplified and then approximated by a set of orthogonal polynomials in the domain of variation of $\mathbf{x}$. This special step is used for three main reasons: In order to preserve numerical stability even close to a singularity, in order to reduce the amount of operations, in order to obtain a computation only involving algebraic expressions thus not dependent upon special functions.
4. The linear part of the algorithm, the computation of $\mathbf{F}()$, and the covariance computation

are translated in C, using an internal C-library for matrix-calculations.

The syntax of the related Maple routine is given in the appendix D.

In order to insure a minimal set of operations we perform the simplifications of each expression with respect to side relations given by expressions computed previously. The result is an expression which is mathematically equivalent but which is in "normal form" with respect to the specified side relations. The specific meaning of "normal form" is determined by Grobner basis concepts. The Grobner basis for the side relations is first computed and then the value returned is used to obtain the fully reduced form of the expression to simplify with respect to the ideal basis founded. It yields a canonical form for polynomials modulo the ideal generated by the Grobner basis.

After this, code has been generated from the computer algebra system. Standard optimizations such as sub-expressions grouping, power to multiplication conversion, are automatically done.

6 Conclusion

We have described a formalism to derive and implement automatically object parameters estimation using a modified version a the original and powerful Kalman Filter. The computer implementation using a computer algebra system is reported elsewhere [40].

As it can be seen from several developments in the field the power of constraints is very high. It allows us to describe a lot of important features such as unitary vectors, rotations in $\mathcal{R}^3$, homogeneous coordinates, qualitative variables, multi-model approaches, etc..

However, comparing to semi-algebraic algorithms, these techniques have a very restrictive field of applications, they are **local**. They find an optimal estimate, but locally. They then do not have neither the algorithmic complexity, nor the potentiality of global algorithms as used in trajectory generation [36].

Although we had to develop quite a few "technical points" this study is not a real new theoretic development but a simple description of a "hack" to deal with non-linearities and numerical unstabilities in computer vision.

A Derivation of the linear Kalman Filter

Let us consider the problem of estimating a parameter $\mathbf{x} \in \mathcal{R}^n$ from a set of linear measurements $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, ...\mathbf{y}_N\}$ with $\mathbf{y}_i \in \mathcal{R}^{q_i}$, $\mathcal{R}$ being the set of real numbers [33]. Each measurement is related to the unknown parameter by a relation of the form: $\mathbf{y}_i = M_i \cdot \mathbf{x}$, M_i being a $q_i \times n$ matrix. This is illustrated in Fig.1.

Let us consider that there is additive noise in the linear relations between the measures $\bar{\mathbf{y}}_i$, and the parameter to be estimated $\mathbf{x}$: $\mathbf{y}_i = M_i \cdot \mathbf{x} + \nu_i$, ν_i being white Gaussian noise with zero-mean, covariance Λ_i.

Following the discussion given in section 2, a reasonable estimate of $\mathbf{x}$, having a set of measurements $\mathbf{Y}$, and an *a priori* estimate $\mathbf{x}_0$ with covariance S_0 is:

$$\mathbf{x}_N = argmin_{\mathbf{x}} \left\{ \Xi^2 = (\mathbf{x} - \mathbf{x}_0)^T S_0^{-1} (\mathbf{x} - \mathbf{x}_0) + \sum_{i=1}^{N} (\bar{\mathbf{y}}_i - M_i \cdot \mathbf{x})^T \cdot \Lambda_i^{-1} \cdot (\bar{\mathbf{y}}_i - M_i \cdot \mathbf{x}) \right\}$$

The first term of this criterion measures the squared distance of $\mathbf{x}$ from an initial estimate, weighted by its covariance matrix, while the second term is nothing else that a weighted least-square criterion. This criterion is just the Mahalanobis distance between $\mathbf{x}$ and the available data $(\mathbf{x}_0, \bar{\mathbf{y}}_1, \ldots, \bar{\mathbf{y}}_N)$, and the Kalman filter estimate corresponds to the minimum of this criterion.

A.1 Derivation of the solution for time-invariant parameters

The matrix associated with this quadratic criterion is positive definite, since it is the sum of definite positive matrices, and has thus only one minimum, which can be made explicit by rewriting Ξ^2 as:

$$\Xi^2 = (\mathbf{x} - S_N \cdot \mathbf{x}'_N)^T \cdot S_N^{-1} \cdot (\mathbf{x} - S_N \cdot \mathbf{x}'_N) + \Xi_N^2$$

with:

$$\begin{array}{rcl} S_N^{-1} & = & S_0^{-1} + \sum_{i=1}^{N} M_i^T \cdot \Lambda_i^{-1} \cdot M_i \\ \mathbf{x}'_N & = & S_0^{-1} \cdot \mathbf{x}_0 + \sum_{i=1}^{N} M_i^T \cdot \Lambda_i^{-1} \cdot \bar{\mathbf{y}}_i \end{array}$$

and the final solution is obviously $\mathbf{x}_N = S_N \cdot \mathbf{x}'_N$.

We also have:

$$\begin{array}{rcl} \Xi_N^2 & = & \mathbf{x}_0^T S_0^{-1} \mathbf{x}_0 + \sum_{i=1}^{N} \bar{\mathbf{y}}_i^T \cdot \Lambda_i^{-1} \cdot \bar{\mathbf{y}}_i - \mathbf{x}_N^T S_N^{-1} \mathbf{x}_N \\ & = & \mathbf{x}_0^T S_0^{-1} \mathbf{x}_0 + \mathbf{x}_N^T S_N^{-1} \mathbf{x}_N - 2\mathbf{x}_0^T S_0^{-1} \mathbf{x}_N \end{array} \quad (2)$$

which corresponds to the residual error at the optimum. The vector $\mathbf{x}'_N$ is a linear combination of the input data and its covariance is $E[\mathbf{x}'_N \mathbf{x}'^T_N] = S_N^{-1}$ as it can be directly computed, using linear formulas on quadratic forms [7] and since $(\mathbf{x}_0, \{\mathbf{y}_1..\mathbf{y}_N\})$ are independent.

The solution could also be computed from the normal equation:

$$\frac{1}{2} \frac{d\Xi^2}{d\mathbf{x}} (\mathbf{x}_N)^T = S_0^{-1} (\mathbf{x}_N - \mathbf{x}_0) + \sum_{i=1}^{N} M_i^T \cdot \Lambda_i^{-1} \cdot (\bar{\mathbf{y}}_i - M_i \cdot \mathbf{x}_N) = 0$$

The covariance of $\mathbf{x}_N$, that is the matrix used to estimate the uncertain knowledge, is just equal to S_N. It is simply obtained by the computation of $E[\mathbf{x}_N \mathbf{x}_N^T] = E[S_N \mathbf{x}'_N \mathbf{x}'^T_N S_N^T] = S_N E[\mathbf{x}'_N \mathbf{x}'^T_N] S_N^T = S_N$.

[7] If $Q_{\mathbf{x}}$ is the matrix of a quadratic form associated to the vector $\mathbf{x}$, $Q_{\mathbf{y}}$ is the matrix of a quadratic form associated to the vector $\mathbf{y}$, and P the matrix of a linear mapping, we have:

$$\begin{array}{rcl} Q_{\mathbf{x}+\mathbf{y}} & = & Q_{\mathbf{x}} + Q_{\mathbf{y}} \\ Q_{P \cdot \mathbf{x}} & = & P \cdot Q_{\mathbf{x}} \cdot P^T \end{array}$$

This criterion has quite a few probabilistic interpretations: it is an *unbiased, minimal variance estimator* among all linear estimators, in the general case [21]. Moreover, if the probabilistic distributions are Gaussian, it is a *general minimum variance estimator* among all estimators since the variance of $\mathbf{x}_N$ is minimal, it is also *maximum likehood estimator*, since the conditional probability $p(\mathbf{x}|\,(\mathbf{x}_0, \bar{\mathbf{y}}_1, \ldots, \bar{\mathbf{y}}_N))$ is maximum for $\mathbf{x}_N$, it corresponds to the *conditional expectation*, since $\mathbf{x}_N = E[\mathbf{x}|\,(\mathbf{x}_0, \bar{\mathbf{y}}_1, \ldots, \bar{\mathbf{y}}_N)]$, it minimizes *the sum of the Mahalanobis distances* between $\mathbf{x}$ and $(\mathbf{x}_0, \bar{\mathbf{y}}_1, \ldots, \bar{\mathbf{y}}_N)$.

In any case, it yields a set of linear equations, which relate the parameter to be estimated to the set of measurements and the initial guess, the linear relation being given by the matrix of uncertainty. This set of equations has different interpretation depending on the underlying assumptions, the most famous being a recursive implementation, but the solution is optimal in every case.

A.2 Performing tests on the estimates

Given a parameter $\mathbf{x}$, we have an uncertain estimate, an "expectation", $\bar{\mathbf{x}}$ of the parameter $\mathbf{x}$. This means that we can provide an initial guess for $\mathbf{x}$, with a confidence neighbourhood $\mathcal{I}$.

This confidence neighbourhood is defined by a quadratic form in $\mathcal{R}^n$ written as:

$$\mathbf{x} \in \mathcal{I} \Leftrightarrow \Xi^2(\mathbf{x}) = (\mathbf{x} - \bar{\mathbf{x}})^T \cdot S^{-1} \cdot (\mathbf{x} - \bar{\mathbf{x}}) < \Xi_0^2$$

where S is a positive definite matrix, the covariance of $\mathbf{x}$. This confidence neighbourhood is, indeed, an ellipsoid, centered on $\bar{\mathbf{x}}$.

In fact $\Xi^2(\mathbf{x})$ is small in two situations:

- If the error $\mathbf{x} - \bar{\mathbf{x}}$ is small, that is if the parameter is close to its expectation.
- If S^{-1} is small, that the estimate is not precise and one cannot determine if the error is high or not.

On the contrary $\Xi^2(\mathbf{x})$ is high if and only if **significantly not equal to** $\bar{\mathbf{x}}$.

Points outside the confidence neighbourhood ($\mathcal{H}_1$) are thus those which are significantly not equal to $\bar{\mathbf{x}}$, whereas points inside the confidence neighbourhood ($\mathcal{H}_0$), are -possibly- close to $\bar{\mathbf{x}}$, but either because they indeed are, or because the confidence neighbourhood is large.

It is thus possible to introduce some tests based on this distinction.

Rejection of erroneous measurements

Outlier measurements can be rejected *a posteriori*, by a statistical test *performed in the Euclidian space* $\mathcal{R}^{q_i}$:

$$\Xi_i^2 = (\bar{\mathbf{y}}_i - M_i \cdot \mathbf{x})^T \cdot \Lambda_i^{-1} \cdot (\bar{\mathbf{y}}_i - M_i \cdot \mathbf{x})$$

which has a $\mathcal{X}^2$ distribution with $n - p - q_i$ degrees of freedom, if there are p constraints between the $\mathbf{x}$ coordinates as discussed in this paper.

Validity of the estimate

In the probabilistic interpretation the residual error Ξ_N^2 (see equation (2)), can be related to a probability level, using a *chi-square test*, with $\sum q_i - n + p$ degrees of freedom, if there are p constraints between the $\mathbf{x}$ coordinates as discussed in this paper.

If the final Ξ_N^2 error is high, then the estimation is to be considered as suspicious, while if it is low the estimate is trustable.

Data matching and data fusion

Testing if two parameters are significantly different Considering two parameters $(\bar{\mathbf{x}}_1, \Lambda_1^{-1})$ and $(\bar{\mathbf{x}}_2, \Lambda_2^{-1})$, in the same coordinate frame, the quantity:

$$\Xi^2 = (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)^T \cdot (\Lambda_1 + \Lambda_2)^{-1} \cdot (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)$$

is the quadratic error of $\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2$, and is thus significantly different from zero if and only if differs $\bar{\mathbf{x}}_1$ from $\bar{\mathbf{x}}_2$. This distance is the Mahalanobis distance between $\bar{\mathbf{x}}_1$ and $\bar{\mathbf{x}}_2$, it has a $\mathcal{X}^2$ distribution, with $2(n-p)$ degrees of freedom, if there are p constraints between the $\mathbf{x}$ coordinates as discussed in this paper.

Fusing two parameters If the two parameters correspond to the same object it is then possible to perform a data fusion, the optimal value $(\bar{\mathbf{x}}_0, \Lambda_0^{-1})$ is given by the following formulas:

$$\begin{aligned} \Lambda_0^{-1} &= \Lambda_1^{-1} + \Lambda_2^{-1} \\ \Lambda_0^{-1} \cdot \bar{\mathbf{x}}_0 &= \Lambda_1^{-1} \cdot \bar{\mathbf{x}}_1 + \Lambda_2^{-1} \cdot \bar{\mathbf{x}}_2 \end{aligned}$$

If these parameters are subject to constraints, this linear estimate has then to be reprojected on the right manifold as discussed in this paper.

Comparing two confidence neighbourhood

It is also possible to compare two confidence regions $\Xi_a^2(\mathbf{x_a})$ and $\Xi_b^2(\mathbf{x_b})$ of two parameters $\mathbf{x_a}$ and $\mathbf{x_b}$, with dimensions n_a and n_b. Considering the ratio:

$$F = \frac{\Xi_a^2(\mathbf{x_a})/n_a}{\Xi_b^2(\mathbf{x_b})/n_b}$$

which will be high if and only parameter a confidence is lower than parameter b confidence. Again in the probabilistic interpretation of this ratio, it corresponds to a Fisher-Snedecor distribution, and it is possible, choosing a probability level, to decide for which value of this ratio, the previous test is significative.

Conclusion

Since the related quadratic form induces a norm $\mathcal{N}(\mathbf{x}, \Lambda_\mathbf{x})$ on the parameter space, it is possible to derive a distance between two parameters, called *the Mahalanobis distance*, equal the norm of the differences between two parameters.

The concept of uncertainty, not only provide a quantitative estimate about the parameter precision, but also allows to qualitatively test some hypotheses about the parameters.

A.3 Estimation in the dynamic case

Adding a new measurement

Suppose a new measurement $(\bar{\mathbf{y}}_{N+1}, \Lambda_{N+1})$ is available. There is no need to redo the computation, since the criterion is additive with respect to quadratic errors, and we easily obtain the following recursive equations:

$$\begin{aligned} S_{N+1}^{-1} \cdot \mathbf{x}_{N+1} &= \mathbf{x}'_{N+1} \\ (S_N^{-1} + M_{N+1}^T \cdot \Lambda_{N+1}^{-1} \cdot M_{N+1}) \cdot \mathbf{x}_{N+1} &= S_N^{-1} \cdot \mathbf{x}_N + M_{N+1}^T \cdot \Lambda_{N+1}^{-1} \cdot \bar{\mathbf{y}}_{N+1} \end{aligned}$$

which can also be written as:

$$\mathbf{x}_{N+1} = \mathbf{x}_N + K_{N+1} \cdot (\bar{\mathbf{y}}_{N+1} - M_{N+1}^T \cdot \mathbf{x}_N)$$

with

$$K_{N+1} = S_{N+1} \cdot M_{N+1}^T \cdot \Lambda_{N+1}^{-1} = S_N \cdot M_{N+1}^T \cdot \left(\Lambda_{N+1} + M_{N+1} \cdot S_N \cdot M_{N+1}^T\right)^{-1}$$

and

$$S_{N+1}^{-1} = S_N^{-1} + M_{N+1}^T \cdot \Lambda_{N+1}^{-1} \cdot M_{N+1} \Leftrightarrow S_{N+1} = S_N - K_{N+1} \cdot M_{N+1} \cdot S_N$$

using the "matrix-inverse lemma" which is a special case of Prop.5.

Computing the evolution of the parameter

Evolution might be modelized either using discrete or continuous equations.

Discrete evolution If the parameter is not a time-invariant quantity, it will change with time, using an affine uncertain model we can write:

$$\mathbf{x}(t+1) \leftarrow A(t) \cdot \mathbf{x}(t) + \mathbf{b}(t) + \xi(t)$$

where $A(t)$ is a deterministic matrix, $\mathbf{b}(t)$ a vector, and $\xi(t)$ an unknown or random parameter, with its quadratic error or covariance equal to $V(t)$.

In other words, we consider having a given knowledge about the evolution of $\mathbf{x}$, through $A(t)$ and $\mathbf{b}(t)$, but again with uncertainty through the quantity ξ.

In that case, we obtain a new estimate of $(\mathbf{x}_N, S_N)$ since:

$$\begin{aligned} \mathbf{x}_N(t+1) &= A(t) \cdot \mathbf{x_N}(t) + \mathbf{b}(t) \\ S_N(t+1) &= A(t) \cdot S_N(t) \cdot A(t)^T + V(t) \end{aligned}$$

Continuous evolution Similar formulas can be obtained if the evolution of $\mathbf{x}(t)$ is governed by a stochastic differential equation of the form:

$$\dot{\mathbf{x}}(t) = A(t) \cdot \mathbf{x}(t) + \mathbf{b}(t) + \xi(t)$$

$\xi(t)$ is a zero-mean white Gaussian noise of covariance $V(t)$.

In that case the estimate is also characterize by a (deterministic) differential equation:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= A \cdot \mathbf{x}(t) + \mathbf{b}(t) \\ \dot{S}(t) &= A(t) \cdot S(t) + S(t) \cdot A(t)^T + V(t) \end{aligned}$$

which is the continuous form of the previous equations.

Correction of a dynamical parameter

Two situations are to be taken into account.

Independent case If we assume that measurements occurring at a given time are related to *the instantaneous value of the parameter*, and if we assume that the *computation is done instantaneously*, the evolution of the parameter does not interfere with the occurrence of measurements. We thus can decoupled the two set of equations and the composite algorithm is simply to:

- Predict the new value of the parameter from the model, and modify and increase the quadratic error since the uncertainty about the parameter increases because of its variations,
- Correct the estimate of the parameter using new measurements.

In fact, these two implicit hypotheses are often used in the Kalman Filter approach.

Inter-dependent case If the previous hypotheses are no longer acceptable, we have to consider that a continuous flow of measurement is given:

$$\mathbf{y}(t) = M(t) \cdot \mathbf{x}(t) + \nu(t)$$

($\nu(t)$ is a zero-mean white Gaussian noise of covariance $\Lambda(t)$).

In that case the continuous form of the Kalman Filter is:

$$\begin{aligned}
\dot{\mathbf{x}}(t) &= [A \cdot \mathbf{x}(t) + \mathbf{b}(t)] + [K(t)\,(\mathbf{y}(t) - M(t) \cdot \mathbf{x}(t))] \\
K(t) &= S(t) \cdot M(t)^T \cdot \Lambda(t)^{-1} \\
\dot{S}(t) &= \left[A(t) \cdot S(t) + S(t) \cdot A(t)^T + V(t)\right] - \left[S(t) \cdot M(t)^T \cdot \Lambda(t)^{-1} \cdot M(t) \cdot S(t)\right]
\end{aligned}$$

and it just additively combines the prediction and the correction of the parameter at each instant.

A full demonstration of these equations can be found in [28], while a more informal but exhaustive development is given in [33].

B Quadratic minimization with regular constraints: an algorithm.

B.1 Introduction

Let us consider the following problem, with $\mathbf{x}_0 \in \mathcal{R}^n$, $\mathbf{c}() : \mathcal{R}^n \to \mathcal{R}^p$, $p \leq n$, and Q is a $n \times n$ positive definite matrix, defining a metric in $\mathcal{R}^n$:

$$\left\{ \begin{array}{c} min_{\mathbf{x}} \frac{1}{2}(\mathbf{x}-\mathbf{x}_0)^T \cdot Q \cdot (\mathbf{x}-\mathbf{x}_0) \\ \mathbf{c}(\mathbf{x}) = 0 \end{array} \right. \tag{3}$$

We consider $\mathbf{c}()$ is a $\mathcal{C}^1$ regular function almost everywhere.

Analytic versus geometric formulation This problem can be considered as an optimization problem, since we want to minimize a quadratic form in $\mathcal{R}^n$, while the solution must satisfy p equalities or constraints. This problem can be rewritten as:

$$\mathbf{x}^* = argmin_{\mathbf{x}} max_{\lambda} J = \left\{ \frac{1}{2}(\mathbf{x}-\mathbf{x}_0)^T \cdot Q \cdot (\mathbf{x}-\mathbf{x}_0) + \lambda^T \mathbf{c}(\mathbf{x}) \right\}$$

The vector λ is a set of Lagrange multipliers. The solutions of this problem, if any, are a sub-set of the solutions of the normal equations related to the minimization problem given in equation (3). These equations are obtained by derivating J with respect to $\mathbf{x}$ and λ. We have:

$$\left\{ \begin{array}{rcl} Q \cdot (\mathbf{x}-\mathbf{x}_0) + \frac{\partial \mathbf{c}}{\partial \mathbf{x}}(\mathbf{x})^T \cdot \lambda & = & 0 \\ \mathbf{c}(\mathbf{x}) & = & 0 \end{array} \right.$$

From a geometrical point of view this problem is very simple: If $\mathbf{c}()$ represent the implicit equations of a manifold $\mathcal{C}$ of dimension $n-p$, while Q defines a norm in $\mathcal{R}^n$, the point $\mathbf{x}^*$ is the point on $\mathcal{C}$, for which the Q-distance from $\mathbf{x}_0$ to $\mathcal{C}$ is minimal. The solution of equation (3) is thus related to the smallest distance between a point $\mathbf{x}_0$ and a manifold $\mathcal{C}$. Several theorems of Differential Geometry [35] provide us with properties about the existence and uniqueness of solutions. For instance, if $\mathcal{C}$ is convex, there is a unique solution, while if $\mathcal{C}$ is regular around $\mathbf{x}^*$ and $\mathbf{x}_0$ is sufficiently close to $\mathcal{C}$ there is also a unique solution.

However, we want to derive an iterative algorithm to compute $\mathbf{x}^*$, if it exists, and to detect an error, if the method does not converge.

B.2 Derivation of the algorithm

An iterative algorithm Here is a simple iterative algorithm which may be used to minimize the quadratic form with the related constraints. This algorithm is specific of the present optimization problem, whereas general but less specific approachs in the field are reviewed in [30].

Proposition 4 *If the following series converges, its limit is one of the solution of equation (3).*

$$\mathbf{x}_{m+1} = F(\mathbf{x}_m) = \mathbf{x}_0 + Q^{-1} \cdot \frac{\partial \mathbf{c}(\mathbf{x}_m)}{\partial \mathbf{x}}^T \cdot \left(\frac{\partial \mathbf{c}(\mathbf{x}_m)}{\partial \mathbf{x}} \cdot Q^{-1} \cdot \frac{\partial \mathbf{c}(\mathbf{x}_m)}{\partial \mathbf{x}}^T \right)^{-1} \cdot \left(\frac{\partial \mathbf{c}(\mathbf{x}_m)}{\partial \mathbf{x}} \cdot (\mathbf{x}_m - \mathbf{x}_0) - \mathbf{c}(\mathbf{x}_m) \right)$$

F is well defined if and only if the function $\mathbf{c}()$ is regular for each $\mathbf{x}_m$.

Proof Without loss of generality we can take $\mathbf{x}_0 = 0$, We just have to consider the series $\mathbf{x}'_m = \mathbf{x}_m - \mathbf{x}_0$ and $\mathbf{c}(\mathbf{x}'_m + \mathbf{x}_0))$ as constraints.

One can also write, if $C_{\mathbf{x}} = \frac{\partial \mathbf{c}}{\partial \mathbf{x}}(\mathbf{x})$ is a $p \times n$ matrix, the normal equations in this format:

$$\left\{ \begin{array}{rcl} Q \cdot \mathbf{x} + C_{\mathbf{x}}^T \cdot \lambda & = & 0 \\ C_{\mathbf{x}} \cdot \mathbf{x} & = & C_{\mathbf{x}} \cdot \mathbf{x} - \mathbf{c}(\mathbf{x}) \end{array} \right.$$

If $C_{\mathbf{x}}$ is of rank p, $\mathbf{c}()$ being regular, we can invert the matrix $\begin{pmatrix} Q & C_{\mathbf{x}}^T \\ C_{\mathbf{x}} & 0 \end{pmatrix}$ (see Prop.5) and obtain: $\lambda = -(C_{\mathbf{x}} \cdot Q^{-1} \cdot C_{\mathbf{x}}^T)^{-1} \cdot (C_{\mathbf{x}} \cdot \mathbf{x} - \mathbf{c}(\mathbf{x}))$, from the system:

$$\begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix} = \begin{pmatrix} Q & C_{\mathbf{x}}^T \\ C_{\mathbf{x}} & 0 \end{pmatrix}^{-1} \cdot \begin{pmatrix} 0 \\ C_{\mathbf{x}} \cdot \mathbf{x} - \mathbf{c}(\mathbf{x}) \end{pmatrix} \tag{4}$$

Using the first normal equation we also have: $\mathbf{x} = -Q^{-1} \cdot C_{\mathbf{x}}^T \cdot \lambda$, and the combination of both equations gives an equation for $\mathbf{x}$, in which the Lagrange multipliers λ are eliminated.

By translating the result from 0 to $\mathbf{x}_0$ we obtain the equation $x = F(x)$ which is a necessary condition for the series to converge.

F is well defined if and only if $(\frac{\partial \mathbf{c}(\mathbf{x})}{\partial \mathbf{x}} \cdot Q^{-1} \cdot \frac{\partial \mathbf{c}(\mathbf{x})}{\partial \mathbf{x}}^T)$ is invertible that is if $\frac{\partial \mathbf{c}(\mathbf{x})}{\partial \mathbf{x}}$ is of rank p, (see Prop.6) which is just the condition for $\mathbf{c}()$ to be regular. **End Of Proof**

In the last demonstration we have made use of the following two results, the former being, in fact, related to *matrix-inversion-lemma*, used in the Kalman Filter development:

Proposition 5 *If Q is a $n \times n$ symmetric positive definite matrix and C is $p \times n$ matrix of rank $p \leq n$ the block matrix $M = \begin{pmatrix} Q & C^T \\ C & 0 \end{pmatrix}$ is symmetric positive definite with :*

$$N = \begin{pmatrix} Q^{-1} - Q^{-1}(C^T \cdot W \cdot C)Q^{-1} & V^T \\ V = W \cdot C \cdot Q^{-1} & -(W = (C \cdot Q^{-1} \cdot C^T)^{-1}) \end{pmatrix}$$

as inverse.

Proof Computing formally $M \cdot N$ gives the result, since the inverse of a symmetric positive definite matrix is also symmetric positive definite. However N is well defined if and only if W is well defined, that is if and only if $C \cdot Q^{-1} \cdot C^T$ is invertible, all other terms of N being well defined. This is shown in the next proposition. **End Of Proof**

.

Proposition 6 *If Q is a $n \times n$ symmetric positive definite matrix and C is $p \times n$ matrix of rank $p \leq n$ the matrix $P = C \cdot Q^{-1} \cdot C^T$ is symmetric positive definite, thus invertible.*

Proof First note that the inverse of symmetric positive definite matrix is also symmetric positive and definite. Then, computes P^T to show it is symmetric. It is also positive since: $\mathbf{x}^T \cdot P \cdot \mathbf{x} = (C^T \cdot \mathbf{x})^T \cdot Q^{-1} \cdot (C^T \cdot \mathbf{x}) \geq 0$ because Q^{-1} is positive. In addition P is definite since: $\mathbf{x}^T \cdot P \cdot \mathbf{x} = (C^T \cdot \mathbf{x})^T \cdot Q^{-1} \cdot (C^T \cdot \mathbf{x}) = 0$ if and only if $C^T \cdot \mathbf{x} = 0$ because Q^{-1} is definite and if and only if $\mathbf{x} = 0$ because C of rank $p \leq n$. **End Of Proof**

Geometric interpretation Although the previous mechanism has been derived from studying the problem as an optimization problem, there is, in fact, a precise geometrical interpretation of this algorithm:

Proposition 7 *The vector $F(\mathbf{x})$ is the result of a non-orthogonal projection in $\mathcal{R}^n$ onto the sub-space orthogonal to $\frac{\partial \mathbf{c}(\mathbf{x})}{\partial \mathbf{x}}$ in a direction parallel to $Q^{-1} \cdot \frac{\partial \mathbf{c}(\mathbf{x})}{\partial \mathbf{x}}^T$.*

Proof We still consider $\mathbf{x}_0$ as the origin.

First note that:

$$C_{\mathbf{x}} \cdot \mathbf{F}(\mathbf{x}) = C_{\mathbf{x}} \cdot \mathbf{x} - \mathbf{c}(\mathbf{x}) \tag{5}$$

as it can be derived from the definition of $\mathbf{F}()$. From equation (5) we can also derive the following identity:

$$Q^{-1} \cdot C_{\mathbf{x}}^T \cdot (C_{\mathbf{x}} Q^{-1} C_{\mathbf{x}}^T)^{-1} \cdot C_{\mathbf{x}} \cdot \mathbf{F}(\mathbf{x}) = \mathbf{F}(\mathbf{x}) \tag{6}$$

and we will note $P(\mathbf{x}) = Q^{-1} \cdot C_{\mathbf{x}}^T \cdot (C_{\mathbf{x}} Q^{-1} C_{\mathbf{x}}^T)^{-1} \cdot C_{\mathbf{x}}$.

Consider now the more general matrix $P = Q^{-1} \cdot C_1^T \cdot (C_0 \cdot Q^{-1} \cdot C_1^T)^{-1} \cdot C_0$ with $C_1 = C_0 = \frac{\partial \mathbf{c}(\mathbf{x})}{\partial \mathbf{x}}$ in our case. P is a projector since $P \cdot P = P$. Its is known that a projector P projects vectors in

the direction of its null-space $Ker(P)$ onto a sub-space of invariant vectors for P, the range of P. But $Ker(P) = Ker(C_0)$ because $C_0 \cdot \mathbf{x} = 0 \Rightarrow P \cdot \mathbf{x} = (Q^{-1} \cdot C_1^T \cdot (C_0 Q^{-1} C_1^T)^{-1}) \cdot (C_0 \cdot \mathbf{x}) = 0$ whereas $P \cdot \mathbf{x} = 0 \Rightarrow C_0 \cdot \mathbf{x} = 0$ since $Q^{-1} \cdot C_1^T \cdot (C_0 Q^{-1} C_1^T)^{-1}$ is of rank p from the hypothesis. In addition, consider a vector $\mathbf{x} = Q^{-1} C_1^T \mathbf{y}$, that is a vector parallel to $Q^{-1} \cdot C_1^T$. One can easily verify that $P \cdot \mathbf{x} = \mathbf{x}$ whereas if $P \cdot \mathbf{x} = \mathbf{x}$ we have $\mathbf{x} = Q^{-1} C_1^T \mathbf{y}$ for $\mathbf{y} = (C_0 Q^{-1} C_1^T)^{-1} \cdot C_0 \cdot \mathbf{x}$, that is those vectors are just the invariant vectors for P. This shows that P is a projector onto the sub-space orthogonal to C_0 in a direction parallel to $Q^{-1} \cdot C_1^T$.

Now using equation (6), we can verify that $P \cdot \mathbf{F}(\mathbf{x}) = \mathbf{F}(\mathbf{x})$, which is the result to be established. **End Of Proof**

B.3 Convergence and Stability

Study of the convergence Finding criteria of convergence for this algorithm is trivial, since the algorithm is just the iterative computation of a series. In particular, if $\|\frac{\partial \mathbf{F}(\mathbf{x})}{\partial \mathbf{x}}\| < 1$, the function is a contractive mapping and the series converges. Another - costless - criterion might be to verify that $\|\mathbf{c}(\mathbf{x}_{m+1})\| = \|\mathbf{c}(\mathbf{F}(\mathbf{x}_m))\| < \|\mathbf{c}(\mathbf{x}_m)\|$ related to the fact that $lim_{n\to\infty} \mathbf{c}(\mathbf{x}_m) = 0$ is expected. However this is not trivial, because the series might be divergent, but having $\|\mathbf{c}(\mathbf{x}_m)\| = 0$. This corresponds to the situation where the series get closer to $\mathcal{C}$ but not converge in it. Hopefully, this is not the case.

More precisely we have the following result:

Proposition 8 *The series $\mathbf{x}_{m+1} = \mathbf{F}(\mathbf{x}_m)$ is convergent if and only if $u_m = \|\mathbf{c}(\mathbf{x}_m)\|$ converges to 0 .*

Let $\mathcal{U}$ be a convex, compact, subset of $\mathcal{R}^n$ on which $\mathbf{c}()$ is $\mathcal{C}^2$.

If we can guaranty that the second order derivatives of $\mathbf{c}()$ are bounded on $\mathcal{U}$ this is a 2nd order method.

If we only can guaranty the first order derivatives of $\mathbf{c}()$ are bounded in a subset of $\mathcal{R}^n$, this is a first order method.

Proof We again consider $\mathbf{x}_0 = 0$, for the demonstration.

If the series $\mathbf{x}_{m+1} = \mathbf{F}(\mathbf{x}_m)$ converges to a limit $\mathbf{x}^*$ then the series $\mathbf{c}(\mathbf{x}_m)$ indeed converges by construction and we have $\mathbf{c}(\mathbf{x}^*) = C_{\mathbf{x}} \cdot (\mathbf{x}^* - \mathbf{F}(\mathbf{x}^*)) = 0$.

Reciprocally, assuming that $lim_{n\to\infty} \mathbf{c}(\mathbf{x}_m) = 0$, and the definition of $\mathbf{F}()$, we have the following equation on limits:

$$lim_{n\to\infty} \left(\mathbf{F}(\mathbf{x}_m) - Q^{-1} \cdot C_{\mathbf{x}}(\mathbf{x}_m)^T \cdot (C_{\mathbf{x}}(\mathbf{x}_m) Q^{-1} C_{\mathbf{x}}(\mathbf{x}_m)^T)^{-1} \cdot C_{\mathbf{x}}(\mathbf{x}_m) \cdot \mathbf{x}_m\right) = lim_{n\to\infty} Q^{-1} \cdot C_{\mathbf{x}}(\mathbf{x}_m)^T \cdot (C_{\mathbf{x}}(\mathbf{x}_m) Q^{-1} C_{\mathbf{x}}(\mathbf{x}_m)^T)^{-1} \cdot \mathbf{c}(\mathbf{x}_m) = 0$$

Let us note again $P(\mathbf{x}_m) = Q^{-1} \cdot C_{\mathbf{x}}(\mathbf{x}_m)^T \cdot (C_{\mathbf{x}}(\mathbf{x}_m) Q^{-1} C_{\mathbf{x}}(\mathbf{x}_m)^T)^{-1} \cdot C_{\mathbf{x}}(\mathbf{x}_m)$. The previous equations means that the series $\mathbf{x}_{m+1} = \mathbf{F}(\mathbf{x}_m)$ converges toward the series $\mathbf{x}_{m+1} = P(\mathbf{x}_m) \cdot \mathbf{x}_m$, if this latter converges and has the limit. But the latter is a series build on the composition of projectors and is convergent (see Prop.9). Then the series $\mathbf{x}_{m+1} = \mathbf{F}(\mathbf{x}_m)$ is thus convergent, which demonstrates the first part of the proposition.

As a consequence, because $\mathcal{U}$ is compact, the convergences of the two series are absolute convergence, and one can write, for some μ:

$$\forall m,\ \|\mathbf{x}_{m+1} - \mathbf{x}_m\| < \mu \|\mathbf{c}(\mathbf{x}_m))\|$$

which will be used now.

Consider now the function $\Phi : [0,1] \to \mathcal{R}^{p\times n}$ given by $\Phi(t) = G(\mathbf{y} + t(\mathbf{y} - \mathbf{x}))$, where $G : \mathcal{R}^n \to \mathcal{R}^{p\times n}$ is $\mathcal{C}^1$ in $\mathcal{U}$. The function $\Phi()$ is well defined because $\mathcal{U}$ is convex. It is differentiable for all $0 \le t \le 1$ where $\mathbf{x}, \mathbf{y} \in \mathcal{U}$. We can apply the finite increment theorem for Φ and state that there exists a constant t_0 such that $\Phi(1) - \Phi(0) = \Phi(t_0)'$. Replacing $\Phi()$ by its value and considering $\mathbf{y}_0 = \mathbf{y} + t_0(\mathbf{y} - \mathbf{x})$ we obtain $G(\mathbf{y}) - G(\mathbf{x}) = \frac{\partial \mathbf{G}(\mathbf{y}_0)}{\partial \mathbf{x}} \cdot (\mathbf{y} - \mathbf{x})$.

Applying the previous formula for $G_{\mathbf{x}}(\mathbf{y}) = \mathbf{c}(\mathbf{y}) - (\mathbf{c}(\mathbf{x}) + C_{\mathbf{x}} \cdot (\mathbf{y} - \mathbf{x}))$, with $G(\mathbf{x}) = 0$ we have $G(\mathbf{y}) = 1/2 \sum_{i=1}^n \mathbf{e}_i (\mathbf{y} - \mathbf{x})^T \cdot C^i_{\mathbf{xx}}(\mathbf{y}_0) \cdot (\mathbf{y} - \mathbf{x})$ where $\mathbf{e}_i$ is the ith Cartesian basis vector, $C_{\mathbf{x}}$ thus Jacobian of $\mathbf{c}()$ evaluated at point $\mathbf{x}$ and similarly $C^i_{\mathbf{xx}}$ is the Hessian of the ith component of $\mathbf{c}()$.

Using this, we obtain an a exact Taylor expansion of $\mathbf{c}(\mathbf{y})$ with second order terms:

$$\mathbf{c}(\mathbf{y}) = \mathbf{c}(\mathbf{x}) + C_{\mathbf{X}} \cdot (\mathbf{y} - \mathbf{x}) + 1/2 \sum_{i=1}^{n} \mathbf{e}_i (\mathbf{y} - \mathbf{x})^T \cdot C^i_{\mathbf{XX}}(\mathbf{y}_0) \cdot (\mathbf{y} - \mathbf{x})$$

We finally obtain, using equation (5), $\mathbf{c}(\mathbf{F}(\mathbf{x})) = 1/2 \sum_{i=1}^{n} \mathbf{e}_i (\mathbf{F}(\mathbf{x}) - \mathbf{x})^T \cdot C^i_{\mathbf{XX}}(\mathbf{y}_0) \cdot (\mathbf{F}(\mathbf{x}) - \mathbf{x})$ and since the second order derivatives of $\mathbf{c}()$ are bounded, let us say $\|C^i_{\mathbf{XX}}\| < \gamma$, we have:

$$\|\mathbf{c}(\mathbf{x}_{m+1}))\| < n\gamma \|\mathbf{x}_{m+1} - \mathbf{x}_m\|^2 < n\gamma\mu^2 \|\mathbf{c}(\mathbf{x}_m))\|^2$$

which demonstrates the second part of the proposition.

Moreover, since $C_{\mathbf{X}} \cdot \mathbf{F}(\mathbf{x}) = C_{\mathbf{X}} \cdot \mathbf{x} - \mathbf{c}(\mathbf{x})$, we also have $\|\mathbf{c}(\mathbf{x})\| = \|C_{\mathbf{X}} \cdot (\mathbf{F}(\mathbf{x}) - \mathbf{x})\| \leq \|C_{\mathbf{X}}\| \|\mathbf{F}(\mathbf{x}) - \mathbf{x}\|$ that is if $\|C_{\mathbf{X}}\|$ is bounded by, let us say, α:

$$\|\mathbf{c}(\mathbf{x}_m)\| < \alpha \|\mathbf{x}_{m+1} - \mathbf{x}_m\| < \alpha\mu \|\mathbf{c}(\mathbf{x}_{n-1})\|$$

which demonstrates the third part of the proposition. **End Of Proof**

In the last demonstration we have made use of the following result:

Proposition 9 *For any vector $\mathbf{x}_0$ of $\mathcal{R}^n$ a series of the form $\mathbf{x}_{m+1} = P(\mathbf{x}_m) \cdot \mathbf{x}_m$ where $P(\mathbf{x})$ is a variable projector, but only function of the vector $\mathbf{x}$, is convergent.*

Proof Consider the two exclusive case:

If, on one hand, $\mathbf{x}_m$ belongs to the range of $P(\mathbf{x}_m)$, $\mathbf{x}_m$ is invariant for $P(\mathbf{x}_m)$, then $\mathbf{x}_{m+1} = \mathbf{x}_m$. At the next iteration the situation will be similar since the projector P is only function of $\mathbf{x}_m$. The series is thus stationary and convergent.

If, on another hand, $\mathbf{x}_m$ does not belong to the range of $P(\mathbf{x}_m)$, one can write $\mathbf{x}_m = \mathbf{x}'_m + \mathbf{x}''_m$, where $\mathbf{x}'_m$ is in the range of $P(\mathbf{x}_m)$ and $\mathbf{x}''_m$ in the null-space of $P(\mathbf{x}_m)$. We then have $\mathbf{x}_{m+1} = P(\mathbf{x}_m) \cdot \mathbf{x}_m = \mathbf{x}'_m$, and $\|\mathbf{x}_{m+1}\| < \|\mathbf{x}_m\|$ follows. It means that the series is strictly decreasing in norm.

Combining the two situations we see that the given series is either strictly decreasing in norm, and converging toward 0, or stationary from a given n apart, thus convergent in any case. **End Of Proof**

Then, Proposition (4) provides an iterative algorithm of the form $\mathbf{x}_{m+1} = F(\mathbf{x}_m)$ starting at $\mathbf{x}_0$. During the process, we just have to check that $\|\mathbf{c}(\mathbf{x}_m)\|$ is strictly decreasing. The fact we can have a second order method is not surprising since we are directly cancelling the first order terms (solving the first-order normal equations) as in the Newton-Raphson method. This method, in fact, precisely reduces to this method in the case of a scalar parameter (see Prop14).

Stability of the algorithm Since the p equations $\mathbf{c}(\mathbf{x})$ are not the only one possible set of equations defining the constraints to be taken into account, one can ask whether the choice of such a set of equation or any change of coordinates has an influence on our algorithm. We have the following negative answer:

Proposition 10 *$F(\mathbf{x})$ is invariant upon any linear, one to one, transformation of the set of equations $\mathbf{c}(\mathbf{x})$.*

Proof Consider $\mathbf{d}(\mathbf{x}) = M \cdot \mathbf{c}(\mathbf{x})$ where M is any $p \times p$ invertible matrix. Computing $F(\mathbf{x})$ for $\mathbf{d}()$ yields: $F(\mathbf{x}) = \mathbf{x}_0 + Q^{-1}C^T M^T (MCQ^{-1}C^T M^T)^{-1} M(C(\mathbf{x} - \mathbf{x}_0) - \mathbf{c}(\mathbf{x}))$, since $\frac{\partial \mathbf{d}(\mathbf{x})}{\partial \mathbf{x}} = M \cdot \frac{\partial \mathbf{c}(\mathbf{x})}{\partial \mathbf{x}}$. But a few algebra shows that the matrix M disappears in the previous expression which is now equal to the computation of $F(\mathbf{x})$ for $\mathbf{c}()$ as expected. **End Of Proof**

Let us now consider an affine transformation of the unknows: $\mathbf{x} = A \cdot \mathbf{y} + \mathbf{b}$. The criterion to minimize is in function of $\mathbf{y}$:

$$\mathbf{y}^* = argmin_{\mathbf{y}} max_{\lambda} \left(\frac{1}{2} (\mathbf{y} - A^{-1}(\mathbf{x}_0 - \mathbf{b}))^T \cdot (A^T Q A) \cdot (\mathbf{y} - A^{-1}(\mathbf{x}_0 - \mathbf{b})) + \lambda^T \mathbf{c}(A \cdot \mathbf{y} + \mathbf{b}) \right)$$

and we have:

Proposition 11 *$F(\mathbf{x})$ transforms upon any affine, one to one, transformation $\mathbf{x} = A \cdot \mathbf{y} + \mathbf{b}$ of the unknows as a vector that is $F(\mathbf{x}) = A \cdot F(\mathbf{y}) + \mathbf{b}$. If $F(\mathbf{x})$ converges towards the minimum of $(\mathbf{x} - \mathbf{x}_0)^T \cdot Q \cdot (\mathbf{x} - \mathbf{x}_0)$ under the constraints $\mathbf{c}(\mathbf{x}) = 0$, $F(\mathbf{y})$ converges towards the minimum of $(\mathbf{y} - A^{-1}(\mathbf{x}_0 - \mathbf{b}))^T \cdot (A^T Q A) \cdot (\mathbf{y} - A^{-1}(\mathbf{x}_0 - \mathbf{b}))$ under the constraints $\mathbf{c}(A \cdot \mathbf{y} + \mathbf{b}) = 0$.*

Proof The proof is straightforward, since $C_{\mathbf{x}}(\mathbf{x}) = C_{\mathbf{y}}(\mathbf{y}) \cdot A$. Compute, from $F(\mathbf{x})$, $F(\mathbf{y}) = \mathbf{x}_0 + Q^{-1}A^{-1T}C_{\mathbf{x}}^T(C_{\mathbf{x}}A^{-1}Q^{-1}A^{-1T}C^T)^{-1}(C_{\mathbf{x}}(A^{-1}(A \cdot \mathbf{y} + \mathbf{b}) - \mathbf{x}_0) - A^{-1}\mathbf{c}(A \cdot \mathbf{y} + \mathbf{b}))$ and compare with the hypothesis, using the fact that $F()$ is invariant upon any linear, one to one, transformation of the set of equations. **End Of Proof**

This result is also another justification for taking $\mathbf{x}_0 = 0$ in the demonstrations, since the two problems are equivalent. In addition it shows that the "canonic" problem we are dealing with is simply, in a non-orthogonal frame of reference:

$$\mathbf{x}^* = argmin_{\mathbf{x}} max_{\lambda} J = \left\{ \|\mathbf{x}\|^2 + \lambda^T \mathbf{c}(\mathbf{x}) \right\}$$

since any other problems is equivalent to this one using an affine transformation. However the fact we want to use the previous formulation, is related to the application we are considering, and the fact we prefer to use an orthogonal frame of reference.

B.4 Relation to standard optimization problems

Let us now see what happens in some simple cases. We choose some problems because they also have an analytic solution, thus we can check if our algorithm converges toward the expected solution.

Quadratic minimization with linear constraints In the first case we consider that $\mathbf{c}()$ is linear. Quadratic minimization in the presence of linear constraints is a common problem in geometry (distance from a point to a line or plane, etc...), or optimization (least-square error for a linear subset of variables, etc...). We expect our algorithm to converge in this very simple case, and to be consistent. In fact, it converges in one iteration, and the solution is unique:

Proposition 12 *In the case of a linear constraint $\mathbf{c}(\mathbf{x}) = \mathbf{c}_0 + C \cdot \mathbf{x}$, there is a unique solution to the problem of equation (3) equal to $F(\mathbf{x}_0)$.*

Proof In the case of a linear constraint the normal equations as computed in the proof of Prop.4 are linear, (we assume again $\mathbf{x}_0 = 0$) :

$$\left\{ \begin{array}{rcl} Q \cdot \mathbf{x} + C^T \cdot \lambda & = & 0 \\ C \cdot \mathbf{x} & = & -\mathbf{c}_0 \end{array} \right.$$

with C constant, and using Prop.5 we have directly the solution of this system, using the first line of equation (4): $\mathbf{x} = Q^{-1} \cdot C^T \cdot (C \cdot Q^{-1} \cdot C^T)^{-1}(-\mathbf{c}_0)$. This equation is well defined for $\mathbf{x}$ since the right hand size is a constant. This is just $F(\mathbf{x}_0)$, as expected. **End Of Proof**

Homogeneous coordinates Another simple problem is related to quadratic minimization of homogeneous vectors of parameters. This problem is not well defined since the parameters are known up to a, non-zero, scale factor, but one can consider only unitary vectors, for instance. The relation with our method is the following. The minimization of an homogeneous quadratic form $\mathbf{x}^T \cdot Q \cdot \mathbf{x}$ with $\|\mathbf{x}\| = 1$ as constraint, is known to have the unary eigen-vectors associated with the smallest eigen-value as solution. This is also the case of our algorithm:

Proposition 13 *In the case of the minimization of an homogeneous quadratic form $\mathbf{x}^T \cdot Q \cdot \mathbf{x}$ with $\|\mathbf{x}\| = 1$ the proposed algorithm converges toward the unary eigen-vector of the metric, associated to the smallest eigen-value of Q, and (if the smallest eigen-value of Q is a multiple eigen-value) which distance to the initial value is minimal.*

Contrary to the standard method given in Prop.4, the algorithm cannot be started from $\mathbf{x}_0 = 0$*, but for any generic vector, not orthogonal to the eigen-direction to be found.*

Proof The problem is equivalent to:

$$\mathbf{x}^* = argmin_{\mathbf{X}} max_{\lambda} \frac{1}{2}\mathbf{x}^T \cdot Q \cdot \mathbf{x} + \lambda^T(\mathbf{x}^T \cdot \mathbf{x} - 1)$$

and our algorithm simplifies to:

$$F(\mathbf{x}) = \frac{1}{\mathbf{x}^T \cdot Q^{-1} \cdot \mathbf{x}} Q^{-1} \cdot \mathbf{x}$$

One can also easily see that if $\mathbf{x} = 0$, $F(\mathbf{x})$ is no more well defined since $C_{\mathbf{X}}(\mathbf{x}) = 2\mathbf{x}^T$ is no more of rank 1.

Let us now show that starting from any generic $\mathbf{x}_0 \neq 0$, the series converges, towards the eigen-vector associated to the smallest eigen-value of Q.

First consider the diagonal decomposition of Q:

$$Q = U^T \cdot D \cdot U = U^T \cdot \begin{pmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ & & \dots & \\ 0 & 0 & \dots & \lambda_n \end{pmatrix} \cdot U$$

where U is an orthogonal matrix, well defined since Q is symmetric. Let us consider the problem for $\mathbf{y}_m = U \cdot \mathbf{x}_m$. Please remind that $Q^m = U^T D^m U$, while in this decomposition U is orthogonal ($U^T = U^{-1}$).

By induction, we can establish that:

$$\mathbf{y}_m = U \cdot \mathbf{F}(U^T \cdot \mathbf{y}_{m-1}) = \frac{1}{\alpha_m} D^{-m} \cdot \mathbf{y}_0$$

with $\alpha_m \alpha_{m+1} = \mathbf{y}^T \cdot D^{-(2m+1)} \cdot \mathbf{y}$ and $\alpha_0 = 1$.

Let us first assume the smallest eigen-value is not a multiple eigen-value.

We have $\alpha_m \alpha_{m+1} = \sum_{i=1}^n \lambda_i^{-(2m+1)} (y_0^i)^2 \rightleftharpoons \lambda_{imin}^{-(2m+1)} (y_0^{imin})^2$ when $m \rightarrow \infty$ where λ_{imin} is the smallest eigen-value of Q. And similarly, for a generic vector $\mathbf{v}$ of $\mathcal{R}^n$, $\mathbf{v}^T \mathbf{y}_m = \frac{1}{\alpha_m} \sum_{i=1}^n \lambda_i^{-m} v^i y_0^i \rightleftharpoons \lambda_{min}^{-m} v^{imin} y_0^{imin}$ when $m \rightarrow \infty$. We can assume $v^{imin} \neq 0$ and $y_0^{imin} \neq 0$ since they are components of generic vectors. One can note the importance for $\mathbf{x}_0 = U^T \mathbf{y}_0$ not be orthogonal to the *imin* direction.

In addition, since $\frac{\alpha_m}{\lambda_{imin}^m} \frac{\alpha_{m+1}}{\lambda_{imin}^{m+1}} \rightleftharpoons (y_0^{imin})^2$ we obtain by induction on m, $\alpha_m \rightleftharpoons |y_0^{imin}| \lambda_{imin}^m$. We finally obtain the following equivalent: $\mathbf{v}^T \mathbf{y}_m \rightleftharpoons \pm v^{imin}$.

This means that, for any generic vector $\mathbf{v}$ of $\mathcal{R}^n$, $lim_{n \rightarrow \infty} |\mathbf{v}^T \mathbf{y}_m| = v^{imin}$. This equality is well defined if and only if $\mathbf{y}_m$ converges towards $(0, ...0, 1, 0...)$ where 1 corresponds to the *imin*th component. This is precisely, according to our decomposition, the unary eigen-vector related to the smallest eigen-value.

If the smallest eigen-value is a multiple eigen-value a similar development leads to the following equivalent: $\mathbf{v}^T \mathbf{y}_m \rightleftharpoons \pm \sum_{i \in Imin} v^i y_0^i$, where $Imin$ is the set of indices corresponding to λ_{imin}. This corresponds to the orthogonal projection of $\mathbf{y}_0$ onto the linear sub-space of eigen-vectors corresponding to λ_{imin}, as expected. **End Of Proof**

Let us complete this demonstration by some explanations about the mechanism of convergence.

- One can see that if $\mathbf{X}^*$ is a fixed point for $F(\mathbf{X})$, it is a unitary vector, eigen-vector of Q^{-1} for the eigen-value $l = \mathbf{X}^T \cdot Q^{-1} \cdot \mathbf{X}$, thus eigen-vector of Q or the eigen-value l^{-1}. Then if the series $\mathbf{X}_{m+1} = \mathbf{F}(\mathbf{X}_m)$ converges, it converges toward an eigen-vector of Q,
- The behavior of the mechanism is related to the fact that the constraints $\mathbf{X}^T \cdot F(\mathbf{X}) = 1$ holds. This relation constraints the direction of $\mathbf{X}_m$ to be controlled by its norm. More precisely, since we have:

$$||\mathbf{X}_{m+1} - \mathbf{X}_m||^2 = (\mathbf{F}(\mathbf{X}_m) - \mathbf{X}_m)^T \cdot (\mathbf{F}(\mathbf{X}_m) - \mathbf{X}_m) = ||\mathbf{F}(\mathbf{X}_m)||^2 + ||\mathbf{X}_m||^2 - 2 = ||\mathbf{X}_{m+1}||^2 + ||\mathbf{X}_m||^2 - 2.$$

The convergence is obtained if and only if $||\mathbf{X}_m||$ converges toward 1, since this establish we have a Cauchy series, thus convergent in $\mathcal{R}^n$.

- We have $\mathbf{X}^T \cdot Q \cdot \mathbf{X} \leq ||\mathbf{X}||||Q \cdot \mathbf{X}||$ since it is a scalar product, and $\mathbf{X}^T \cdot Q \cdot \mathbf{X} = ||\mathbf{X}||||Q \cdot \mathbf{X}|| \Leftrightarrow \mathbf{X} \parallel Q \cdot \mathbf{X} \Leftrightarrow \mathbf{X}\ eigen-vector\ of\ Q$, from the definition of eigen-vectors. This mechanism forces, in fact, the ratio $\frac{\mathbf{X}^T \cdot Q \cdot \mathbf{X}}{||\mathbf{X}||||Q \cdot \mathbf{X}||}$ to converges towards 1, since we have:

$$\frac{F(\mathbf{X})^T \cdot Q \cdot F(\mathbf{X})}{||F(\mathbf{X})^T||||Q \cdot F(\mathbf{X})||} = \frac{(\mathbf{X}^T \cdot Q^{-1} \cdot \mathbf{X})}{(\mathbf{X}^T \cdot Q^{-1} \cdot \mathbf{X})^2} \frac{1}{||\frac{Q^{-1} \cdot \mathbf{X}}{\mathbf{X}^T \cdot Q^{-1} \cdot \mathbf{X}}||||\frac{\mathbf{X}}{\mathbf{X}^T \cdot Q^{-1} \cdot \mathbf{X}}||} = \frac{\mathbf{X}^T \cdot Q^{-1} \cdot \mathbf{X}}{||Q^{-1} \cdot \mathbf{X}||||\mathbf{X}||}$$

- The fact the convergence is obtained in the direction of the smallest eigen-value is clear from the demonstration, since the contributions of these eigen-value becomes preponderant in Q^{-m}, while others terms becomes negligible.

Unstability at the beginning of the convergence

Another question[8] is to now how "fast" the convergence could be, especially when the $\mathbf{c}()$ manifold is seen as a concave surface. In order to study this point, let us consider the following problem:

$$min_{\mathbf{X}}||\mathbf{X} - \mathbf{X}_0|| \quad with \quad ||\mathbf{X}|| = 1$$

Our algorithm is starting from $\mathbf{X}_0$ and we have:

$$\mathbf{X}_{n+1} = \mathbf{X}_0 + \frac{\mathbf{X}_n^T \cdot \mathbf{X}_n - 2\mathbf{X}_n^T \cdot \mathbf{X}_0 + 1}{2\mathbf{X}_n^T \cdot \mathbf{X}_n}\mathbf{X}_n$$

and it is easy to see by induction that $\mathbf{X}_n \| \mathbf{X}_0$, in other words the series obtained by our algorithm is a set of vectors aligned with $\mathbf{X}_0$ and which norm is becoming closer to one. Let us note $l_n = ||\mathbf{X}_n||$. One can see that we have the following equation for l_n:

$$l_{n+1} = \frac{1}{2}\left[l_n + \frac{1}{l_n}\right]$$

which converges if and only if $l_0 \neq 0$, as expected from Prop. 13.

However, assume l_0 is very small, then l_1 will be very large and the algorithm will first diverges before being convergent. In other words, in the case of concave manifolds the algorithm might have an erratic behavior, even if convergence is obtained at last.

Monodimensional constraints In a third set of examples we consider the case of a scalar constraints.

One case is related to the use of "qualitative" variable, that is a variable which takes only a finite number of value. Let us consider that $x \in \{x_1, x_2, ...x_r\}$, this is equivalent to use the polynomial constraint: $0 = \prod_{i=1}^{r}(x - x_i)$. The possibility to introduce qualitative variables ia a powerful mean if the proposed method converges in this situation. The tricky point is that the constraints are no longer regular for $x \in \{x_1, x_2, ...x_r\}$, but this is not required by the algorithm which only requires the constraint to be regular during the iterations. Hopefully, with some care, we have a simple positive answer to this problem, which is precisely:

Proposition 14 *In the case of the minimization of a scalar variable our method reduces to the Newton-Raphson iterative scheme.*

If the constraint is algebraic, with all roots positive, starting from $x_0 = 0$, the proposed algorithm converges toward the smallest root of the constraint.

If only two roots, it converges towards the closest one.

Proof The problem is equivalent to:

$$x^* = argmin_x max_\lambda \frac{1}{2}x^2 + \lambda c(x)$$

[8] This example has been proposed by Steve Maybank, and is given here, thanks to his courtesy.

with $c(x) = \prod_{i=1}^{r}(x - x_i)$, and our algorithm simplifies to:

$$F(x) = c'(x)(c'(x)c'(x))^{-1}(c'(x)x - c(x)) = x - c'(x)^{-1}c(x)$$

which is just the Newton-Raphson algorithm, as announced.

Now, using the definition of $c(x) = \prod_{i=1}^{r}(x - x_i)$, we have after a few algebra:

$$F(x) = x + \frac{1}{\sum_{i=1}^{r} \frac{1}{(x_i - x_m)}}$$

Let us first show by induction that $x_{m+1} = F(x_m)$, with $x_0 = 0$, is always smaller that any root of $c(x)$. This is true for x_0 since all roots are positive. But for a given root x_n, we have

$$x_{m+1} = F(x_m) = x_m + \frac{1}{\sum_{i=1}^{r} \frac{1}{(x_i - x_m)}} < x_m + \frac{1}{\frac{1}{(x_{i0} - x_m)}} = x_{i0}$$

which demonstrates the statement. The inequality is true because each term $\frac{1}{(x_i - x_m)}$ was positive since $x_i > x_m$ by induction.

Now the series converges since $F()$ is a contractive mapping because:

$$0 < F'(x) = 1 - \left(\frac{1}{\sum_{i=1}^{r} \frac{1}{(x_i - x)}}\right)^2 \sum_{i=1}^{r} \frac{1}{(x_i - x)^2} < 1 \Leftrightarrow 0 < \left(\frac{1}{\sum_{i=1}^{r} \frac{1}{(x_i - x)}}\right)^2 \sum_{i=1}^{r} \frac{1}{(x_i - x)^2} < 1$$

This last inequality is true because each term $a_i = \frac{1}{(x_i - x)}$ is positive, while its easy to verify that $\sum_{i=1}^{r} a_i^2 < (\sum_{i=1}^{r} a_i)^2$ for a set $\{a_i\}_{i=1..r}$ of positive real numbers.

Finally since the series converges, it must converge toward a fixed point of $F()$, that is -by definition- one root of $c()$. But since the series reminds smaller than any root of $c()$, the only possible fixed point is the smallest root $c()$, as expected.

Consider now two roots, with $x_0 = 0$. If both are positive or both negative, we can use the previous result, but if one is positive and one negative, one can consider without lost of generality a constraint of the form: $c(x) = (x-1)(x+(1+e^2))$, the closest root being 1. Since, we have $F(x) - x = \frac{(1-x)((x+1+e^2)}{2x+e^2}$, starting at $x = 0$, the series $x_{n+1} = F(x_n)$ is strictly increasing, and remains below 1, for $x_n \in [0, 1]$, thus converges to 1 which is a stationary point. **End Of Proof**

This result, although adequate, has to two restrictions: roots should be positive in the general case, and the algorithm is ill-conditioned around these roots. This last point is not a real problem since the algorithm is always stopped before the limit is reached, while a qualitative variable can always be defined though a set $\{x_1, x_2, ...x_r\}$ of positive numbers.

Since the method is identical to the Newton-Raphson iterative scheme, in this case, we also established that the Newton-Raphson method converges as described above. This was not known, I think.

Let us finally discuss the case of general scalar constraints, and review a general set of results related to the convergence of an optimization method:

Proposition 15 *If* $\mathbf{c}(\mathbf{x})$ *is a real function* ($p = 1$) *the algorithm is of the form:*

$$\mathbf{x}_{m+1} = \lambda_m Q^{-1} \cdot \frac{\partial \mathbf{c}(\mathbf{x}_m)}{\partial \mathbf{x}} = G(\lambda_m, \mathbf{x}_m)$$

It is always possible to choose a sequence of λ_m*, such that* $G()$ *is a contractive mapping, thus having convergence, if a minimum exists.*

Let $\mathcal{U}$ *be a convex of* $\mathcal{R}^n$*. If* $\mathbf{c}(\mathbf{x})$ *is a convex real function, over* $\mathcal{U}$*, the algorithm converges.*

Proof We still consider $\mathbf{x}_0$ as the origin.

Rewriting $F(\mathbf{x})$ yields the first part of the proposition with:

$$\lambda_m = \frac{\frac{\partial \mathbf{c}(\mathbf{x}_m)}{\partial \mathbf{x}}^T \cdot \mathbf{x}_m - c(\mathbf{x}_m)}{\frac{\partial \mathbf{c}(\mathbf{x}_m)}{\partial \mathbf{x}}^T \cdot Q^{-1} \cdot \frac{\partial \mathbf{c}(\mathbf{x}_m)}{\partial \mathbf{x}}}$$

Now consider the following modified sequence:

$$\lambda_m^* = min\left(\lambda_m, \frac{||Q||}{||\frac{\partial^2 \mathbf{C}(\mathbf{X}_m)}{\partial \mathbf{X}^2}}\right)$$

it is clear that $||G_{\mathbf{X}}(\lambda_m^*, \mathbf{x})|| < 1$ as required. We then will have convergence. This "trick" is nothing else than use a simple first-order gradient algorithm, until $F()$ is a contractive mapping and then switch to a second-order method as proposed in this paper.

The second part of the proposition is no more than a standard theorem of Optimization since the quadratic positive criterion we minimize is also convex [9]. **End Of Proof**

C Application to Sensory-Motor Tasks : Reflex Behaviors

We now would like to describe a particular application of the previous formalism in *Active Vision*. In this case a generalization of optimal control equations is used to tune reflex behaviors which are lower layers in architecture related to sensory-motor mechanisms.

C.1 Current approaches in designing sensory-motor tasks

Current approaches in designing sensory-motor tasks might be divided into three classes. In the first class authors have tried to formalize sensing behaviors as a very general optimization problem, and defined the objective of the sensing behavior as a criterion to minimize. They implement the behaviors using minimization techniques (for a recent example see [1]), including optimal control techniques [9]. In a second class people use the concept of "task-function" as defined by [34]. This approach has already yielded promising results [14, 15] in vision, and several slow real-time experiments have been realized. It is however much more suitable to low-level control loops using the visual sensor as an input, than to the design of complex sensing behaviors [31]. The last class of studies, including neuro-physiological modeling of the oculomotor system, defines sensing behaviors as a combination of "black-boxes", building a functional model of the system [8]. There is however a lack of studies on the way to combine different techniques at different level of speed or complexity. Despite the existence of relevant studies on integration of sensing behaviors, and perception system architecture (see [20, 26]), there is no result about the stability of complex behaviors (expect for [34]).

Let us now try to formalize specific behaviors which have to be used in cooperation with vision, when intending to realize active perception of the visual surroundings, as defined previously.

The specificity of the present approach is first to use of a powerful algorithm given by control theory. We thus define an optimal estimator and controller from a sensory-motor linear system. We then design a specific architecture to deal with real time implementations but without giving up the use of non-linear constraints, since we make use of extension of the original Kalman Filter.

C.2 Theoretical framework

Let us consider a system $\mathcal{S}$ with an input $\mathbf{u}(t)$ and an internal state $\mathbf{x}(t)$. We assume it has the following linear evolution equation:

$$\dot{\mathbf{x}}(t) = A(t) \cdot \mathbf{x}(t) + B(t) \cdot \mathbf{u}(t) + \xi(t)$$

($\xi(t)$ is a zero-mean white Gaussian noise of covariance $V(t)$),

Let us assume we obtain a partial approximate linear measure $\mathbf{y}$ of its state $\mathbf{x}$:

$$\mathbf{y}(t) = M(t) \cdot \mathbf{x}(t) + \nu(t)$$

($\nu(t)$ is a zero-mean white Gaussian noise of covariance $\Lambda(t)$).

We want, on one hand, to observe the internal state of this system (sensory task or "observer") and, on another hand, to control the input and the internal state of the system (motor task or "controller"). Such a mechanism is shown on Fig.3.

In this formalism, a *reflex task* is related to the minimization of a given quadratic statistical criterion:

$$J = E[\int_{t_0}^{\infty} \underbrace{(\mathbf{x}(t) - \mathbf{x}_0(t))^T \cdot Q(t) \cdot (\mathbf{x}(t) - \mathbf{x}_0(t))}_{observer} + \underbrace{(\mathbf{u}(t) - \mathbf{u}_0(t))^T \cdot R(t) \cdot (\mathbf{u}(t) - \mathbf{u}_0(t))}_{controller} \; dt]$$

The first term of the task is related to the estimate of the internal state of the system $\mathcal{S}$ under observation, while the second term is related to control the input of the system $\mathcal{S}$. In

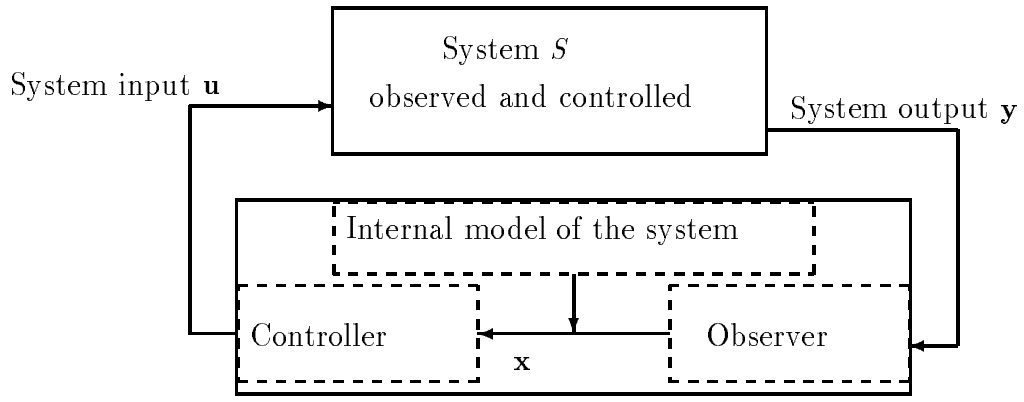

Figure 3: Controlling an input/output system

other words, this quadratic form defines a *sensory-motor* task, this is an elementary perceptual task, called a *reflex*.

In other words, the task is described by a trajectory and a quadratic form which defines how accurately this trajectory has to be followed.

Proposition 16 *The separation principle [24, 2] allows us to compute the optimal command as:*

$$\begin{array}{rcl} \mathbf{u}^*(t) & = & \mathbf{u}_0(t) - L(t) \cdot \mathbf{x}^*(t) \\ & & L(t) = \left[R(t)^{-1} \cdot B(t)^T \cdot P(t)\right] \\ & with & \\ \dot{\mathbf{x}}^*(t) & = & \mathbf{x}_0(t) + A(t) \cdot \mathbf{x}^*(t) + B(t) \cdot \mathbf{u}^*(t) + K(t) \cdot (\mathbf{y}(t) - M(t) \cdot \mathbf{x}^*(t)) \\ & & K(t) = \left[S(t) \cdot M(t)^T \cdot \Lambda(t)^{-1}\right] \end{array} \tag{7}$$

In other words, the system can be decomposed into two parts: an optimal controller (given by the first equation) and an optimal observer (the second equation), which is just a linear Kalman filter.

Let us give the demonstration of this proposition since we use the original result in a different way as [2].

Proof Let us take $\mathbf{x}_0(t) = \mathbf{u}_0(t) = 0$ without loss of generality, as done in the appendix B.

We will consider the residual quantities $\tilde{\mathbf{x}} = \mathbf{x} - \mathbf{x}^*$ and $\tilde{\mathbf{u}} = \mathbf{u} - \mathbf{u}^*$.

Since the evolution of $\mathbf{x}$ is known from the model we have:

$$\begin{array}{rcl} \dot{\mathbf{x}} & = & A \cdot \mathbf{x} + B \cdot \mathbf{u} + \xi \\ & = & A \cdot \mathbf{x} + B \cdot \left[-R^{-1} \cdot B^T \cdot P \cdot (\mathbf{x} - \tilde{\mathbf{x}}) + \tilde{\mathbf{u}}\right] + \xi \\ & = & \left[A - B \cdot R^{-1} \cdot B^T \cdot P\right] \cdot \mathbf{x} + B \cdot R^{-1} \cdot B^T \cdot P \cdot \tilde{\mathbf{x}} + B \cdot \tilde{\mathbf{u}} + \xi \end{array} \tag{8}$$

Let us write $J = E\left[\int_{t_0}^{\infty} \mathcal{J}(t)dt\right]$. We can also expand $\mathcal{J}$:

$$\begin{array}{rcll} \mathcal{J} & = & \mathbf{x}^T \cdot Q \cdot \mathbf{x} + \mathbf{u}^T \cdot R \cdot \mathbf{u} & \\ & = & \mathbf{x}^T \cdot \left[Q + P \cdot B \cdot R^{-1} \cdot B^T \cdot P\right] \cdot \mathbf{x} - 2\mathbf{x}^T \cdot P \cdot B \cdot \tilde{\mathbf{u}} & \\ & & -2\mathbf{x}^T \cdot P \cdot B \cdot R^{-1} \cdot B^T \cdot P \cdot \tilde{\mathbf{x}} & \\ & & + \left[\tilde{\mathbf{u}} + R^{-1} \cdot B^T \cdot P \cdot \tilde{\mathbf{x}}\right]^T \cdot R \cdot \left[\tilde{\mathbf{u}} + R^{-1} \cdot B^T \cdot P \cdot \tilde{\mathbf{x}}\right] & using \quad (7) \\ & = & \mathbf{x}^T \cdot \left[Q - P \cdot B \cdot R^{-1} \cdot B^T \cdot P\right] \cdot \mathbf{x} + 2\mathbf{x}^T \cdot P \cdot A \cdot \mathbf{x} - 2\mathbf{x}^T \cdot P\dot{\mathbf{x}} & \\ & & + \left[\tilde{\mathbf{u}} + R^{-1} \cdot B^T \cdot P \cdot \tilde{\mathbf{x}}\right]^T \cdot R \cdot \left[\tilde{\mathbf{u}} + R^{-1} \cdot B^T \cdot P \cdot \tilde{\mathbf{x}}\right] & using \quad (8) \\ & = & \mathbf{x}^T \cdot \mathcal{M} \cdot \mathbf{x} - 2\mathbf{x}^T \cdot P\dot{\mathbf{x}} & \\ & & + \left[\tilde{\mathbf{u}} + R^{-1} \cdot B^T \cdot P \cdot \tilde{\mathbf{x}}\right]^T \cdot R \cdot \left[\tilde{\mathbf{u}} + R^{-1} \cdot B^T \cdot P \cdot \tilde{\mathbf{x}}\right] & \\ with & & \mathcal{M} = Q + 2P \cdot A - P \cdot B \cdot R^{-1} \cdot B^T \cdot P & \end{array}$$

But $\tilde{\mathbf{u}}$ should only depend on the measures $\mathbf{y}$, thus $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{u}}$ are independent. Then $E\left[\tilde{\mathbf{u}}^T \cdot R^{-1} \cdot B^T \cdot P \cdot \tilde{\mathbf{x}}\right] = 0$ and we can further simplify J [9]:

$$\begin{aligned} J &= \mathcal{I}(\mathbf{x}, \dot{\mathbf{x}}) + E\left[\int_{t_0}^{\infty} \left[\tilde{\mathbf{u}} + R^{-1} \cdot B^T \cdot P \cdot \tilde{\mathbf{x}}\right]^T \cdot R \cdot \left[\tilde{\mathbf{u}} + R^{-1} \cdot B^T \cdot P \cdot \tilde{\mathbf{x}}\right] dt\right] \\ &= \mathcal{I}(\mathbf{x}, \dot{\mathbf{x}}) + E\left[\int_{t_0}^{\infty} \tilde{\mathbf{u}}^T \cdot \tilde{\mathbf{u}} dt\right] + E\left[\int_{t_0}^{\infty} \tilde{\mathbf{x}}^T \cdot P \cdot B \cdot R^{-1} \cdot B^T \cdot P \cdot \tilde{\mathbf{x}} dt\right] \\ &= \mathcal{I}(\mathbf{x}, \dot{\mathbf{x}}) + \int_{t_0}^{\infty} E\left[\tilde{\mathbf{u}}^T \cdot \tilde{\mathbf{u}}\right] dt + \int_{t_0}^{\infty} trace\left\{P \cdot B \cdot R^{-1} \cdot B^T \cdot P \cdot S\right\} dt \end{aligned}$$

It is now obvious that the criterion J is minimum if on one hand $\tilde{\mathbf{u}} = 0$ which corresponds to using $\mathbf{u}^*$ as output command, and on the other hand if the covariance S of $\tilde{\mathbf{x}}$ is minimal which corresponds to use the Kalman filter optimal estimate for $\mathbf{x}$. **End Of Proof**

These equations depend upon two matrices $P(t)$ and $S(t)$ and are related to the internal model of the system by a set of quadratic differential equations (Riccati equations) as derived usually:

$$\begin{aligned} Q(t) &= -\dot{P}(t) - \left(P(t) \cdot A(t) + A(t)^T \cdot P(t) - P(t) \cdot B(t) \cdot R(t)^{-1} \cdot B(t)^T \cdot P(t)\right) \\ V(t) &= +\dot{S}(t) - \left(S(t) \cdot A(t)^T + A(t) \cdot S(t) - S(t) \cdot M(t)^T \cdot \Lambda(t)^{-1} \cdot M(t) \cdot S(t)\right) \end{aligned} \tag{9}$$

Obviously, $S(t)$ is the covariance of the $\mathbf{x}(t)$ estimate. We wrote these equations in a form in which $P(t)$ and $S(t)$ are input variables which tune the metric $Q(t)$ defining the minimization criterion, and the covariance $V(t)$ on the model. Thus, instead of having to solve these two differential equations, we might just attempt to adjust indirectly $P(t)$ and $S(t)$, in order to obtain the required $Q(t)$ and $V(t)$.

C.3 Architecture and application of reflex behaviors

As a computer module, the system is defined by the following data structure:

- Input : set of scalar real parameters $\mathbf{y}$
- Output : set of scalar real parameters $\mathbf{u}$
- Internal state : set of scalar vectorial parameters $\mathbf{x}$
- Internal model :
  - Evolution equation (A, B, V)
  - Measurement equation (C, Λ^{-1})
- Local task :
  - Output control $(R^{-1}, \mathbf{u}_0(t))$
  - State control $(Q, \mathbf{x}_0(t))$

its functional architecture is shown on Fig.4.
The algorithm is thus implemented using three layers:

1. The lower layer is in fact a simple two stage linear feedback, computing $\mathbf{x}$ and $\mathbf{u}$ form equation (7). The implementation is very fast, one iteration is of the form $\mathbf{u}_n = L\mathbf{x}_n$ and $\mathbf{x}_{n+1} = \mathbf{x}_n + B \cdot \mathbf{u} + K \cdot (\mathbf{y}_n - C \cdot \mathbf{x}_n)$ and requires a fixed amount of operations, precisely $o(dim_x + 2dim_x dim_u + 2dim_x dim_y)$ operations.

[9] We write $\mathcal{I}(\mathbf{x}, \dot{\mathbf{x}}) = E\left[\int_{t_0}^{\infty} \mathbf{x}^T \cdot \mathcal{M} \cdot \mathbf{x} - 2\mathbf{x}^T \cdot P\dot{\mathbf{x}} dt\right]$ which is not dependent upon $\tilde{\mathbf{x}}$ or $\tilde{\mathbf{u}}$ thus not modified by the choice of these parameters. Moreover, it can be showned that we have:

$$\mathcal{I}(\mathbf{x}, \dot{\mathbf{x}}) = \mathbf{x}(0)^T \cdot P(0) \cdot \mathbf{x}(0) + \int_{t_0}^{\infty} trace\left\{P \cdot V\right\} dt + P(0) \cdot \Lambda(0)$$

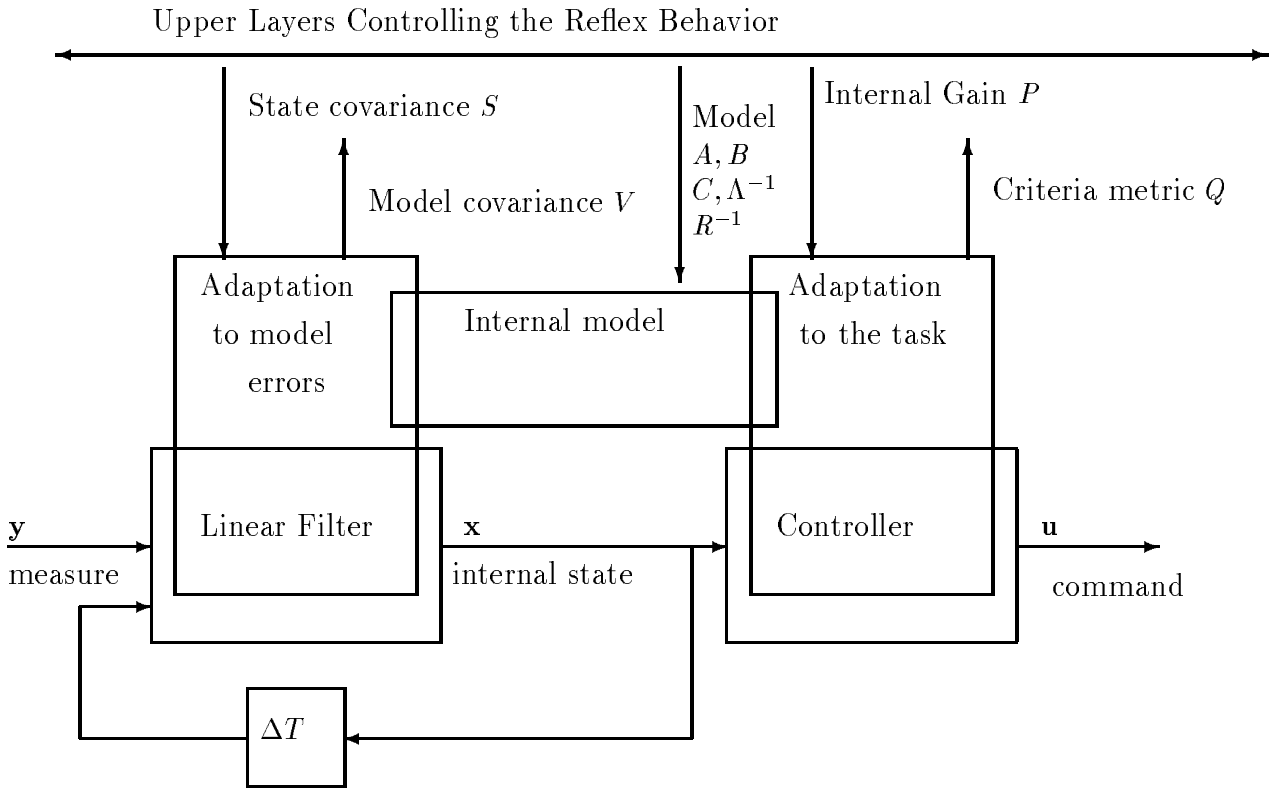

Figure 4: Functional architecture of a sensing reflex

2. The intermediate layer computes the gains $L(t)$ and $K(t)$ given in equation (7) which are related to $P(t)$ and $S(t)$. It also computes the matrix $Q(t)$ and $V(t)$ from equation (9) depending on $P(t)$, $S(t)$ and their variations. It again requires a fixed amount of operations, but it is not necessarily computed at each iteration, since the gains can remains fixed during several operation as in a steady-state mode. The amount of operations varies depending of the format of the variables, but it is of order $o(dim_x^3 + dim_u^3)$ at most. Since $P(t)$ and $S(t)$ are locally constant $\dot{P(t)}$ and $\dot{S(t)}$ can be negligibled.
3. The upper layer attempts to compute the matrix $P(t)$ and $S(t)$ in order to fit with the desired matrix $Q(t)$ and $V(t)$. This adaptive layers use equation (9) as error equations.

In fact, the implementation of gain adaptation, if the matrix P and S are taken as state variables and the matrix Q and V as control variables, does not required explicit resolution of the Riccati equations, but a simple matrix computation, and a recursive minimization.

Discussion This combination of a linear stochastic filter and a optimal linear command, which are known to be optimal as a sequential system, might be considered as the upper bound - in term of complexity - for reflex behaviors. Such a mechanism is a linear feedback, its closed-loop gain being adaptive as biological reflexes are [8].

In addition, since the algorithm is related to a minimization criterion, it can be understood by upper layers as a low level actor with some precise objective (minimization of the criterion). Moreover, since the composite gain of the reflex is related to statistical parameters of the system (input errors, criterion to minimize, etc.) the modification of these gains is governed by interpretable criteria.

If the internal model is close to the true system this mechanism is stable, thus the stability of reflex embedded in a complex functional architecture will be easy to maintain, if the parameters

of the model and of the task are smoothly tuned.

We can consider as reflex behaviors, determinist synchronous, input-output linear systems, working at a fixed sampling-rate, which can be modelized in term of recursive or differential equations. As a computer program, this layer is build up of polynomial algorithms, with neither iterative nor recursive pieces of code, except for the adaptation of the gains. The amount of operations is fixed and known in advance.

Usual hardware, including internal analog loops, might be considered as reflex behaviors, since their related models have this general form. An internal representation of these sub-systems being provided, it is thus possible to identify there behaviors with software reflexes.

This formalism make use of a differential equation for the model, not a discrete time equation, since the system to control is in fact an "analog" physical system.

Common non-adaptive feedback such as: target tracking by cancelling error in velocity, picture stabilization, or system saccade, are tasks which can de defined in the previous formalism, their implementation corresponding to the reflex behavior implementation.

If we assume the system is in a steady state mode, each matrix is constant. This fundamental restriction gives the nice properties of constant gain, fast-computation, and the simple direct relation between gain P, Λ and Q, V.

Parameter adaptation can be done in several ways:

- Tuning Q means system adaptation to a modified task,
- Tuning V means system adaptation to a variation of the model validity,
- Modifying the system model given by A, B, C or Λ^{-1} means system adaptation to a new internal ego-representation of the system,
- The output metric R^{-1} is often related to energy, or mechanical limits of the system and can also be adapted

The trajectories $\mathbf{x}_0(t)$ and $\mathbf{u}_0(t)$ can also be used to control those reflex behaviors along time, and to track a a general trajectory in the state space, as in the task-function approach.

Tuning reflex behaviors is no more connected with real-time, whereas the variation of their parameters is assumed to be slow with respect to the control-loops.

C.4 Introducing constraints in a reflex behavior

Let us consider the augmented state vector and matrix :

$$\mathbf{X} = \left| \begin{array}{c} \mathbf{x} \\ \mathbf{u} \end{array} \right. \quad O = \begin{pmatrix} Q & 0 \\ 0 & R \end{pmatrix}$$

The previous mechanism is based on the minimization of a quadratic definite positive criterion and the related algorithm is a recursive algorithm which minimizes this quadratic form, say:

$$(\mathbf{X} - \mathbf{X}_0)^T \cdot O \cdot (\mathbf{X} - \mathbf{X}_0)$$

Let us now no longer allow the vector $\mathbf{X}$ to be everywhere in $\mathcal{R}^n$, but we consider state vectors belonging to a manifold, defined by p implicit equations $\mathbf{c}(\mathbf{X}) = 0_{\mathcal{R}^p}$, as developed in this paper. Now the criterion to minimize is:

$$\left\{ \begin{array}{c} min_{\mathbf{X}} \frac{1}{2}(\mathbf{X} - \mathbf{X}_0)^T \cdot O \cdot (\mathbf{X} - \mathbf{X}_0) \\ \mathbf{c}(\mathbf{X}) = 0 \end{array} \right.$$

But this is exactly what is done in a Pseudo Kalman Filter and the related algorithm can be implemented again here : A linear unconstrained estimate (implemented using three layers as discussed here) and a reprojection mechanism dealing with constraints.

In order to obtain the minimization of the previous criterion, our theory states that one can first obtain the minimum without constraint, as done in the previous section, then reproject

on the manifold using the covariances as a linear-space metric. It is thus possible to introduce non-linear constraints in the estimate, not in real-time but at upper layers tuning the previous reflex behavior.

With the introduction of non-linear constraints, the class of models to be used is no longer limited. Moreover, the quadratic criterion is not a restriction, since any regular convex $\mathbf{C}^2$ criterion can be related to a quadratic criterion by a one-to-one mapping. In addition, qualitative variables, unitary vectors, homogeneous coordinates etc. can be introduced, as already pointed out.

This *iterative behavior* drives at higher level, the reflex behavior as a "controller". Roughly speaking, implementation of iterative behaviors are simply made of a one-level internal loop, called "as often as possible", and provides outputs on request, while the precision depends upon the number of iterations realized.

The way we designed the reprojection in a Pseudo Kaman Filter has two consequences. On one hand, they can react very quickly, after one loop, iterative behaviors already provide an estimation of the final state (the convergence is in fact quadratic under reasonable assumptions). Moreover, they can be controlled, using a synchronous programming language, since the amount of calculation for one iteration, is polynomial, and known in advance by the system, as for reflex behaviors. They, thus, can be implemented in the same architecture, as reflex behaviors.

C.5 Association of reflex behaviors

Given two or more reflex behaviors the problem is also to associate them and form composite behaviors. In fact two situations can occurred. These behaviors are compatible (cooperation) if their related constraints can be satisfied together, otherwise they are concurrent (competition).

- **Cooperative behaviors** (decoupling in input or output) In this first class, we consider that two or more behaviors can be decoupled either because they work on different inputs, or because they can be applied on independent degrees of freedom. This formalism corresponds to the notion of primary and secondary task in the *task-function approach* [34]. Such decoupling is not limited simply to a choice between the available degrees of freedom, but is extended as in the task-function approach to a use of different degrees of freedom in complete synergy.
  The notion of cooperation between behaviors is very simple: Two (or more) generic iterative behaviors can cooperate if and only if their normal equations have solutions in common. The result is also an iterative behavior, which minimizes conjointly the given criteria. It corresponds to the composite criterion:

$$\begin{aligned}\mathbf{X} = argmin_{\mathbf{X}} max_{\lambda,\mu} (\mathbf{X}-\mathbf{X}_0)^T \cdot O \cdot (\mathbf{X}-\mathbf{X}_0) \\ +\lambda_a^T \cdot \mathbf{c}_a(\mathbf{X}) + \lambda_b^T \cdot \mathbf{c}_b(\mathbf{X})\end{aligned}$$

  with normal equations:

$$\left\{\begin{array}{rcl} O(\mathbf{X}-\mathbf{X}_0)+ & & \\ \frac{\partial \mathbf{c}_a}{\partial \mathbf{X}}(\mathbf{X})^T \cdot \lambda_a + \frac{\partial \mathbf{c}_b}{\partial \mathbf{X}}(\mathbf{X})^T \cdot \lambda_b & = & 0 \\ \mathbf{c}_a(\mathbf{X}) & = & 0 \\ \mathbf{c}_b(\mathbf{X}) & = & 0 \end{array}\right.$$

- **Competitive behaviors** (linear combination of behaviors, priority and threshold criteria). If the previous property is not valid, minimizing each criterion related to two or more behaviors is no more possible, while one can try to minimize a linear combination of these criteria. In this case there is a "competition" between the different behaviors. Nevertheless, the result is also an iterative behavior. Performing commutations between behaviors, thresholds or others heuristics can be implemented using adequate weights $\mu_{a/b}(\mathbf{X})$:

$$\begin{aligned}\mathbf{X} = argmin_{\mathbf{X}} max_{\lambda,\mu} (\mathbf{X}-\mathbf{X}_0)^T \cdot O \cdot (\mathbf{X}-\mathbf{X}_0) \\ +\lambda_a^T \cdot \mu_{a/b}(\mathbf{X}) \cdot \mathbf{c}_a(\mathbf{X}) + \lambda_b^T \cdot (Id - \mu_{a/b}(\mathbf{X})) \cdot \mathbf{c}_b(\mathbf{X})\end{aligned}$$

with normal equations:

$$\left\{ \begin{array}{rcl} O(\mathbf{X}-\mathbf{X}_0)+ & & \\ \mu_{a/b}(\mathbf{X}) \cdot \frac{\partial \mathbf{c}_a}{\partial \mathbf{X}}(\mathbf{X})^T \cdot \lambda_a + (Id - \mu_{a/b}(\mathbf{X})) \cdot \frac{\partial \mathbf{c}_b}{\partial \mathbf{X}}(\mathbf{X})^T \cdot \lambda_b & = & 0 \\ \mu_{a/b}(\mathbf{X}) \cdot \mathbf{c}_a(\mathbf{X}) & = & 0 \\ (Id - \mu_{a/b}(\mathbf{X})) \cdot \mathbf{c}_b(\mathbf{X}) & = & 0 \end{array} \right.$$

- **Sequences of behaviors : scripts** In addition to those two parallel combination of reflex behaviors, one might think about combining behaviors sequentially. This is done in several reactive systems programming languages such as Esterel [6]. Just remind that such a programming language translates the code (including sequential or parallel primitives) into a deterministic automata [7] thus yielding algorithmic complexity of $o(1)$.
  Such a mechanism can be formally defined as follows. Let us note:
  $\mathcal{B}[\mathbf{X}, \mathbf{c}(), \{M_i, W_i\}]$ an iterative behavior (state, constraint, relation to measurements).
  $\mathcal{B}_a[\ldots] \bigoplus \mathcal{B}_b[\ldots]$ the cooperation between two behaviors, as defined previously.
  $\mathcal{B}_a[\ldots] \bigsqcup \mathcal{B}_b[\ldots]$ the concurrency between two behaviors, as defined previously.
  $\mathcal{B}_a[\ldots] \bigwedge \mathcal{B}_b[\ldots]$ the parallel execution between two behaviors, having two different states ($\mathbf{X}_a$ and $\mathbf{X}_b$) and no interaction.
  $\div[\mathbf{X}]$ a test on one vectorial state.
  A sequential combination of behaviors or **script** is a sequence of $If\ \ Then$ constructs of the form:
  $If\ \ \mathcal{E}\ \ Then\ \ \mathcal{B}$
  where $\mathcal{E}$ is a boolean expression of $\div[]$ tests and $\mathcal{B}$ a behavior expression of elementary behaviors using $\bigoplus$, $\bigsqcup$ or $\bigwedge$ operators.

The overall system architecture has then a form shown in Fig.5. The proposed formalism has three advantages:

- Any combination of behaviors is still a behavior, thus stability, convergence, implementation cost and other properties can be studied as for standard iterative behaviors.
- There is a criterion to decide whether two behaviors can be cooperative or if they are in competition.
- There is a direct link between those algorithmic schemas and reactive system synchronous programming. Then one can use these behaviors as primitives to be combined to form scripts of visual perception.

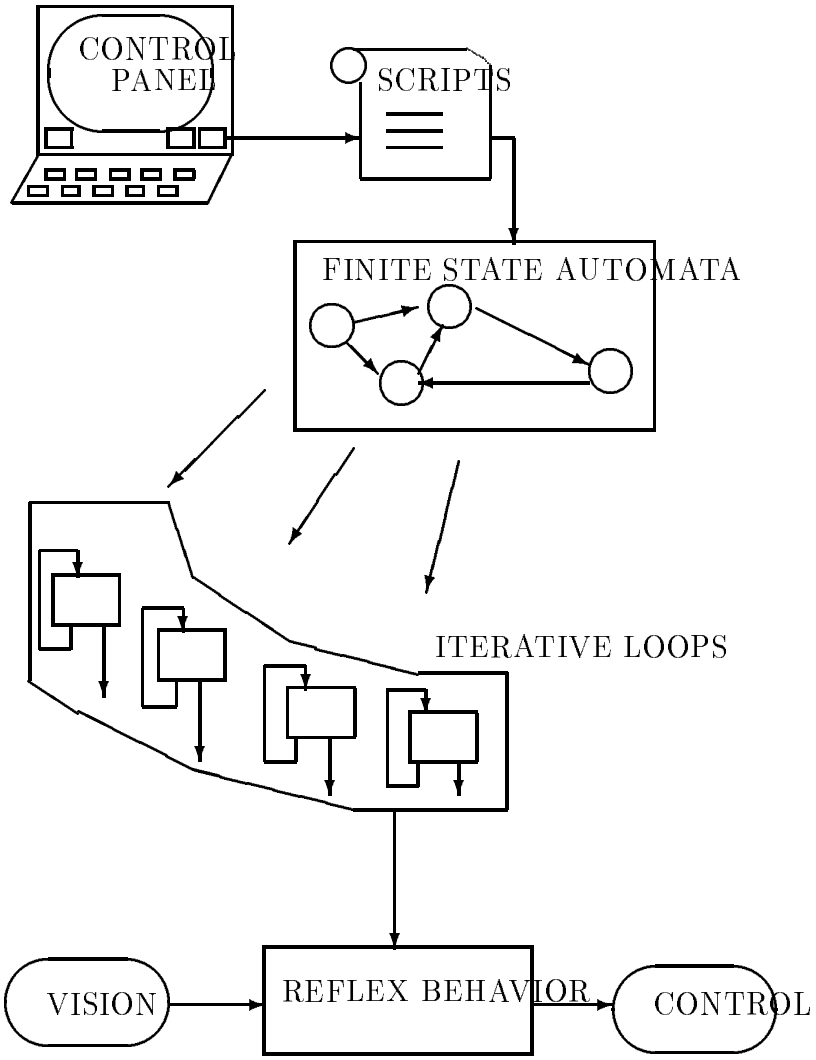

Figure 5: Overall architecture of a system involving Constrained Reflex-Behaviors

D A maple routine to generate C-code of Pseudo Kalman Filters

```
#############################################################################
# FUNCTION: pkalman - generates a C-code for a static pseudo Kalman Filter #
#                                                                           #
# CALLING SEQUENCE: pkalman(`proc_name`,x,xmin,xmax,c,M,C,i)                #
#                                                                           #
# PARAMETERS:                                                               #
#  `proc_name`      is the filename for the C-code to be put in             #
#  x                is the state-vector to be estimated                     #
#  xmin,xmax        are min and max values of the state vector              #
#  xstep            is the sampling of x (acceptable error)                 #
#  c                is the set,vector,list or boolean expression of         #
#                   constraints                                             #
#  M                is the matrix or vector describing linear relations     #
#                   between state and measures                              #
#  i                is the index used in M to label the measures            #
#                                                                           #
# SYNOPSIS:                                                                 #
#                                                                           #
# - The pseudo kalman filter is defined as the minimization of a set of     #
# linear measurements equations in the presence of constraints.             #
#                                                                           #
# - The output is a block of C-code with no argument, to be included        #
# in a C-procedure.                                                         #
#                                                                           #
# SEE ALSO :  linalg, linalg2, kalman                                       #
#############################################################################
```

References

[1] A. Abbott and N. Ahuja. Active surface reconstruction by integrating focus, vergence stereo and camera calibration. In *Proceedings of the 3th ICCV, Osaka*, 1990.

[2] K. Astrom. *Introduction to Stochastic Control Theory.* Academic Press, 1970.

[3] N. Ayache. *Artificial Vision for Mobile Robots.* MIT Press, Cambridge, Massachusetts, 1989.

[4] N. Ayache and O. Faugeras. Maintaining representations of the environment of a mobile robot. *IEEE Transactions on Robotics and Automation*, 1989.

[5] Y. Bar Shalom and T. E. Fortmann. *Tracking and Data Association.* Academic-Press, Boston, 1988.

[6] G. Berry and L. Cosserat. The synchronous programming langage esterel and its mathematical semantics. In *Seminar on Concurrency, LNCS.* Springer Verlag, 1985.

[7] G. Berry and R. Sethi. From regular expressions to determinist automata. *Transactions on Computer Science*, 25:117–126, 1987.

[8] A. Berthoz and G.Melvill Jones. *Adpative Mechanism in Gaze Control.* Elsevier, Amsterdam, 1985.

[9] R. Boudarel, J. Delmas, and P. Guichet. *Commande Optimale des Processus.* Dunod, Paris, 1968. Vol : 2, 3, 4.

[10] A. E. Bryson and Y. C. Ho. *Applied Optimal Control.* Blaisdell Publishing Company, Waltham, Massachusetts, 1977.

[11] P. R. C. Garnier and Y. Papegay. Modelisation dynamique litterale. *Computer Methods in Applied Mechanics and Engineering*, 75:215–225, 1989.

[12] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* Wiley Interscience, New-York, 1973.

[13] M. Emery. *Stochastic Calculus in Manifolds.* Springer Verlag, New-York, 1989.

[14] B. Espiau, F. Chaumette, and P. Rives. Une nouvelle approche de la relation vision-commande en robotique. Technical Report RR-1172, INRIA, Sophia, France, 1988.

[15] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Transaction on Robotics and Automation*, 1991. in press.

[16] O. D. Faugeras, R. Deriche, N. Ayache, F. Lustman, and E. Giuliano. Depth and motion analysis: the machine being developed within esprit project 940. In *Proceedings of the IAPR Workshop on Computer Vision (Special Hardware and Industrial Applications), Tokyo, Japan*, pages 35–44, October 1988.

[17] H. F.Durrant-Whyte. *Integration Coordination and Control of Multi-Sensor Robot System.* Kluwer Academic Publishers, 1988.

[18] E. Francois and P. Bouthemy. Multiframe-based identification of mobile components of a scene with a moving camera. Technical Report 564, IRISA, Rennes, France, 1990.

[19] P. E. Gill and W. Murray. Algorithms for the solution of non-linear least squares problem. *SIAM Journal on Numerical Analysis*, 15:977–992, 1978.

[20] R. Grupen and T. Henderson. Autochtonous behaviors - mapping perception to action. In H. T., editor, *Traditional and Non-Traditional Robotic Sensors.* Springer-Verlag, Berlin, Sept. 1989.

[21] A. M. Jazwinsky. *Stochastic Processes and Filtering Theory.* Academic Press, 1970.

[22] L. Kanal and J. Lemmer. *Uncertainty in Artificial Intelligence.* North-Holland Press, Amsterdam, 1988.

[23] D. Kapur and J. Mundy. *Geometric Reasoning.* MIT Press, Cambridge, 1989.

[24] R. Lee. *Optimal Estimation, identification and control.* MIT Press, Cambridge, 1964.

[25] F. LeGland and A. Gondel. Systematic numerical experiments in non-linear filtering with automatic fortran code generation. In *Proceedings of the 25th IEEE Conference on Decision and Control*, Athenes, 1986.

[26] R. Lumia. Sensor-based robot control requirement for space. In H. T., editor, *Traditional and Non-Traditional Robotic Sensors.* Springer-Verlag, Berlin, Sept. 1989.

[27] S. Maybank. Filter based estimates of depth. In *Proceedings of the British Machine Vision Association.* Univeristy of Oxford, Sept. 1990.

[28] P. Maybeck. *Stochastic models, estimation and control.* Academic-Press, New-York, 1979.

[29] P.E.Jupp and K.V.Martin. A unified view of the theory of directional statistics. *Inernational Statistical Review*, 57, 1989.

[30] M. J. D. Powell. The convergence of variable metric methods for nonlinearly constrained optimization calculations. In O. Mangasarian, R. Meyer, and S. Robinson, editors, *Nonlienar Programming*. Academic press, New York, 1978.

[31] P. Rives. Dynamic vision : Theoretical capabilities and practical problems. In *NATO Workshop on "Kinematic and Dynamic Issues in Sensor Based Control", Italy*, Oct. 1990.

[32] R.T.Collins and R.S.Weiss. Vanishing point calculation as a statistical inference on the unit sphere. In *Proceedings of the 3th ICCV, Osaka*, 1990.

[33] P. A. Ruymgaart and T. T. Soong. *Mathematics of Kalman-Bucy filtering*. Springer Verlag, Berlin, 1985.

[34] S. Samson, M. le Borgne, and B. Espiau. *Robot Control: The Task-function Approach*. Oxford University Press, 1990.

[35] M. Spivak. *A Comprehensive Introduction to Differential Geometry*. Berkeley, 1971. Vol. 1 to 5.

[36] P. Tournassoud. *Géométrie et intelligence artificielle pour les robots*. Hermes, Paris, 1988.

[37] T. Viéville. Construction d'un modèle robotique du contrôle neurophysiologique des mouvements oculaires en vue de l'élaboration de robots de 3ème génération. Technical Report Rapport de Recherche No 86J0326, Ministère de la Recherche et de la Technologie, 1987.

[38] T. Viéville. Estimation of 3D-motion and structure from tracking 2D-lines in a séquence of images. In *Proceedings of the 1th ECCV, Antibes*, 1990.

[39] T. Viéville. Real time gaze control : Architecture for sensing behaviours. In *Fifth Scandinavian Workshop on Computational Vision*, 1991.

[40] T. Viéville. Using a symbolic calculator as a program generator : Application to algorithms of visual perception. In *Greco de Calcul Formel*, Luminy, Marseille, 1991.

[41] T. Viéville. Computation of ego-motion and structure from visual and inertial sensors using the vertical cue. Technical Report RR, INRIA, Sophia, France, 1992. in preparation.

[42] T. Viéville and O. Faugeras. Feed forward recovery of motion and structure from a sequence of 2D-lines matches. In *Proceedings of the 3th ICCV, Osaka*, 1990.

[43] Z. Zhang and O. Faugeras. Building a 3D world model with a mobile robot: 3D line segment representation and integration. In *Proc. 10th International Conference on Pattern Recognition*, pages 38–42, Atlantic City, New Jersey, USA, June 1990. IEEE.

Acknowledgments **Olivier Faugeras** and **Steve Maybank** are gratefully acknowledge for the fruitful discussions we had during the progress of this work.

We are specially thankful to **Fabien Campillo** and **François Legland** for their precious advices, and to **C. LeMaréchal** for some fundamental discussions.

Thanks to **Jacques Droulez** and **Valérie Cornilleau-Perez** for their powerful ideas about formal models of sensory-motor tasks.

Formal computations have been derived using the **Maple** software and the INRIA *mpls* package. This work was partially completed under **Esprit Project P5390/RTGC**.