Globally Convergent Autocalibration

A. Benedetti*
Division of Engineering and Applied Science
California Institute of Technology
Pasadena, CA 91125
arrigo@vision.caltech.edu

A. Busti, M. Farenzena, and A. Fusiello,
Dipartimento di Informatica
Università degli studi di Verona
I-37134 Verona, Italy
fusiello@sci.univr.it

Abstract

Existing autocalibration techniques use numerical optimization algorithms that are prone to the problem of local minima. To address this problem, we have developed a method where an interval branch-and-bound method is employed for numerical minimization. Thanks to the properties of Interval Analysis this method is guaranteed to converge to the global solution with mathematical certainty and arbitrary accuracy, and the only input information it requires from the user is a set of point correspondences and a search box. The cost function is based on the Huang-Faugeras constraint of the fundamental matrix. A recently proposed interval extension based on Bernstein polynomial forms has been investigated to speed up the search for the solution. Finally, some experimental results on synthetic images are presented.

1 Introduction

The goal of Computer Vision is to compute properties (mainly geometric) of the three-dimensional world from images. One of the challenging problems of Computer Vision is to *reconstruct* a three-dimensional model of the scene from a moving camera. Most of the earlier studies in the field assume that the intrinsic parameters of the camera (focal length, image center and aspect ratio) are known. Computing camera motion in this case is a well known problem for which several methods are available (see [12] for a review). Given all the parameters of the camera, reconstruction is straightforward.

However, there are situations wherein the intrinsic parameters are unknown and the camera is not accessible (e.g. when using stock footage). In these cases the only information one can exploit is contained in the video sequence itself.

The classical approach to *autocalibration* (or *self-calibration*), in the case of a single moving camera with constant but unknown intrinsic parameters, is based on the Kruppa equations [19], which have been found to be very sensitive to noise [16], possibly due to the instability in the computation of the epipole [15]. Indeed, formulations which avoid the epipole seems to be more stable [7, 15].

Other methods [25, 8, 30], based on the *stratification* approach, upgrade a projective reconstruction to an Euclidean one without solving explicitly for the intrinsic parameters (see [1] for a review). The constant intrinsic parameters constraint has been released in [9, 24], by assuming that some other parameters are known.

Recently, Mendonça and Cipolla [20] presented an algorithm which directly recovers the intrinsic parameters from fundamental matrices, like the Kruppa equations, but it is simpler and copes with varying parameters.

Under the assumption that only the (varying) focal length is unknown, closed form and linear solutions can be obtained [5, 29]. In all the other cases the parameters comes from the solution of a system of polynomial equations or from the minimization of a non-linear function. In principle continuation (homotopy) techniques could be applied in the former case, though—in practice—iterative minimization techniques must be used [16], as homotopy algorithms are applicable only in the case of few displacements, and can give rise to bifurcation phenomena. When minimizing a non-linear function by gradient descent methods, convergence to the global minimum is not guaranteed: it depends on the initialization—for deterministic algorithms,—or it is guaranteed only in probability—for stochastic algorithms [28]. Quasi-linear approaches reduce the sensitivity to the initial guess [26, 30], but they do not solve the problem. The solutions of a simpler problem (only focal is unknown) have been used to initialize the minimization in [24, 10]. In [4] a stratified approach have been proposed, based on the direct evaluation of a dense sampling of the search space. Albeit some of these techniques are effective, none of the existing methods is provably convergent.

[*] Currently affiliated with KLA-Tencor, 5 Technology Dr., Milpitas (CA) 95035. E-mail: `Arrigo.Benedetti@kla-tencor.com`.

Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV 2003) 2-Volume Set
0-7695-1950-4/03 $17.00 © 2003 **IEEE**

IEEE COMPUTER SOCIETY

In this paper we introduce a method for autocalibration that is *guaranteed* to converge to the global minimum, regardless of the starting point. In the same spirit of [20, 15, 28], we compute directly the intrinsic parameters from fundamental matrices. We assume constant intrinsic parameters, but the technique is flexible and can be adapted to varying parameters as well.

The minimization algorithm is based on Interval Analysis (IA) [22], a branch of numerical analysis that has received increasing attention during the last decade and has been strangely overlooked by the computer vision community.

Classical numerical optimization methods for the multidimensional case start from some approximate trial points and sample the objective function at only a finite number of points. There is no way to guarantee that the function does not have some unexpectedly small values between these trial points, without making specific assumptions. On the contrary, IA optimization algorithms [3] evaluate the objective function over a continuum of points, including those points that are not finitely representable on the computer. They solve the optimization problem with the global *automatic result verification*, i.e. with the guarantee that the global minimizers have been found.

2 Background and problem formulation

Throughout this paper we will use the general projective camera model. Let $w = [x, y, z, 1]^{\mathsf{T}}$ be the homogeneous coordinates of a 3D point in the world reference frame. The homogeneous coordinates of the projected image point are given by[1] $m \backsim A\,[R|t]\,w$, where the 3×3 rotation matrix R and the translation vector t represent the camera's position and orientation (*extrinsic parameters*). The matrix A contains the *intrinsic parameters*, and has the following form:

$$A = \begin{bmatrix} \alpha_u & \gamma & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad (1)$$

where α_u, α_v are the *focal lengths* in horizontal and vertical pixels, respectively, (u_0, v_0) are the coordinates of the *principal point*, given by the intersection of the optical axis with the retinal plane, and γ is the *skew* factor, that models non-rectangular pixels.

Two conjugate points m and m' are related by the *fundamental matrix F*:

$$m'^{\mathsf{T}} F m = 0 \qquad (2)$$

The rank of F is in general two and, being defined up to a scale factor, it depends upon seven parameters. Its computation requires a minimum of eight conjugate points to obtain a unique solution. F depends on the intrinsic and extrinsic parameters in the following way [2]:

$$F \backsim A'^{-\mathsf{T}}([t]_\times R)A^{-1}. \qquad (3)$$

When conjugate points are in normalized coordinates ($A^{-1}m$), i.e., intrinsic parameters are known, one obtains the *essential matrix E*:

$$E \backsim [t]_\times R. \qquad (4)$$

The essential matrix encodes the rigid transformation between the two cameras, and it depends upon five independent parameters: three for the rotation and two for the translation up to a scale factor.

2.1 Autocalibration

In many practical cases, the intrinsic parameters are unknown and point correspondences are the only information that can be extracted from a sequence of images. *Autocalibration* consists in computing the intrinsic parameters, or—in general—recovering the Euclidean *stratum*, starting from point correspondences. In this section we will see which constraints are available for the autocalibration.

As we saw in Sec. 2, the epipolar geometry of two views is described by the fundamental matrix, which depends on seven parameters. Since the five parameters of the essential matrix are needed to describe the rigid displacement, two independent constraints are available for the computation of the intrinsic parameters from the fundamental matrix. Indeed, the essential matrix is characterized by the following Theorem [11, 5]:

Theorem 1 *A real 3×3 matrix E can be factored as the product of a nonzero skew-symmetric matrix and a rotation matrix if and only if E has two identical singular values and one zero singular value.*

It can be shown (see Sec. 3) that the conditions on the singular values are equivalent to:

$$\det(E) = 0 \quad \wedge \quad 2\,\mathrm{tr}(EE^{\mathsf{T}})^2 - \mathrm{tr}^2(EE^{\mathsf{T}}) = 0, \qquad (5)$$

which in turn is equivalent to the Kruppa equations [16]. The second clause of (5) can be decomposed in two independent polynomial constraints.

These equivalent constraints are algebraic interpretations of the so-called *rigidity constraint*, namely the fact that for any fundamental matrix F there exist two intrinsic parameters matrix A and A' and a rigid motion represented by t and R such that Eq. (3) is satisfied.

The autocalibration method by Mendonça and Cipolla is based on Theorem 1. Let F_{ij} the fundamental matrix relating views i and j (computed from point correspondences),

[1] $\backsim$ denotes equality up to a scale factor.

[2] $[t]_\times$ is the skew-symmetric matrix associated with the cross-product.

and let A_i and A_j be the respective (unknown) intrinsic parameter matrices. The cost function is

$$\chi(A_i,\, i \triangleq 1 \dots n) = \sum_{i=1}^{n} \sum_{j>n}^{n} w_{ij} \frac{{}^1\sigma_{ij} - {}^2\sigma_{ij}}{{}^1\sigma_{ij} + {}^2\sigma_{ij}}, \qquad (6)$$

where ${}^1\sigma_{ij} > {}^2\sigma_{ij}$ are the non zero singular values of

$$E_{ij} = A_i^{\mathsf{T}} F_{ij} A_j, \qquad (7)$$

and w_{ij} are normalized weight factors. In the general case of n views, the $n(n-1)/2$ fundamental matrices are not independent, neither are the $n(n-1)$ constraints that can be derived from them. It can be shown [24] that, if n_k parameters are known and n_c parameters are constant, the unknown intrinsic parameters can be computed provided that

$$n(n_k + n_c) \geq 8 + n_c. \qquad (8)$$

For example, if the intrinsic parameters are constant, three views are sufficient to recover them. If the skew is zero and the other parameters are varying, at least eight views are needed.

3 The cost function

The use of Eq. (6) as an optimization criterion has been considered, however it has posed several problems. First, its Hessian matrix is singular at the solution (${}^1\sigma_{ij} = {}^2\sigma_{ij}$), which can lead to higher run times of the optimization procedure [3]. Secondly, bounding the ranges of ${}^1\sigma_{ij}$ of an interval essential matrix with wide entries is not trivial, since it requires the solution of a min-max optimization problem. For these reasons we seek to minimize a cost function based on the equivalent constraint given by Eq. (5).

In the same spirit of the Mendonça-Cipolla algorithm, we minimize

$$\chi(A_i) \triangleq \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{ij} \frac{2\,\mathrm{tr}(E_{ij}{E_{ij}}^{\mathsf{T}})^2 - \mathrm{tr}^2(E_{ij}{E_{ij}}^{\mathsf{T}})}{\mathrm{tr}^2(E_{ij}{E_{ij}}^{\mathsf{T}})}. \qquad (9)$$

By using the property that the trace of a square matrix X is equal to the sum of its eigenvalues and the property that the eigenvalues of XX^{T} are equal to the squares of the singular values of X, we can write:

$$\mathrm{tr}(EE^{\mathsf{T}})^2 = \sum_{k=1}^{3} \sigma_k^4(E). \qquad (10)$$

Hence, the left hand side of Eq. (5) can be rewritten as

$$\begin{gathered} 2\,\mathrm{tr}(EE^{\mathsf{T}})^2 - \mathrm{tr}^2(EE^{\mathsf{T}}) = \\ 2(\sigma_1^4 + \sigma_2^4 + \sigma_3^4) - (\sigma_1^2 + \sigma_2^2 + \sigma_3^2)^2 = \\ (\sigma_1^2 - \sigma_2^2)^2 + \sigma_3^2(\sigma_3^2 - 2(\sigma_1^2 + \sigma_2^2)). \end{gathered} \qquad (11)$$

Therefore, provided that $\sigma_3 = 0$, the cost function expressed by Eq. (9) is the square of the Mendonça-Cipolla function (Eq. (6)). The essential matrix E is derived from the fundamental matrix via Eq. (7); if F is computed with an algorithm that enforces its rank to be two, then $\sigma_3 = 0$. As the left hand side of Eq. (5) is always positive, we do not need to take its square, as it would be required in a generic least squares problem. This is a very desirable property, since it reduces the order of the numerator and the denominator of the cost function from sixteen to eight.

In the following we assume that the intrinsic parameters of the camera are constant for the n views, i.e.

$$\chi(A_i) \triangleq \chi(A) \triangleq \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{ij} \left(\frac{2\,\mathrm{tr}(E_{ij}{E_{ij}}^{\mathsf{T}})^2}{\mathrm{tr}^2(E_{ij}{E_{ij}}^{\mathsf{T}})} - 1 \right). \qquad (12)$$

4 Global optimization using Interval Analysis

Interval Arithmetic [21] is an arithmetic defined on intervals, rather than on real numbers. In the beginning, Interval Arithmetic was mainly employed for bounding the measurement errors of physical quantities for which no statistical distribution was known. Later on it was leveraged to a broad new field of applied mathematics, aptly named Interval Analysis, where rigorous proofs are the consequence of numerical computations.

4.1 Notation and useful results

In the sequel of this section we shall follow the notation used in [14], where intervals are denoted by boldface, scalar quantities are denoted by lower case letters and vectors and matrices are denoted by upper case. Brackets "$[\cdot]$" will delimit intervals, while parentheses "$(\cdot)$" will delimit vectors and matrices. Underscores and overscores will represent respectively lower and upper bounds of intervals. An interval $\boldsymbol{x}$ is called *degenerate* when $\underline{\boldsymbol{x}} = \overline{\boldsymbol{x}} = x$. $\mathbb{I}\,\mathbb{R}$and $\mathbb{I}\,\mathbb{R}^n$ stand respectively for the set of real intervals and the set of interval vectors of dimension n. The midpoint of an interval $\boldsymbol{x}$ is denoted by $\mathrm{m}(\boldsymbol{x})$, and the vector whose entries are midpoints of the entries of $\boldsymbol{X} \in \mathbb{I}\,\mathbb{R}^n$ is denoted by $\mathrm{m}(\boldsymbol{X})$. The *width* of $\boldsymbol{x}$ is defined as $\mathrm{w}(\boldsymbol{x}) = \overline{\boldsymbol{x}} - \underline{\boldsymbol{x}}$. If $\boldsymbol{X} \in \mathbb{I}\,\mathbb{R}^n$ then $\mathrm{w}(\boldsymbol{X}) = \max\{\mathrm{w}(\boldsymbol{x}_i),\ i = 1, \dots, n\}$. If $f(x)$ is a function defined over an interval $\boldsymbol{x}$ then $\boldsymbol{f}^u(\boldsymbol{x})$ denotes the range of $f(x)$ over $\boldsymbol{x}$. Similarly, the range of $F : \mathbb{R}^n \to \mathbb{R}$ over $\boldsymbol{X}$ is denoted by $\mathbf{F}^u(\boldsymbol{X})$.

Interval arithmetic is an arithmetic defined on sets of intervals. If $\boldsymbol{x} = [\underline{\boldsymbol{x}}, \overline{\boldsymbol{x}}]$ and $\boldsymbol{y} = [\underline{\boldsymbol{y}}, \overline{\boldsymbol{y}}]$, a binary operation in the *ideal interval arithmetic* between $\boldsymbol{x}$ and $\boldsymbol{y}$ is defined as:

$$\begin{gathered} \boldsymbol{x} \,\mathrm{op}\, \boldsymbol{y} \triangleq \{x \,\mathrm{op}\, y \mid x \in \boldsymbol{y} \text{ and } y \in \boldsymbol{y}\}, \\ \text{for op} \in \{+, -, \times, \div\}. \end{gathered}$$

Thus, the ranges of the four elementary interval operations are exactly the ranges of the corresponding real operations. The operational definitions for the four elementary interval arithmetic operations are

$$
\begin{aligned}
\boldsymbol{x}+\boldsymbol{y} &\triangleq \left[\underline{\boldsymbol{x}}+\underline{\boldsymbol{y}}, \overline{\boldsymbol{x}}+\overline{\boldsymbol{y}}\right], \\
\boldsymbol{x}-\boldsymbol{y} &\triangleq \left[\underline{\boldsymbol{x}}-\overline{\boldsymbol{y}}, \overline{\boldsymbol{x}}-\underline{\boldsymbol{y}}\right], \\
\boldsymbol{x}\times\boldsymbol{y} &\triangleq \left[\min\left\{\underline{\boldsymbol{x}}\,\underline{\boldsymbol{y}}, \underline{\boldsymbol{x}}\,\overline{\boldsymbol{y}}, \overline{\boldsymbol{x}}\,\underline{\boldsymbol{y}}, \overline{\boldsymbol{x}}\,\overline{\boldsymbol{y}}\right\},\right. \\
&\qquad \left.\max\left\{\underline{\boldsymbol{x}}\,\underline{\boldsymbol{y}}, \underline{\boldsymbol{x}}\,\overline{\boldsymbol{y}}, \overline{\boldsymbol{x}}\,\underline{\boldsymbol{y}}, \overline{\boldsymbol{x}}\,\overline{\boldsymbol{y}}\right\}\right], \\
\frac{1}{\boldsymbol{x}} &\triangleq \begin{cases} [1/\overline{\boldsymbol{x}}, 1/\underline{\boldsymbol{x}}] & \text{if } \underline{\boldsymbol{x}} > 0 \\ [1/\underline{\boldsymbol{x}}, 1/\overline{\boldsymbol{x}}] & \text{if } \overline{\boldsymbol{x}} < 0 \end{cases} \quad (0 \notin [\underline{\boldsymbol{x}}, \overline{\boldsymbol{x}}]), \\
\boldsymbol{x}\div\boldsymbol{y} &\triangleq \boldsymbol{x}\times 1/\boldsymbol{y}.
\end{aligned}
$$

The above definitions imply the ability to perform the four elementary operations with arbitrary precision. When implemented in a digital computer, however, truncation errors occur, that may cause the result not to contain the result that would be obtained with ideal interval arithmetic. In order to avoid this effect, the result corresponding to the lower endpoint of the interval must be rounded down to the nearest machine number less than the mathematically correct result, and the upper endpoint must be rounded up to the nearest machine number greater than the mathematically correct result. This mode of operation, called *direct rounding*, is available on any machine supporting the IEEE floating point standard.
Our use of IA is motivated by the need to obtain bounds on the range of mathematical functions.

Definition 1 (Interval extension) *A function* $\boldsymbol{F} : \mathbb{I}\,\mathbb{R}^n \to \mathbb{I}\,\mathbb{R}$*is said to be an* interval extension *of* $F : \mathbb{R}^n \to \mathbb{R}$ *provided*

$$\mathbf{F}^u(\boldsymbol{X}) \subseteq \mathbf{F}(\boldsymbol{X})$$

for all intervals $\boldsymbol{X} \subset \mathbb{I}\,\mathbb{R}^n$ *within the domain of* $\boldsymbol{F}$ *[14].*

The *natural* interval extension of a function is obtained by replacing variables with intervals and executing all operations according to the rule above. However these bounds are usually too wide or pessimistic to be of value. The following definition characterizes how sharply interval extensions enclose the range of a function.

Definition 2 (Order α inclusion function) *Let* $\boldsymbol{F}(\boldsymbol{X})$ *be an interval extension of* $F : \mathbb{R}^n \to \mathbb{R}$ *evaluated over a box* $\boldsymbol{X}$*. We say that* $\boldsymbol{F}$ *is an* order α inclusion function *for* F *if there is a constant* K*, independent of the box* $\boldsymbol{X}$*, such that*

$$w(\boldsymbol{F}(\boldsymbol{X})) - w(\mathbf{F}^u(\boldsymbol{X})) \leq K w(\boldsymbol{X})^\alpha \tag{13}$$

for all boxes $\boldsymbol{X}$ *with* $w(\boldsymbol{X})$ *sufficiently small.*

It can be shown [14] that natural interval extensions are first order. Higher-order inclusion functions are key to the design of efficient global optimization algorithms, as we shall see in the next section.

Definition 3 (Interval Newton method) *Let* $f : \boldsymbol{x} \subset \mathbb{R} \to \mathbb{R}$ *be a function with continuous first derivative on* $\boldsymbol{x}$ *and let* $x \in \boldsymbol{x}$*. If* $\boldsymbol{f}'(\boldsymbol{x})$ *is any interval extension of the derivative of* f *over* $\boldsymbol{x}$*, then the operator*

$$\boldsymbol{N}(f; \boldsymbol{x}, x) \triangleq x - f(x)/\boldsymbol{f}'(\boldsymbol{x}) \tag{14}$$

is called the univariate interval Newton method.

It can be shown that if $\boldsymbol{N}(f; \boldsymbol{x}, x) \subset \boldsymbol{x}$, then there exist a unique solution of $f(x) = 0$ in $\boldsymbol{x}$. The interval Newton method provides a quadratically convergent iteration

$$\boldsymbol{x} \leftarrow \boldsymbol{N}(f; \boldsymbol{x}, x) \cap \boldsymbol{x}. \tag{15}$$

Any solution within $\boldsymbol{x}$ must also be within $\boldsymbol{N}(f; \boldsymbol{x}, x)$. This fact underlies the use of the interval Newton method to sharpen bounds of the solutions to nonlinear equations and global optimizers. If $0 \in \boldsymbol{f}'(\boldsymbol{x})$ the quotient in Eq. (14) is computed using the rules for extended interval division defined in [3]. Multivariate interval Newton methods can be defined as well [14].

4.2 Global optimization

The ability of Interval Analysis to compute bounds of the range of functions has been most successful in global optimization. IA algorithms are usually based on branch-and-bound schemes [3]. They start from an initial box $\boldsymbol{X}$ in which the global minimum is sought, subdivide $\boldsymbol{X}$ and store the sub-boxes in a list $\mathcal{L}$. Sub-boxes which are guaranteed not to contain a global minimizer are discarded, and the process is repeated recursively until the desired accuracy, defined by the width of the sub-boxes in the list, is achieved. The criteria used to delete boxes are based on rigorous bounds, therefore the global minimizer is never deleted even in the presence of rounding errors.
We employed an algorithm inspired by a recently proposed global optimization method [27], based on the Moore-Skelboe branch-and-bound algorithm and Bernstein polynomials for bounding the range of the objective function. A combination of several test have been used in our implementation. The *cut-off* test determines or improves an upper bound $\hat{F}$ of the global minimum of the objective function F (any value taken by F is an upper bound for the global minimum), and discards an interval $\boldsymbol{X}$ from $\mathcal{L}$ if $\boldsymbol{F}(\boldsymbol{X}) > \hat{F}$. The *monotonicity* test determines whether the function F is strictly monotone in an entire sub-box $\boldsymbol{X}$. Denote the interval extension of the $i-$th component of the gradient evaluated in $\boldsymbol{X}$ by $\boldsymbol{g}_i(\boldsymbol{X})$. If $0 \notin \boldsymbol{g}_i(\boldsymbol{X})$ then $\boldsymbol{X}$ can be deleted.
The *concavity* test examines the concavity of F, using its Hessian matrix H. Let $\boldsymbol{H}_{i,i}(\boldsymbol{X})$ denotes the interval extension of the $i-$th diagonal entry of Hessian evaluated in $\boldsymbol{X}$. A box can be deleted if $\overline{\boldsymbol{H}}_{i,i}(\boldsymbol{X}) < 0$ for some i.

The *Interval Newton step* applies one step of the interval Newton method (Eq. (15)) to the non-linear system $\nabla F(X) = 0,\ X \in \boldsymbol{X}$. As a consequence we may validate that $\boldsymbol{X}$ contains no stationary points, in which case we discard $\boldsymbol{X}$, otherwise we may contract or subdivide $\boldsymbol{X}$.
A problem of global optimization algorithms based on IA is the so called *cluster effect*: as observed in [13], subboxes containing no solutions cannot be easily eliminated if there is a local minimum nearby. As a consequence of overestimation in range bounding, many small boxes are created by repeated splitting, whose processing may dominate the total work spent on global search. This phenomenon occurs when the order of the inclusion function is less than three [13], hence we shall look for sharper inclusion functions.

4.2.1 Taylor-Bernstein forms

An interesting extension of IA that reduces the overestimation is based on Taylor polynomials.

Definition 4 (Taylor Model) *Let $F : \boldsymbol{X} \subset \mathbb{R}^n \to \mathbb{R}$ be a function that is $(m+1)$ times continuously partially differentiable. Let X_0 be a point in $\boldsymbol{X}$ and $P_{m,F}$ the the m-th order Taylor polynomial of F around X_0. Let $\boldsymbol{I}_{m,F}$ be an interval such that*

$$F(X) \in P_{m,F}(X - X_0) + \boldsymbol{I}_{m,F} \quad \forall X \in \boldsymbol{X}. \qquad (16)$$

We call the pair $(P_{m,F}, \boldsymbol{I}_{m,F})$ an m-th order Taylor model *of F [18] .*

Taylor models of any computable function can be obtained recursively using the *Taylor Model Arithmetic* described in [18]. In order to bound the range of a function F over a domain $\boldsymbol{X}$, it is sufficient to compute an interval extension $\boldsymbol{P}_{m,F}(\boldsymbol{X})$ for the polynomial $P_{m,F}$, since from Definition 4 it follows that

$$\mathbf{F}^u(\boldsymbol{X}) \subseteq \boldsymbol{P}_{m,F}(\boldsymbol{X}) + \boldsymbol{I}_{m,F}.$$

The sharpness of the bounds depends on the method used to obtain the inclusion function for $P_{m,F}$. More precisely, if $\boldsymbol{P}^u_{m,F}(\boldsymbol{X})$ is the exact range of $P_{m,F}$, then $\boldsymbol{P}^u_{m,F}(\boldsymbol{X}) + \boldsymbol{I}_{m,F}$ is an $m+1$ order inclusion function for F over $\boldsymbol{X}$, where m is the degree of the Taylor polynomial [27].
A *Taylor-Bernstein form* is a Taylor model where the polynomial is expressed in the Bernstein basis rather than in the canonical power basis. The advantage is that the Taylor-Bernstein form allows to compute the exact range of the polynomial part. Hence, with $n \geq 2$, the cluster effect is avoided. A Bernstein polynomial has the form (in one dimension):

$$p(x) = \sum_{i=1}^{n} a_i \begin{pmatrix} n \\ i \end{pmatrix} x^i (1-x)^{n-i}. \qquad (17)$$

An important property of these polynomials is that $p(x)$ on $\boldsymbol{x}$ is a convex combination of a_i's, so that the coefficients of the Bernstein form provide lower and upper bounds to the range:

$$\boldsymbol{p}^u(\boldsymbol{x}) \subseteq [\min\{a_i\}, \max\{a_i\}].$$

If the polynomial is monotone over a domain $\boldsymbol{x}$ then the Bernstein form gives the exact range since the minimum and maximum occurs respectively at a_1 and a_n, $a_1 = p(\underline{\boldsymbol{x}})$ and $a_n = p(\overline{\boldsymbol{x}})$. This suggests that the exact range of a polynomial p on $\boldsymbol{x}$ can be obtained by transforming the polynomial into Bernstein form and then repeatedly subdividing it the until the bounds of all sub-boxes are exact. The subdivision can be easily done with De Casteljau algorithm, well known in Computer Graphics. Bernstein polynomials can be easily extended to the multivariate case, where analogous properties hold.
The knowledge of the exact range of $P_{m,F}$ helps to make the cut-off test more effective. Indeed, if $\boldsymbol{P}^u_{m,F}(\boldsymbol{X})$ is the exact range, then $\underline{\boldsymbol{P}}^u_{m,F}(\boldsymbol{X}) = \min\{P_{m,F}\}$ and the minimum of F over $\boldsymbol{X}$ is contained in $\underline{\boldsymbol{P}}^u_{m,F}(\boldsymbol{X}) + \boldsymbol{I}_{m,F}$. Then $\underline{\boldsymbol{P}}^u_{m,F}(\boldsymbol{X}) + \overline{\boldsymbol{I}}_{m,F}$ is an upper bound of the minimum of F over $\boldsymbol{X}$. The cut-off value $\hat{F}$ is the smallest upper bound for all the boxes in the list.
The advantages and limits of Taylor models are widely discussed in [23], where the author also points out that the Taylor-Bernstein method is well suited to low dimension problems.

5 Derivatives of the cost function

As seen above, the minimization algorithm makes use of the Jacobian and Hessian matrix of the cost function. Instead of reverting to subscript notation for computing the derivatives of the cost function, we perform the entire operation by using the elegant *matrix differential calculus* introduced by Magnus and Neudecker [17]. Space constraints do not allow us to report the results, which can be found in [2].

6 Experimental results

The algorithm was tested on synthetic data, which consisted of 50 points randomly scattered in a sphere of unit radius, centered at the origin. Views were generated by placing cameras at random positions, at a mean distance from the center of 2.5 units with a standard deviation of 0.25. The orientations of the cameras were chosen randomly with the constraint that the optical axis should point toward the center. The intrinsic parameters were given a known value: $\alpha_u = \alpha_v = 800, u_0 = v_0 = 256$. As customary it was assumed $\gamma = 0$. Image points were (roughly) contained in a 512×512 image. Fundamental matrices were computed using the linear 8-point algorithm with data normalization as described by Hartley in [6]. The weight w_{ij} has been defined as the residual of the estimation of F_{ij} [20]. We used Taylor models of degree four. The required accuracy was 10^{-10}; using this value we typically get a box width of 2.5 pixels.

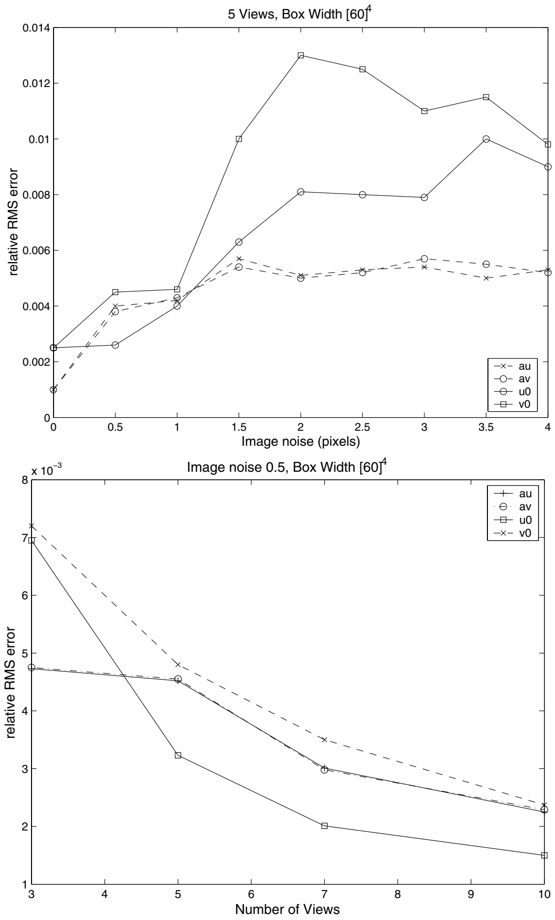

Figure 1. Relative RMS error on intrinsic parameters versus image noise standard deviation (left) and number of views (right).

In order to assess the accuracy of the method, Gaussian noise with variable standard deviation was added to image points and the number of views was varied as well (the minimum number of views required to achieve autocalibration is three, according to Eq. (8)). The algorithm was started with the box $[0, 60]^4$ centered on the true solution. Since the fundamental matrices are affected by image noise, the minimizer of the cost function does not coincide with the actual intrinsic parameters. The relative RMS error is reported in Figure 1. Each point is the average of 50 independent trials. Computation times were recorded for varying number of views, initial box width and number of unknowns. Table 1 reports computation times versus number of views. Table 2 reports computation times versus box width (5 views) starting from a reference box of $[300, 1700] \times [300, 1700] \times [156, 356] \times [156, 356]$. In the first column all the four parameters were considered unknown, in the second one only focal lengths were unknown, whereas the principal point was set at (256, 256). These figures refers to our implementation in MATLAB and C, on a Pentium III 900 MHz processor.

Number of views	3	5	7	10
Time [min]	1.2	6.6	8.0	17.1

Table 1. Computation times versus number of views. The initial box was $[0, 60]^4$.

Box Width	Time [min] (4 unknowns)	Time [min] (2 unknowns)
Ref.	23.2	9.1
-10%	16.6	8.8
-20%	15.3	7.2
+10%	28.3	11.9
+20%	32.0	14.8

Table 2. Computation times versus initial box width. In the rightmost column the principal point was known.

In order to compare our minimization with a standard gradient method, we used the quasi-Newton method implemented by the `fminunc` function in the MATLAB Optimization Toolbox. The algorithm was initialized by randomly choosing a point in the domain $[300, 1700] \times [300, 1700] \times [156, 356\ ]\times [156, 356]$. After performing 100 trials we recorded how many times the algorithm converged to the correct solution, which was assumed to be the one to which it converged when initialized with the true intrinsic parameters (within a 10% tollerance). The quasi-Newton method converged in the 86% of cases, with 5 views and 1.0 pixel noise. Average running time was 0.9 sec.

7 Conclusions and future work

Global optimization based on Interval Analysis has been applied to the autocalibration problem, obtaining a technique that is guaranteed to converge to the global solution with mathematical certainty and arbitrary accuracy. The results, albeit preliminary, shows that our implementation is correct and achieves the global minimum in a reasonable time. The choice of the initial box is not critical for the successful termination of the algorithm – provided that it contains the global minimizer – because it only influences the computation time.

The accuracy of the method is in agreement with the figures reported in [20, 28], as we use basically the same cost function. As customary, results can be refined by bundle ad-

justment, in order to obtain a maximum likelihood solution with respect to the underlying measures.
Future work will aim at reducing computation time by testing several variations to the present model.
We also plan to explore the use IA tools to automatically detect degenerate configurations, which are known to afflict autocalibration (see [26] for a summary).

Acknowledgments. The authors are grateful to R. P. Mendonça for introducing them to matrix differential calculus. A special thanks goes to Arnold Neumaier for many helpful discussion concerning the rigorous implementation of Taylor models. This work is supported by the NSF Engineering Research Center for Neuromorphic Systems Engineering (CNSE) at Caltech (EEC-9402726).

References

[1] A. Fusiello. Uncalibrated Euclidean reconstruction: A review. *Image and Vision Computing*, 18(6-7):555–563, 2000.

[2] A. Fusiello, A. Benedetti, M. Farenzena, and A. Busti. Globally convergent autocalibration using interval analysis. Technical Report RR 09/2003, Dipartimento di Informatica, Università di Verona, 2003.

[3] E. R. Hansen. *Global Optimization Using Interval Analysis*. Marcel Dekker, New York, 1992.

[4] R. Hartley, E. Hayman, L. de Agapito, and I. Reid. Camera calibration and the search for infinity. In *Proceedings of the IEEE International Conference on Computer Vision*, 1999.

[5] R. I. Hartley. Estimation of relative camera position for uncalibrated cameras. In *Proceedings of the European Conference on Computer Vision*, pages 579–587, 1992.

[6] R. I. Hartley. In defence of the 8-point algorithm. In *Proceedings of the IEEE International Conference on Computer Vision*, 1995.

[7] R. I. Hartley. Kruppa's equations derived from the fundamental matrix. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):133–135, 1997.

[8] A. Heyden and K. Åström. Euclidean reconstruction from constant intrinsic parameters. In *Proceedings of the International Conference on Pattern Recognition*, pages 339–343, 1996.

[9] A. Heyden and K. Åström. Euclidean reconstruction from image sequences with varying and unknown focal length and principal point. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 438–443, 1997.

[10] A. Heyden and K. Åström. Minimal conditions on intrinsic parameters for Euclidean reconstruction. In *Proceedings of the Asian Conference on Computer Vision*, 1998.

[11] T. Huang and O. Faugeras. Some properties of the E matrix in two-view motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(12):1310–1312, 1989.

[12] T. S. Huang and A. N. Netravali. Motion and structure from feature correspondences: A review. *Proceedings of IEEE*, 82(2):252–267, 1994.

[13] B. Kearfott and Du. The cluster problem in multivariate global optimization. *Journal of Global Optimization*, 5:253–365, 1994.

[14] R. B. Kearfott. *Rigorous Global Search: Continuous Problems*. Kluwer, 1996.

[15] M. I. Lourakis and R. Deriche. Camera self-calibration using the singular value decomposition of the fundamental matrix. In *Proc. of the 4th Asian Conference on Computer Vision*, volume I, pages 403–408, 2000.

[16] Q.-T. Luong and O. Faugeras. Self-calibration of a moving camera from point correspondences and fundamental matrices. *International Journal of Computer Vision*, 22(3):261–289, 1997.

[17] J. R. Magnus and H. Neudecker. *"Matrix Differential Calculus with Applications in Statistics and Econometrics"*. John Wiley & Sons, revised edition, 1999.

[18] K. Makino and M. Berz. Taylor models and other validated functional inclusion methods. *International Journal of Pure and Applied Mathematics*, 4(4):379–456, 2003.

[19] S. J. Maybank and O. Faugeras. A theory of self-calibration of a moving camera. *International Journal of Computer Vision*, 8(2):123–151, 1992.

[20] P. Mendonça and R. Cipolla. A simple techinique for self-calibration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages I:500–505, 1999.

[21] R. E. Moore. *Interval Analysis*. Prentice-Hall, 1966.

[22] A. Neumaier. *Introduction to Numerical Analysis*. Cambridge Univ. Press, 2001.

[23] A. Neumaier. Taylor forms - use and limits. *Reliable Computing*, 9:43 – 79, 2002.

[24] M. Pollefeys, R. Koch, and L. Van Gool. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 90–95, 1998.

[25] M. Pollefeys and L. Van Gool. A stratified approach to metric self-calibration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 407–412, 1997.

[26] J. Ponce. On computing metric upgrades of projective reconstructions under the rectangular pixel assumption. In *Proc. of the SMILE 2000 Workshop on 3D Structure from Multiple Images of Large-Scale Environments,*, number 2018 in LNCS, pages 52–67. Springer-Verlag, 2000.

[27] P.S.V.Nataraj and K.Kotecha. An algorithm for global optimization using the taylor-bernstein form as an inclusion function. *International Journal of Global Optimization*, 24:417–436, 2002.

[28] G. Roth and A. Whitehead. Some improvements on two autcalibration algorithms based on the fundamental matrix. In *Proceedings of the International Conference on Pattern Recognition*, volume 2, pages 312–315, 2002.

[29] P. Sturm. On focal length calibration fro two views. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume II, pages 145–150, 2001.

[30] B. Triggs. Autocalibration and the absolute quadric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 609–614, 1997.