

# Efficient Validation of Matching Hypotheses using Mahalanobis Distance

J.M.M. Montiel and L. Montano

*{josemari,montano}@posta.unizar.es*

*Dpto. Informática e Ingeniería de Sistemas Centro Politécnico Superior.*

*University of Zaragoza, María de Luna 3, E-50015 Zaragoza Spain,*

*Tel. 34 (976) 76 19 75, Fax. 34 (976) 76 18 61*

*<http://www.cps.unizar.es/deps/DIIS/robot/>*

---

## Abstract

The validation of matching hypotheses using Mahalanobis distance is extensively utilized in robotic applications, and in general data-association techniques. The Mahalanobis distance, defined by the innovation and its covariance, is compared with a threshold defined by the chi-square distribution to validate a matching hypothesis; the validation test is a time-consuming operation. This paper presents an efficient computation for this test.

The validation test implies a computational overhead for two reasons: first, because of covariance matrix inversion, and second because the computation of the covariance and innovation terms are also expensive operations, in fact, more expensive than the inversion itself.

The method described here can be summarized as an incremental, non-decreasing computation for the Mahalanobis distance; if the incrementally computed value exceeds the threshold then the computation is stopped. The elements of covariance and innovation, and the matrix inversion itself, are only computed if they are used; progressivity is the major advantage of the method. The method is based upon the square-root-free Cholesky's factorization.

In addition, a lower bound for the Mahalanobis distance is proposed. This lower bound has two advantages: it can be progressively computed, and it is greater than the classical trace lower bound.

*Key words:* Mahalanobis distance, data-association, gate validation, matching.

---

## 1 Introduction

In robotic applications, real-time aspects are important, and efficient algorithms are needed to implement both recognition and robot reactivity from recognition. Focusing the attention on the issue dealt with in this paper, in many robotic applications a combination and fusion of error-prone or imprecise information obtained from multiple sensors is achieved. Many current works use a probabilistic representation of the uncertainty and a Kalman Filter formulation to fuse and validate sensor information, and to estimate geometric parameters describing the environment. In (Ayache and Faugeras, 1987, Tardós 1992), nonlinear implicit equations are used to represent the relationships between sensor measurements, and the geometric parameters representing the geometric features and calibration parameters involved, or to represent geometric constraints such as perpendicularity, intersection, and parallelism. A chi-square test is proposed here so as to accept new measurements before integrating them. Durrant-White (1988) also uses implicit equations to represent geometric features, and the Mahalanobis distance test is carried out for clustering different measurements in a general multisensor system. In (Crowley 1995), a general formulation for linear and nonlinear measurement equations is presented, and applied to navigation and perception for autonomous mobile robots. The Mahalanobis distance is used for matching observations and state prediction. Castellanos et al (1996) also use this matching framework to pair and fuse laser and vision observations of a 3D segment for mobile robot navigation.

The matching between the estimated features and the observations detected after a sensing operation is determined using data-association techniques. The use of data association for matching allows one to benefit from using a well-founded theory (Bar-Shalom and Fortmann 1988). Data-association algorithms include a hypothesis-validation step, based on Mahalanobis distance; this step is one of the most time-consuming operation of the matching process (Cox 1993). This validation step is known also as “gate validation”.

After a sensing operation,  $M$  feature location estimates, and  $N$  measurements, are available. The problem is how to associate each measurement with a feature estimate; spurious measurements and target initiation should be also considered. Initially, there are  $MN$  pairing hypotheses; however most of the data-association techniques perform a validation test for each pairing hypothesis in order to work with only a reduced set of validated hypotheses. The validation is performed using a statistical test based on the Mahalanobis distance (left-hand term in the inequality):

$$\mathbf{v}^T \mathbf{C}^{-1} \mathbf{v} \leq \chi^2. \quad (1)$$

The validation procedure is time-consuming because:

- Matrix inversion is a time-consuming operation.
- The computation of matrix  $\mathbf{C}$  and vector  $\mathbf{v}$  is in itself time-consuming due to linearizations.
- The test should theoretically be computed for  $MN$  hypotheses.

This paper presents a progressive method of computing the Mahalanobis distance. By progressive is meant that the Mahalanobis distance is computed incrementally as a non-decreasing quantity. If the incrementally computed value exceeds the  $\chi^2$  threshold, the computation is stopped. The elements of  $\mathbf{C}$ , and  $\mathbf{v}$ , and the matrix inversion itself, are computed only if they are used before the computation is stopped. The progressivity is the core idea of the proposal, because the rejected hypotheses can be detected at early stages, and hence the computational load is reduced.

From the progressive computation, a lower bound for the Mahalanobis distance can be deduced, allowing some hypotheses to be rejected without Mahalanobis distance computation. Orr (1992) proposed a lower bound based on trace of the matrix  $\mathbf{C}$ . The lower bound presented in this paper has two main advantages over the trace bound:

- It can be computed progressively, using the  $\mathbf{C}$  elements progressively; so if the hypothesis is rejected at an early stage only the elements actually utilized are computed.
- It is greater than the trace lower bound, implying a bigger rejection ratio.

In fact, in the experimental results below, a comparison between the two techniques is presented.

Several solutions have been proposed to reduce the computational load due to hypothesis validation. The proposed techniques are focused on the reduction of the number of hypotheses to be validated. Zhang and Faugeras (1992) proposed the use of “bucketing” to reduce the number of hypotheses when the estimated feature and the measurement are 3D segments. Collins and Uhlmann (1992) proposed a technique which significantly reduces the problem complexity, using multidimensional search trees. In any case, after the reduction of the hypotheses, the hypotheses should finally be validated, and the progressive test proposed in this paper will perform more efficiently than computing the Mahalanobis distance first and testing the threshold afterwards. Besides, the reduction of hypotheses is not easily performed when the observations are partial observations of the estimated feature, because they are related by a nonlinear implicit measurement equation.

The next section is devoted to presenting the notation and computation of the Mahalanobis distance using the  $\mathbf{LDL}^T$  factorization (Bierman 1977); it is

a square-root-free Cholesky’s factorization. Section 3 presents a description of how to perform the validation test progressively. Section 4 is devoted to computing a lower bound for the Mahalanobis distance. Next, Section 5 presents a geometrical interpretation for the lower bound, and Section 6 summarizes the computational cost for the proposed test, the non-progressive test and two lower bounds for the Mahalanobis distance. Section 7 shows the computation reduction on real sensor information. Finally in section 8 conclusions are drawn.

## 2 Problem statement

### 2.1 Notation

Lower-case letters will be used,  $\mathbf{v}$ ,  $\mathbf{y}$ ,  $\dots$ , for vectors; their components are represented by the corresponding lower-case letter with a simple subindex:  $v_i$ ,  $x_i$ ,  $y_i$ ,  $\dots$ . Bold upper-case letters  $\mathbf{C}$ ,  $\mathbf{G}$ ,  $\mathbf{H}$ ,  $\dots$  represent matrices; components of these matrices are represented by the corresponding lower-case letter and a double subscript:  $c_{ij}$ ,  $g_{ij}$ ,  $h_{ij}$ ,  $\dots$ .  $(\mathbf{LD})_{ij}$  represents the element  $ij$  for the matrix product  $\mathbf{LD}$ . Where necessary, the matrix dimension is expressed in a parenthesized subindex,  $\mathbf{H}_{(m \times n)}$ . The superscript  $T$  represents the transpose operation. The superscript  $-T$  stands for the transpose of the inverse of a matrix.

### 2.2 Computation of the Mahalanobis distance

The goal is to determine whether the following inequality is fulfilled:

$$\mathbf{v}^T \mathbf{C}^{-1} \mathbf{v} \leq \chi^2$$

provided that the matrix  $\mathbf{C}_{(n \times n)}$  is symmetric positive definite because it is a non-singular covariance matrix. This section is devoted to presenting an efficient method of Mahalanobis distance computation. This method is the basis for the definitive proposal presented in Section 3. The proposed method is based on  $\mathbf{LDL}^T$  factorization. This factorization is closely related with the  $\mathbf{UDU}^T$  factorization; in fact both of them were presented by Bierman (1977). The unique difference between them is that the former uses a lower triangular matrix  $\mathbf{L}$ , while the latter uses,  $\mathbf{U}$ , an upper triangular matrix. The  $\mathbf{UDU}^T$  factorization has been widely used for square-root filtering, an implementation of the Kalman filter which is computationally efficient and robust to numerical errors.

```

FUNCTION LDLT_factorization(C)
; INPUT: C symmetric positive definite matrix
; RETURN VALUE:
;           L lower triangular matrix
;           D-1 inverse of the diagonal matrix D
BEGIN
  FOR  $i = 1$  TO  $n$  DO
    FOR  $j = 1$  TO  $i - 1$  DO
       $(\mathbf{LD})_{ij} = \left( c_{ij} - \sum_{k=1}^{k < j} l_{ik} (\mathbf{LD})_{jk} \right)$ 
       $l_{ij} = (\mathbf{LD})_{ij} d_{jj}^{-1}$ 
    END FOR
     $d_{ii}^{-1} = \left( c_{ii} - \sum_{k=1}^{k < i} l_{ik} (\mathbf{LD})_{ik} \right)^{-1}$ 
  END FOR
  RETURN L, D-1
END

```

Fig. 1.  $\mathbf{LDL}^T$  factorization algorithm.

It is a well-known fact from numerical analysis (Burden et al 1981), that the product:  $\mathbf{C}^{-1}\mathbf{v} = \mathbf{y}$  is more efficiently computed solving the linear system  $\mathbf{C}\mathbf{y} = \mathbf{v}$ . Besides, using a factorization method to determine the Mahalanobis distance is even more efficient than solving the previous linear system. The usual factorization method for symmetric positive definite matrices is Cholesky's factorization. However, the factorization  $\mathbf{LDL}^T$  (and  $\mathbf{UDU}^T$ ) is more suitable for the computation of Mahalanobis distances because, unlike Cholesky's factorization, it does not need  $n$  square roots ( $n$  is the dimension of matrix  $\mathbf{C}$ ).

Let  $\mathbf{C}_{(n \times n)}$  be a positive definite matrix; then it can be factorized as:

$$\mathbf{C} = \mathbf{LDL}^T \tag{2}$$

where:

$\mathbf{L}$  is a lower triangular matrix, whose main diagonal elements are 1's.  
 $\mathbf{D}$  is a diagonal positive definite matrix.

Figure 1 presents the factorization algorithm. The algorithm computes  $\mathbf{L}$  and  $\mathbf{D}^{-1}$ .  $\mathbf{D}^{-1}$  is computed instead of  $\mathbf{D}$  because only the inverse of the diagonal elements are used in the algorithm and subsequent computations. The matrix product  $\mathbf{LD}$  is stored as an intermediate value.

step	ops. iteration $i$			total ops. dimension $n$		
	+/-	$\times$	$\div$	+/-	$\times$	$\div$
factor.	$\frac{i^2+i-2}{2}$	$\frac{i^2+3i-4}{2}$	1	$\frac{n^3-n}{6}$	$\frac{n^3+3n^2-4n}{6}$	$n$
forw sub.	$i-1$	$i-1$		$\frac{n^2-n}{2}$	$\frac{n^2-n}{2}$	
dot prod.	1	2		$n$	$n+n$	
total	$\frac{i^2+3i-2}{2}$	$\frac{i^2+5i-4}{2}$	1	$\frac{n^3+3n^2+2n}{6}$	$\frac{n^3+6n^2+5n}{6}$	$n$

Table 1

Operations to compute the Mahalanobis distance

Next, this section presents how to use (2) to compute the Mahalanobis distance:

$$\mathbf{v}^T \mathbf{C}^{-1} \mathbf{v} = \mathbf{v}^T (\mathbf{L} \mathbf{D} \mathbf{L}^T)^{-1} \mathbf{v} = \mathbf{v}^T \mathbf{L}^{-T} \mathbf{D}^{-1} \mathbf{L}^{-1} \mathbf{v}$$

so:

$$\mathbf{v}^T \mathbf{C}^{-1} \mathbf{v} = \mathbf{w}^T \mathbf{D}^{-1} \mathbf{w} = \sum_{k=1}^n w_k^2 d_{kk}^{-1} \quad (3)$$

where  $\mathbf{w} = \mathbf{L}^{-1} \mathbf{v}$  is computed solving the linear system:

$$\mathbf{L} \mathbf{w} = \mathbf{v}.$$

As  $\mathbf{L}$  is lower diagonal with 1's in its diagonal, the previous system is solved by a *forward substitution*:

$$w_i = v_i - \sum_{k=1}^{k<i} l_{ik} w_k. \quad (4)$$

Table 1 shows the total of the operations required for the Mahalanobis distance computation, split into the following steps: factorization, i.e. computation of  $\mathbf{L}$  and  $\mathbf{D}^{-1}$ ; forward substitution, expression (4); and dot product, expression (3). The operations for step  $i$  of algorithm 1, and the total operations for a dimension- $n$  problem are shown.

### 2.3 Computation of the covariance matrix

The validation test, usually comes from the linearization of a nonlinear measurement equation. The focus here is on solving problems using a implicit

measurement equation (Ayache and Faugeras 1987, Tardós 1992):

$$0 = \mathbf{f}(\mathbf{x}, \mathbf{z})$$

where  $\mathbf{x}$  is the state and  $\mathbf{y}$  the measurement. Linearizing around the point  $\mathbf{x} = 0, \mathbf{z} = 0$ :

$$0 \approx \mathbf{v} + \mathbf{H}\mathbf{x} + \mathbf{G}\mathbf{z}$$

where:

$$\mathbf{H} = \left. \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{z})}{\partial \mathbf{x}} \right|_{(\mathbf{x}=0, \mathbf{z}=0)} \quad \mathbf{G} = \left. \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{z})}{\partial \mathbf{z}} \right|_{(\mathbf{x}=0, \mathbf{z}=0)}$$

$$\mathbf{v} = \mathbf{f}|_{(\mathbf{x}=0, \mathbf{z}=0)}.$$

So the covariance matrix  $\mathbf{C} = \text{Cov}(\mathbf{v})$  is:

$$\mathbf{C}_{(n \times n)} = \mathbf{H}\mathbf{P}\mathbf{H}^T + \mathbf{G}\mathbf{Q}\mathbf{G}^T \quad (5)$$

where  $\mathbf{P} = \text{Cov}(\mathbf{x})$  and  $\mathbf{Q} = \text{Cov}(\mathbf{z})$  are the covariance matrices corresponding to  $\mathbf{x}$  and  $\mathbf{z}$ . It should be pointed out that  $\mathbf{H}$  and  $\mathbf{G}$ , the linearization matrices, depend on both of the elements of the pairing hypothesis. Because of this there are different linearization matrices ( $\mathbf{H}$  and  $\mathbf{G}$ ) for each hypothesis, thus impeding the to use of the hypothesis reduction proposed by (Collins and Uhlmann 1992), because no explicit measurement equation is available.

Consider the first addend (the other matricial addend can be considered similarly):

$$\mathbf{H}_{(n \times m)}\mathbf{P}_{(m \times m)}\mathbf{H}^T. \quad (6)$$

As will be shown in Section 3.1, the complexity of the previous matrix product is (the reduction due to matrix symmetry has been considered):

$$n(m^2 + m) + \frac{n^2 - n}{2}m \quad \text{products} \quad (7)$$

$$n(m^2 + m) + \frac{n^2 - n}{2}m \quad \text{additions.}$$

Normally  $n < m$  because normally the state dimension is bigger than the measurement dimension. So,  $m = n$  is an optimistic assignment with regard to complexity, and thus covariance matrix computation implies  $O\left(3\frac{n^3}{2}\right)$  products and additions per matricial addend, which is bigger than the distance cost,

$O\left(\frac{n^3}{6}\right)$ . Additionally, the elements of  $\mathbf{H}$  and  $\mathbf{G}$  should be computed, and often imply time-consuming operations such as trigonometric functions or square roots.

### 3 Progressive validation

In order to reduce the computational load for hypothesis validation this paper proposes the use of the algorithm presented in Section 2.2 in a progressive way. The Mahalanobis distance is computed in a progressive and non-decreasing way, so as soon as the distance is detected as being bigger than  $\chi^2$ , the computation is stopped. The elements of the covariance matrix  $\mathbf{C}$  and  $\mathbf{v}$  can also be computed progressively, so only the elements that are actually used are computed. The progressive computation is an efficient technique because in many applications, most of the hypotheses (99% or even more) are rejected; so in most of the cases the computation is stopped before computing the full Mahalanobis distance.

This section is devoted to presenting the progressive algorithm for the Mahalanobis distance computation. Section 3.1 is devoted to the progressive computation of the covariance matrix.

The core idea of the progressive algorithm is that the Mahalanobis distance is computed as the dot product (see (3)):

$$\mathbf{w}^T \mathbf{D}^{-1} \mathbf{w} = \sum_{i=1}^n w_i^2 d_{ii}^{-1}.$$

As  $\mathbf{D}$  is positive definite, all the addends of the summation are positive. Thus, it is computed as an addition of positive terms. The elements of  $\mathbf{w}$  and  $\mathbf{D}^{-1}$  can be computed progressively, so the Mahalanobis distance can be computed progressively as a non-decreasing quantity, in  $n$  steps;  $n$  being the dimension of the covariance matrix.

The main steps of the progressive algorithm are also factorization, forward substitution and dot product. A detailed algorithm is presented in Fig. 2. The complexity of the algorithm is shown in Table 1.

#### 3.1 Progressive computation for the covariance matrix

As has been mentioned, one of the main advantages of the proposed algorithm is that covariance matrix elements are computed only when necessary. This

```

FUNCTION progressive_validation(P, Q, v,  $\chi^2$ )
; INPUT: P, Q covariance matrices
;          v vector to validate
;           $\chi^2$  threshold for the statistical test
; RETURN VALUE:
;          IF  $\mathbf{v}^T \mathbf{C}^{-1} \mathbf{v} > \chi^2$  THEN -1.0
;          ELSE  $\mathbf{v}^T \mathbf{C}^{-1} \mathbf{v}$ 
BEGIN
  maha_dist = 0.0
  FOR  $i = 1$  TO  $n$  DO
    ;factorization
    {H, G} = compute_i_row_linearization(H, G,  $i$ )
    C = compute_i_row_covariance(P, Q, C, H, G,  $i$ )
    FOR  $j = 1$  TO  $i - 1$  DO
       $(\mathbf{LD})_{ij} = \left( c_{ij} - \sum_{k=1}^{k < j} l_{ik} (\mathbf{LD})_{jk} \right)$ 
       $l_{ij} = (\mathbf{LD})_{ij} d_{jj}^{-1}$ 
    END FOR
     $d_{ii}^{-1} = \left( c_{ii} - \sum_{k=1}^{k < i} l_{ik} (\mathbf{LD})_{ik} \right)^{-1}$ 
     $w_i = v_i - \sum_{k=1}^{k < i} l_{ik} w_k$  ;forward substitution
    maha_dist = maha_dist +  $w_i^2 d_{ii}^{-1}$ ; dot prod.
    IF maha_dist  $> \chi^2$  THEN RETURN -1.0
  END FOR
  RETURN maha_dist
END

```

Fig. 2. Progressive validation (PV) algorithm

section presents some remarks and how to do this progressive computation efficiently.

Consider again the first addend of expression (5) (the other addend can be treated similarly). The product can be expressed as:

$$\mathbf{H}_{(n \times m)} \mathbf{P}_{(m \times m)} \mathbf{H}^T = \begin{pmatrix} \mathbf{h}_1^T \\ \vdots \\ \mathbf{h}_i^T \\ \vdots \\ \mathbf{h}_n^T \end{pmatrix} \mathbf{P}(\mathbf{h}_1 \dots \mathbf{h}_j \dots \mathbf{h}_n)$$

so:

$$c_{ij} = \mathbf{h}_i^T \mathbf{P} \mathbf{h}_j.$$

As  $\mathbf{C}$  is a symmetric matrix, only the terms  $j \leq i$  have to be computed. In addition, the algorithm uses one row after another progressively, and uses only the computed terms in each row ( $i \leq j$ ).

When a term  $c_{ij}$  is the first one computed in that row, two matrix products should be computed, first

$$\mathbf{h}_i^T \mathbf{P} \quad m^2 \text{ products, } m^2 \text{ additions}$$

and then

$$\left( \mathbf{h}_i^T \mathbf{P} \right) \mathbf{h}_j \quad m \text{ products, } m \text{ additions}$$

where the complexity is shown next to each product. For any additional element  $c_{ik}$  belonging to the same row, only the product:

$$\left( \mathbf{h}_i^T \mathbf{P} \right) \mathbf{h}_k \quad m \text{ products, } m \text{ additions}$$

has to be performed. Without loss of generality one can consider that the  $n$  diagonal terms are computed first, which implies

$$n(m^2 + m) \text{ products, } n(m^2 + m) \text{ additions.}$$

The  $\frac{n^2-n}{2}$  off-diagonal terms imply  $\frac{n^2-n}{2}m$  additions and multiplications. So the total complexity is:

$$\begin{aligned} n(m^2 + m) + \frac{n^2-n}{2}m & \text{ products} \\ n(m^2 + m) + \frac{n^2-n}{2}m & \text{ additions.} \end{aligned} \tag{8}$$

It also should be noted that the computation of the elements of a vector  $\mathbf{h}_i$  can be time-consuming. As  $j \leq i$ , the  $\mathbf{h}_j$  vectors can be reused from the computation of the previous rows.

## 4 One-component bound

From the progressive validation, a lower bound for the Mahalanobis distance can be deduced:

$$\mathbf{v}^T \mathbf{C}^{-1} \mathbf{v} = \sum_{i=1}^n w_i^2 d_{ii}^{-1} \geq w_1^2 d_{11}^{-1} = \frac{v_1^2}{c_{11}}. \quad (9)$$

The deduction is taking straight from the algorithm of Fig. 2 after the first iteration:

$$\begin{aligned} \text{maha\_dist} &= w_1^2 d_{11}^{-1} \\ w_1 &= v_1 \\ d_{11}^{-1} &= c_{11}^{-1}. \end{aligned}$$

It should be noted that the vector  $\mathbf{v}$  can be reordered; this implies only permutations in the rows and columns of  $\mathbf{C}$  to achieve the covariance matrix for the reordered vector. Reordering  $\mathbf{v}$ , every  $v_i$  can be brought into the first row so, inequality (9) yields for every  $i$ :

$$\mathbf{v}^T \mathbf{C}^{-1} \mathbf{v} \geq \frac{v_i^2}{c_{ii}} \quad i = 1, \dots, n.$$

So, the greatest lower bound that can be obtained is:

$$\mathbf{v}^T \mathbf{C}^{-1} \mathbf{v} \geq \max_{\{i=1..n\}} \left\{ \frac{v_i^2}{c_{ii}} \right\}. \quad (10)$$

The previous lower bound will be referred to as the ‘‘One Component Bound’’, (OCB).

As the OCB implies a low computational load, it can be used to test if a hypothesis can be rejected before computing the Mahalanobis distance. If the OCB is greater than the  $\chi^2$  threshold, then the hypothesis can be rejected; otherwise, the Mahalanobis distance must be computed to test whether the hypothesis can be eventually accepted. The use of a lower bound is normally useful because many hypotheses can be rejected without computing the Mahalanobis distance. Its main pitfall is that when a hypothesis is not rejected, the Mahalanobis distance has to be computed, and so for the accepted hypotheses, the OCB is an extra cost.

Orr (1992) proposed another lower bound, which will be referred to here as the “Trace Bound” (TB).

$$\mathbf{v}^T \mathbf{C}^{-1} \mathbf{v} \geq \frac{\sum_{i=1}^n v_i^2}{\sum_{i=1}^n c_{ii}} = \frac{\sum_{i=1}^n v_i^2}{\text{trace}(\mathbf{C})}. \quad (11)$$

As the following two propositions will prove, the OCB (10) is a greater lower bound than the TB (11), i.e.:

$$\mathbf{v}^T \mathbf{C}^{-1} \mathbf{v} \geq \max_{\{i=1 \dots n\}} \left\{ \frac{v_i^2}{c_{ii}} \right\} \geq \frac{\sum_{i=1}^n v_i^2}{\sum_{i=1}^n c_{ii}}. \quad (12)$$

**Proposition 1** *Let be  $a, b, c, d > 0$  then:*

$$\frac{a}{b} \geq \frac{c}{d} \Rightarrow \frac{a}{b} \geq \frac{a+c}{b+d} \geq \frac{c}{d}. \quad (13)$$

*The equality case yields:*

$$\frac{a}{b} = \frac{c}{d} \Rightarrow \frac{a+c}{b+d} = \frac{a}{b} = \frac{c}{d}.$$

**PROOF.** *Since:*

$$\begin{aligned} \frac{a}{b} \geq \frac{c}{d} &\Rightarrow a \geq \frac{bc}{d}, \\ \frac{a+c}{b+d} &\geq \frac{\frac{bc}{d} + c}{b+d} = \frac{c(b+d)}{d(b+d)} = \frac{c}{d}. \end{aligned}$$

*Analogously:*

$$\frac{a}{b} \geq \frac{c}{d} \Rightarrow c \leq \frac{ad}{b},$$

*and then:*

$$\frac{a+c}{b+d} \leq \frac{a + \frac{ad}{b}}{b+d} = \frac{a(b+d)}{b(b+d)} = \frac{a}{b}.$$

**Proposition 2** *Let be  $\mathbf{v} \in \Re^n$  and  $\mathbf{C}$  a definite positive matrix of dimension  $n$ . Without loss of generality, consider that the values  $\left\{ \frac{v_i^2}{c_{ii}} \right\} \quad i = 1 \dots n$  can be sorted as:*

$$\frac{v_1^2}{c_{11}} \leq \frac{v_2^2}{c_{22}} \leq \dots \leq \frac{v_n^2}{c_{nn}}.$$

Then:

$$\frac{v_n^2}{c_{nn}} \geq \frac{\sum_{i=1}^n v_i^2}{\sum_{i=1}^n c_{ii}} = \frac{\sum_{i=1}^n v_i^2}{\text{trace}(\mathbf{C})}.$$

The equality case implies that if:

$$\frac{v_1^2}{c_{11}} = \frac{v_2^2}{c_{22}} = \dots = \frac{v_n^2}{c_{nn}}$$

then:

$$\frac{\sum_{i=1}^n v_i^2}{\sum_{i=1}^n c_{ii}} = \frac{v_1^2}{c_{11}} = \frac{v_2^2}{c_{22}} = \dots = \frac{v_n^2}{c_{nn}}.$$

**PROOF.** This proposition is demonstrated by induction:

- It is fulfilled for  $n = 1$ . This is trivial.
- Provided that is fulfilled for  $n$ ,

$$\frac{v_n^2}{c_{nn}} \geq \frac{\sum_{i=1}^n v_i^2}{\sum_{i=1}^n c_{ii}} \Rightarrow \frac{v_{n+1}^2}{c_{n+1\ n+1}} \geq \frac{\sum_{i=1}^n v_i^2}{\sum_{i=1}^n c_{ii}}$$

because  $\frac{v_{n+1}^2}{c_{n+1\ n+1}} \geq \frac{v_n^2}{c_{nn}}$ . Identifying in (13):

$$a \rightarrow v_{n+1}^2 \quad b \rightarrow c_{n+1\ n+1} \quad c \rightarrow \sum_{i=1}^n v_i^2 \quad d \rightarrow \sum_{i=1}^n c_{ii}$$

the conditions for proposition 1 are fulfilled and then,

$$\frac{v_{n+1}^2}{c_{n+1\ n+1}} \geq \frac{\sum_{i=1}^{n+1} v_i^2}{\sum_{i=1}^{n+1} c_{ii}}.$$

So (12) has been proved. Thus the OCB can reject more hypotheses before computing the Mahalanobis distance, than the TB.

It should be pointed out that the two lower bounds, the OCB and the TB, have an important computational cost due to the matrix covariance terms involved. Both bounds use  $c_{ii}$  elements, and all of those elements belong to different rows, so the computation of all of them implies (see section 3.1):  $n(m^2 + m)$  additions and multiplications per matricial addend. Additionally, all the elements for the linearization matrices and for vector  $\mathbf{v}$  should be computed.

However, OCB can be computed progressively. In fact, it is not necessary to compute all the  $\frac{v_i^2}{c_{ii}}$  terms, then select the maximum and then test whether it

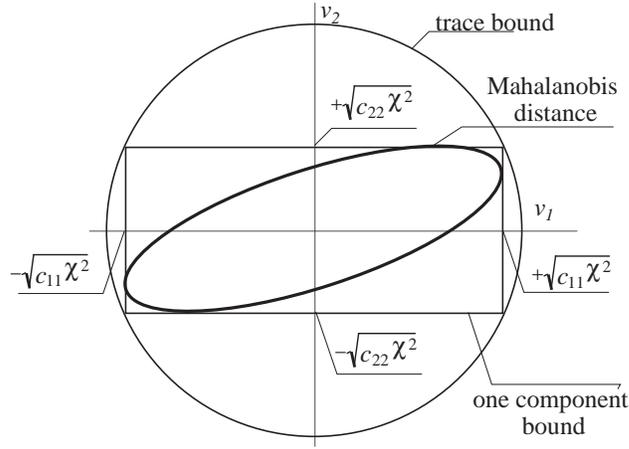


Fig. 3. Geometrical interpretation for the acceptance regions of the OCB, TB and the Mahalanobis distance test, for a dimension two example.

is greater than the  $\chi^2$  threshold. It is enough to compute the  $\frac{v_i^2}{c_{ii}}$  progressively until one of them is greater than the threshold.

## 5 Geometrical interpretation

This section is devoted to presenting a geometric interpretation of the acceptance regions associated with each of the lower bounds, i.e the OCB and the TB. It is a well-known fact that the acceptance region for the validation test based on the Mahalanobis distance:

$$\mathbf{v}^T \mathbf{C}^{-1} \mathbf{v} \leq \chi^2 \quad (14)$$

is the n-dimensional ellipsoid defined by the previous quadratic form. This section presents a geometric interpretation of the acceptance regions for the lower bounds:

**One-Component Bound (OCB).**- Collins (Collins and Uhlmann 1992) proves that the border for the region:

$$\left\{ \mathbf{v}^T = (v_1, \dots, v_n) \in \mathfrak{R}^n / \max_{\{i=1 \dots n\}} \left\{ \frac{v_i^2}{c_{ii}} \right\} \leq \chi^2 \right\} \quad (15)$$

is the n-dimensional bounding box tangent to the ellipsoid (14). The  $2n$  hyperplanes that define the bounding box are normal to the coordinate axis, and are defined by the equations:

$$v_i = \pm \sqrt{c_{ii}\chi^2} \quad i = 1 \dots n.$$

test	<b>H G</b> rows	<b>v</b> elem	<b>C</b> diag	<b>C</b> off-diag	operations
FV	$n$	$n$	$n$	$\frac{n^2-n}{2}$	opsFV( $n$ )
PV	$k$	$k$	$k$	$\frac{k^2-k}{2}$	opsFV( $k$ )
reject TB	$n$	$n$	$n$		$n *$ , $n - 1 +$ $1 \div$
accept TB	$n$	$n$	$n$	$\frac{n^2-n}{2}$	opsFV( $n$ ), $n *$ , $n - 1 +$ , $1 \div$
reject OCB	$k$	$k$	$k$		$k *$ , $k \div$
accept OCB	$n$	$n$	$n$	$\frac{n^2-n}{2}$	opsFV( $n$ ), $n *$ , $n \div$

Table 2  
Computational Cost Per Hypothesis

Figure 3 shows a graphical representation for a bidimensional example.  
**Trace Bound (TB).**- The acceptance region for TB:

$$\left\{ \mathbf{v}^T = (v_1, \dots, v_n) \in \mathfrak{R}^n / \frac{\sum_{i=1}^n v_i^2}{\text{trace}(\mathbf{C})} \leq \chi^2 \right\} \quad (16)$$

defines a  $n$ -dimensional sphere; the spheres radius is  $\sqrt{\text{trace}(\mathbf{C})\chi^2}$ . The previous sphere circumscribes the bounding box defined by (15). To prove this, observe that all the points of the bounding box are included in the sphere and its border, because of proposition 2:

$$\forall \mathbf{v} \in \mathfrak{R}^n / \max_{\{i=1..n\}} \left\{ \frac{v_i^2}{c_{ii}} \right\} \leq \chi^2, \Rightarrow \frac{\sum_{i=1}^n v_i^2}{\text{trace}(\mathbf{C})} \leq \chi^2$$

and the equality holds for the “corners” of the bounding box (intersection of  $n$  hyperplanes):

$$v_i = \sqrt{c_{ii}\chi^2} \quad i = 1 \dots n \Rightarrow \frac{v_i^2}{c_{ii}} = \chi^2 \Rightarrow \frac{\sum_{i=1}^n v_i^2}{\text{trace}(\mathbf{C})} = \chi^2$$

so the corners belong to the sphere border. See Fig.3 for a bidimensional example.

## 6 Computational cost

This section is devoted to presenting a summary of the computational cost of validating a hypothesis when using four different techniques. The cost is split into five components: (1) Number of rows of the linearization matrices (**H** and **G**) that should be computed, (2) number of elements of the innovation vector,

$\mathbf{v}$ , that should be computed, (3) number of diagonal elements of covariance matrix  $\mathbf{C}$  that should be computed, (4) number of off-diagonal elements of  $\mathbf{C}$  that must be computed, and (5), arithmetic operations involved. Table 2 summarizes the computational cost; the expression  $\text{opsFV}(m)$  stands for the number of operations required to perform a full validation for a dimension  $m$  problem (see Table 1). Next the cost for each validation technique is detailed:

**Full Validation (FV).**- First, the Mahalanobis distance is computed, and then this is tested to see if it is greater than the threshold. The computational cost is fixed for all hypotheses, irrespective of whether or not they are validated.

**Progressive Validation (PV).**- The algorithm is applied progressively until the progressive distance is greater than the  $\chi^2$  threshold, or the total distance is computed; so, the total cost for a hypothesis depends on the iteration,  $k$ , at which the progressive computation is stopped.

**Trace Bound (TB).**- First the TB is computed. If it is greater than the  $\chi^2$  threshold, the hypothesis is rejected; otherwise the full validation is computed to determine if the hypothesis is definitively rejected. The computational cost for a hypothesis depends on whether it is rejected by the trace test, or not.

**One-Component Bound (OCB).**- First the OCB is computed. If it is greater than the  $\chi^2$  threshold, the hypothesis is rejected; otherwise the full validation is computed to determine whether the hypothesis is definitively rejected. The OCB is computed progressively, so the cost of a hypothesis depends on the step,  $k$ , at which it is rejected, or on whether the full tests is computed.

## 7 Experimental results

This section is devoted to presenting an example of the simplification that can be achieved following the proposed progressive validation (PV) and the proposed one-component lower bound (OCB). The system of the example is a trinocular stereo system, used to determine the correspondences between the straight image segments in three images (see Montiel et al (1995) for more details); the example processes real sensor information. Two experiments are considered:

**Matching in the second image.** The matches between the segments in the first and the second images are computed. After the generation of hypotheses, 69117 hypotheses are selected to be validated using (1). 3922 hypotheses are finally validated, so the acceptance ratio is 5.67%.

**Matching in the third image.** The matches for the third image segments are considered. After hypothesis generation, 160464 hypotheses are consid-

ered, to be validated using (1). 257 hypotheses are finally validated, so the acceptance ratio is 0.16%.

These two examples are presented to show the behavior of the different validation methods with respect to the acceptance ratio. For each experiment, two cases for computing the covariance matrix are considered:

- (1) The covariance matrix obtained through the linearization (see Section. 2.3):

$$\mathbf{C}_{(3 \times 3)} = \mathbf{H}_{(3 \times 3)} \mathbf{P}_{(3 \times 3)} \mathbf{H}^T + \mathbf{G}_{(3 \times 11)} \mathbf{Q}_{(11 \times 11)} \mathbf{G}^T$$

In this example,  $\mathbf{Q}$  is block diagonal; considering that:

$$\mathbf{G} = \begin{bmatrix} \mathbf{J}_{(3 \times 5)} & \mathbf{K}_{(3 \times 6)} \end{bmatrix} \quad \mathbf{Q} = \begin{bmatrix} \mathbf{R}_{(5 \times 5)} & 0 \\ 0 & \mathbf{S}_{(6 \times 6)} \end{bmatrix}$$

the  $\mathbf{C}$  computation can be expressed as:

$$\mathbf{C} = \mathbf{H}_{(3 \times 3)} \mathbf{P}_{(3 \times 3)} \mathbf{H}^T + \mathbf{J}_{(3 \times 5)} \mathbf{R}_{(5 \times 5)} \mathbf{J}^T + \mathbf{K}_{(3 \times 6)} \mathbf{S}_{(6 \times 6)} \mathbf{K}^T$$

which is the linearization used by the real system. From complexity analysis presented in Section 3.1, the complexity for each diagonal term is 252 additions and 252 products; for each additional off-diagonal term in the same row 42 additions and products are necessary. As this cost is bigger than the dimension-3 full validation (12 additions, 16 products and 2 divisions), the complexity analysis for this case is approximated by the complexity of the covariance computation. Experimental results validate this simplification because the time required for validation is negligible with respect to covariance computation.

- (2) The covariance matrix is readily available, without any additional computational cost. The summary proposed in Section 6 is used to determine the complexity, but removing the terms related to covariance computation and linearization; i.e. using the column “operations” of Table 2.

These two cases are presented to show the importance of the matrix covariance computation in nonlinear systems.

For each of the four cases (image 2 and image 3, with and without consideration of covariance computing) the validation has been done using four methods: FV, PV, OCB and TB. For each experiment, the theoretical complexity is determined, expressed as the number of arithmetic operations to be performed. The execution time of the corresponding algorithm on a computer Sun Sparc 20, 150MHz is also shown. In order to compare the results, the reduction factor with respect to FV is computed for the rest of the validation techniques. Table 3 shows a summary of the results.

In both images 2 and 3, the rejection rate for the OCB is bigger than for the TB. This validates experimentally the benefits of using a greater lower bound. The high rejection rate achieved for the OCB before the last step can also be seen. Note that the PV rejects more hypotheses than the OCB; this can be seen by considering the accumulated number of rejected hypotheses at each step.

Next, this paper will focus on the validation, considering the covariance computation cost. In all the experiments the computation time taking into account the computation of covariance is more than one order of magnitude greater than the computation of the validation, showing how important is to avoid the computation of the covariance matrix elements. Also, due to the computational load for the covariance, the TB cannot reduce the load, because it needs all the diagonal elements for every hypothesis. Unlike the TB, the PV and the OCB can do the validation twice faster in image 3 than in image 2, because the rejection ratio is higher in image 3, and most of the hypothesis can be rejected at an early stage. The reduction predicted by the complexity analysis agrees with the computation times measured. Progressive methods (PV,OCB) perform better than the TB, and of course better than the FV. There are not big differences between the OCB and the PV. However, the PV performs better when the acceptance ratio is not very low because, unlike the OCB, it does not add any extra computation for the accepted hypotheses.

Next, consider the cost when the covariance computation is not taken into account. When the acceptance ratio is high (image 2), the OCB performs better than the TB because it can reject more hypotheses without distance computation; remember that the trace bound is always greater than the one-component bound (see Section 4). Because of the high acceptance ratio, the PV performs better than the OCB; this is because the validated hypotheses have no additional extra cost. When the acceptance ratio is low, there are no big differences between the three methods (OCB, TB, PV). Despite the number of additions and products being greatly reduced for the OCB, TB and PV with respect to the FV, it is not the same for divisions; because of that the computation time is only reduced by a factor 0.4 (image 3 without covariance computation) with respect to full validation.

## 8 Conclusions

Computation of the Mahalanobis distance is a time-consuming operation because of matrix inversion, and because of the computation of the covariance matrix itself. Due to the linearizations, the most important computational load is the computation of the covariance matrix.

IMAGE 2 MATCHING				
total hypo. 69117, accept. 3922, accept. ratio 5.67%				
	FV	PV	TB	OCB
rejected step 1		40.68%		40.68%
rejected step 2		38.22%		23.79%
rejected step 3		15.43%	52.07%	8.83%
considering matrix covariance computation				
$\mu$ sec.	29.3	15.3	27.3	16.8
reduct. factor	1	0.53	0.93	0.57
adds/prods.	882	497	816	525
reduct. factor	1	0.56	0.92	0.59
without considering matrix covariance computation				
$\mu$ sec.	0.8	0.53	0.74	0.58
reduct. factor	1	0.67	0.93	0.72
adds	12	4.06	9.75	3.20
prod.	16	6.08	10.67	6.22
div	3	1.80	2.24	2.75
reduct. factor +	1	0.33	0.81	0.26
reduct. factor *	1	0.37	0.66	0.38
reduct. factor $\div$	1	0.60	0.81	0.91

IMAGE 3 MATCHING				
total hypo. 160464, accept. 257, accept. ratio 0.16%				
	FV	PV	TB	OCB
rejected step 1		84.54%		84.54%
rejected step 2		15.16%		15.04%
rejected step 3		0.14%	99.19%	0.16 %
considering matrix covariance computation				
$\mu$ sec.	29.3	8.92	24.1	8.55
reduct. factor	1	0.30	0.82	0.29
adds/prods.	882	298	757	292
reduct. factor	1	0.33	0.85	0.33
without considering matrix covariance computation				
$\mu$ sec.	0.8	0.35	0.35	0.3
reduct. factor	1	0.44	0.44	0.37
adds	12	1.70	4.10	0.03
prod.	16	1.80	3.13	1.20
div	3	1.16	1.02	1.17
reduct. factor +	1	0.14	0.34	0.002
reduct. factor *	1	0.11	0.19	0.07
reduct. factor $\div$	1	0.38	0.34	0.38

Table 3  
average per hypothesis complexity

In order to detect efficiently whether the Mahalanobis distance is greater than a threshold, it is important to compute the Mahalanobis distance, or a lower bound to it, progressively, and to stop the computation as soon as it is detected that the distance is greater than the threshold.

This paper has presented two progressive computations, which have similar performance:

**One-Component bound.-** This computes a lower bound for the Mahalanobis distance, avoiding its computation. It has two main advantages over the trace bound, normally used in the literature: (1) the computation is progressive, (2) it determines a greater lower bound. The use of the lower

bound is interesting when the rejection rate is very high.

**Progressive validation.-** This computes the Mahalanobis distance progressively, so that the computation can be stopped as soon as the distance is detected to be bigger than the threshold. Besides, the progressiveness incurs, no any additional cost over the Mahalanobis distance computation in any hypothesis (accepted or rejected). This test has good performance in every situation, with high or low rejection rates, and with or without covariance computation.

The complexity reduction for a dimension-3 example is in the interval  $[0.3, 0.4]$ , with a 0.16% acceptance ratio; and in the interval  $[0.5, 0.7]$  with a %5.67 acceptance ratio. For problems of higher dimension, a better reduction can be achieved.

## Acknowledgments

This work was supported in part by the Spanish CICYT project TAP94-0390.

## References

- [1] N. Ayache and O.D. Faugeras. Maintaining representations of the environment of a mobile robot. In R. Bolles and B. Roth, editors, *Robotics Research: The Fourth International Symposium*. MIT Press, 1987.
- [2] Y. Bar-Shalom and T.E. Fortmann. *Tracking and Data Association*, volume 179 of *Mathematics in Science and Engineering*. Academic Press, INC., San Diego, 1988.
- [3] G.J. Bierman. *Factorization Methods for Discrete Sequential Estimation*. Academic Press, 1977.
- [4] R.L. Burden, J.D. Faires, and A.C. Reinolds. *Numerical Analysis*. Prindle, Weber & Schmidt, 1981.
- [5] J.A. Castellanos, J. Neira, O. Strauss, and J.D. Tardós. Detecting high level features for mobile robot localizacion. In *IEEE Int. Conf. Multisensor Fusion and Integration for Intelligent Systems*, Washington, Dec 1996.
- [6] J.B. Collins and J.K Uhlmann. Efficient gating in data association with multivariate gaussian distributed states. *IEEE Transactions on Aerospace and Electronic Systems*, 28(3):909–916, July 1992.
- [7] J.C. Cox. A review of statistical data association techniques for motion correspondence. *Int. Journal of Computer Vision*, 10(1):53–66, 1993.

- [8] J. Crowley. Mathematical foundations of navigation and perception for an autonomous mobile robot. In *Int. Workshop on Reasoning with uncertainty in Robotics*, Dec. 1995.
- [9] H. F. Durrant-Whyte. Sensor models and multisensor integration. *Int. Journal of Robotics Research*, 7(6):97–113, Dec 1988.
- [10] J.M Martínez Montiel, Z. Zhang, and L. Montano. Segment-based structure from an imprecisely located moving camera. In *IEEE Int. Symposium on Computer Vision*, pages 182–187, Florida, November 1995.
- [11] M.J.L. Orr, J. Hallam, and R.B Fisher. Fusion through interpretation. In *Second European Conference on Computer Vision*, 1992.
- [12] J.D. Tardós. Representing partial and uncertain sensorial information using the theory of symmetries. In *IEEE Int. Conf. on Robotics and Automation*, pages 1799–1804, Nice, France, May 1992.
- [13] Z. Zhang and O. Faugeras. Three-dimensional motion computation and object segmentation in a long secuence of stereo frames. *Int. Journal of Computer Vision*, 7(3):211–241, 1992.