Autocalibration from Planar Scenes

Bill Triggs

INRIA Rhône-Alpes, 655 avenue de l'Europe, 38330 Montbonnot St. Martin, France.

Bill.Triggs@inrialpes.fr ◇ *http://www.inrialpes.fr/movi/people/Triggs*

Abstract

This paper describes a theory and a practical algorithm for the autocalibration of a moving projective camera, from $m \geq 5$ views of a *planar* scene. The unknown camera calibration, and (up to scale) the unknown scene geometry and camera motion are recovered from the hypothesis that the camera's internal parameters remain constant during the motion. This work extends the various existing methods for non-planar autocalibration to a practically common situation in which it is not possible to bootstrap the calibration from an intermediate projective reconstruction. It also extends Hartley's method for the internal calibration of a rotating camera, to allow camera translation and to provide 3D as well as calibration information. The basic constraint is that the projections of orthogonal direction vectors (points at infinity) in the plane must be orthogonal in the calibrated camera frame of each image. Abstractly, since the two circular points of the 3D plane (representing its Euclidean structure) lie on the 3D absolute conic, their projections into each image must lie on the absolute conic's image (representing the camera calibration). The resulting numerical algorithm optimizes this constraint over all circular points and projective calibration parameters, using the inter-image homographies as a projective scene representation.

Keywords: Autocalibration, Euclidean structure, Absolute Conic & Quadric, Planar Scenes.

1 Introduction

This paper describes a method of autocalibrating a moving projective camera with general, unknown motion and unknown intrinsic parameters, from $m \geq 5$ views of a *planar* scene. Autocalibration is the recovery of metric information — for example the internal and external calibration parameters of a moving projective camera — from non-metric information and (metric) self-consistency constraints — for example the knowledge that the camera's internal parameters are constant during the motion, and the inter-image consistency constraints that this entails. Since the seminal work of Maybank & Faugeras [14, 3], a number of different approaches to autocalibration have been developed [5, 6, 1, 27, 26, 2, 13, 9, 16, 15, 22, 10]. For the 'classical' problem of

4/3/98. This is an extension of a preliminary version of my ECCV'98 paper [23]. It goes into a little more detail on various topics, notably Kruppa instability, and contains appendices describing the SVD-based relative orientation method for calibrated cameras used for the initialization search, and an unused (but potentially useful) factorization method for homographies between $m \geq 2$ images. The work was supported by Esprit LTR project CUMULI and INRIA Rhône-Alpes. I would like to thank P. Sturm for discussions, the anonymous reviewers for comments, G. Csurka and A. Ruf for the test images and calibration data, and C. Gramkow for pointing out some missing constants in eqns. (11) and (12).

a single perspective camera with constant but unknown internal parameters moving with a general but unknown motion in a 3D scene, the original Kruppa equation based approach [14] seems to be being displaced by approaches based on the 'rectification' of an intermediate projective reconstruction [5, 9, 15, 22, 10]. More specialized methods exist for particular types of motion and simplified calibration models [6, 24, 1, 16]. Stereo heads can also be autocalibrated [27, 11]. Solutions are still — in theory — possible if some of the intrinsic parameters are allowed to vary [9, 15]. Hartley [6] has given a particularly simple internal calibration method for the case of a single camera whose translation is known to be negligible compared to the distances of some identifiable (real or synthetic) points in the scene, and Faugeras [2] has elaborated a 'stratification' paradigm for autocalibration based on this. The numerical conditioning of classical autocalibration is historically delicate, although recent algorithms have improved the situation significantly [9, 15, 22]. The main problem is that classical autocalibration has some restrictive intrinsic degeneracies — classes of motion for which no algorithm can recover a full unique solution. Sturm [18, 19] has given a catalogue of these. In particular, at least 3 views, some translation and some rotation about at least two non-aligned axes are required.

Planar Autocalibration: All of the existing approaches to classical autocalibration rely on information equivalent to a 3D projective reconstruction of the scene. In the Kruppa approach this is the fundamental matrices and epipoles, while for most other methods it is an explicit 3D reconstruction. For some applications (especially in man-made environments) this is potentially a problem, because planar or near-planar scenes sometimes occur for which stable 3D projective reconstructions (or fundamental matrices, *etc.*) can not be calculated. This well-known failing of projective reconstruction is something of an embarrassment: the *calibrated* reconstruction of planar scenes is not difficult, so it is exactly in this case when autocalibration fails that it would be most useful. The current paper aims to rectify this by providing autocalibration methods that work in the planar case, by 'rectifying' the inter-image homographies induced by the plane. In the longer term, we would like to find ways around the ill-conditioning of projective reconstruction for near-planar scenes, and also to develop 'structure-free' internal calibration methods similar to Hartley's zero-translation one [6], but which work for non-zero translations. The hope is that planar methods may offer one way to attack these problems.

Planar autocalibration has other potential advantages. Planes are very common in man-made environments, and often easily identifiable and rather accurately planar. They are simple to process and allow very reliable and precise feature-based or intensity-based matching, by fitting the homographies between image pairs. They are also naturally well adapted to the calibration of lens distortion as some of the subtleties of 3D geometry are avoided[1].

The main *dis*advantage of planar autocalibration (besides the need for a nice, flat, textured plane) seems to be the number of images required. Generically, $m \geq \lceil \frac{n+4}{2} \rceil$ images are needed for an internal camera model with n free parameters, *e.g.* $m \geq 5$ for the classical 5 parameter projective model (focal length, aspect ratio, skew, principal point), or $m \geq 3$ if only focal length is estimated. However for good accuracy and reliability, at least 8–10 images are recommended in practice. Also, almost any attempt at algebraic elimination across so many images rapidly leads to a combinatorial explosion. Hence, the approach is resolutely numerical, and it seems impracticable

[1] We will ignore lens distortion throughout this paper. If necessary it can be corrected by a nominal model, or — at least in theory — estimated up to an overall 3×3 projectivity by a bundled adjustment over all the inter-image homographies. (The pixel-pixel mapping induced by geometric homography $\boldsymbol{H}_i$ is $\boldsymbol{D}\boldsymbol{H}_i\boldsymbol{D}^{-1}$ where $\boldsymbol{D}$ is the distortion model).

to initialize the optimization from a minimal algebraic solution. Although for the most part the numerical domain of convergence seems to be sufficient to allow moderately reliable convergence from a fixed default initialization, and we have also developed a numerical initialization search which may be useful in some cases, occasional convergence to false minima remains a problem.

Organization: Section 2 gives a direction-vector based formulation of the theory of autocalibration, and discusses how both non-planar and planar autocalibration can be approached within this framework. Section 3 describes the statistically-motivated cost function we optimize. Section 4 discusses the numerical algorithm, and the method used to initialize it. Section 5 gives experimental results on synthetic and real images, and section 6 concludes the paper.

Notation will be introduced as required. Briefly we use bold upright $\mathbf{x}$ for homogeneous 3D (4 component) vectors and matrices; bold italic $\boldsymbol{x}$ for 3 component ones (homogeneous image, inhomogeneous 3D, 3-component parts of homogeneous 4-component objects); $\boldsymbol{P}$ for image projections and $\boldsymbol{H}$ for inter-image homographies; $\boldsymbol{K}$, $\boldsymbol{C} = \boldsymbol{K}^{-1}$ for upper triangular camera calibration and inverse calibration matrices; Ω and Ω^* for the absolute (hyperplane) quadric and (direction) conic; and $\boldsymbol{\omega} = \boldsymbol{K}\boldsymbol{K}^\top = \boldsymbol{P}\,\Omega\,\boldsymbol{P}^\top$ and $\boldsymbol{\omega}^{-1} = \boldsymbol{C}^\top\boldsymbol{C}$ for their images. $[\,\cdot\,]_\times$ denotes the matrix generating the cross product: $[\,\boldsymbol{x}\,]_\times\boldsymbol{y} = \boldsymbol{x} \wedge \boldsymbol{y}$.

2 Euclidean Structure and Autocalibration

To recover the metric information implicit in projective images, we need a projective encoding of Euclidean structure. The key to Euclidean structure is the dot product between direction vectors ("points at infinity"), or dually the dot product between (normals to) hyperplanes. The former leads to the stratified "hyperplane at infinity + absolute (direction) conic" formulation (affine + metric structure) [17], the latter to the "absolute (hyperplane) quadric" one [22]. These are just dual ways of saying the same thing. The hyperplane formalism is preferable for 'pure' autocalibration where there is no *a priori* decomposition into affine and metric strata, while the point one is simpler if such a stratification is given.

Generalities: Consider k-dimensional Euclidean space. We will need the cases $k = 2$ (the planar scene and its 2D images) and $k = 3$ (ordinary 3D space). Introducing homogeneous Euclidean coordinates, points, displacement vectors and hyperplanes are encoded respectively as homogeneous $k+1$ component column vectors $\mathbf{x} = (\boldsymbol{x}, 1)^\top$, $\mathbf{t} = (\boldsymbol{t}, 0)^\top$ and row vectors $\mathbf{p} = (\boldsymbol{n}, d)$. Here $\boldsymbol{x}$, $\boldsymbol{t}$ and $\boldsymbol{n}$ are the usual k-D coordinate vectors of the point, the displacement, and the hyperplane normal, and d is the hyperplane offset. Points and displacements on the plane satisfy respectively $\mathbf{p}\cdot\mathbf{x} = \boldsymbol{n}\cdot\boldsymbol{x} + d = 0$ and $\mathbf{p}\cdot\mathbf{t} = \boldsymbol{n}\cdot\boldsymbol{t} = 0$. Displacement directions can be appended to the point space, as a **hyperplane at infinity** $\mathbf{p}_\infty$ of **points at infinity** or **vanishing points**. Projectively, $\mathbf{p}_\infty$ behaves much like any other hyperplane. In Euclidean coordinates, $\mathbf{p}_\infty = (\boldsymbol{0}, 1)$ so that $\mathbf{p}_\infty\cdot\mathbf{t} = 0$ for any displacement $\mathbf{t} = (\boldsymbol{t}, 0)$. **Projective transformations** mix finite and infinite points. Under a projective transformation encoded by an arbitrary nonsingular $(k+1)\times(k+1)$ matrix $\mathbf{T}$, points and directions (column vectors) transform **contravariantly**, *i.e.* by $\mathbf{T}$ acting on the left: $\mathbf{x} \to \mathbf{T}\,\mathbf{x}$, $\mathbf{v} \to \mathbf{T}\,\mathbf{v}$. To preserve the point-on-plane relation $\mathbf{p}\cdot\mathbf{x} = \boldsymbol{n}\cdot\boldsymbol{x} + d = 0$, hyperplanes (row vectors) transform **covariantly**, *i.e.* by $\mathbf{T}^{-1}$ acting on the right: $\mathbf{p} \to \mathbf{p}\,\mathbf{T}^{-1}$.

Absolute Quadric & Conic: The usual Euclidean dot product between hyperplane normals is $\boldsymbol{n}_1 \cdot \boldsymbol{n}_2 = \mathbf{p}_1 \, \boldsymbol{\Omega} \, \mathbf{p}_2^\top$ where the symmetric, rank k, positive semidefinite matrix

$$\boldsymbol{\Omega} = \begin{pmatrix} \boldsymbol{I}_{k\times k} & \boldsymbol{0} \\ \boldsymbol{0} & 0 \end{pmatrix}$$

is called the **absolute (hyperplane) quadric**[2]. $\boldsymbol{\Omega}$ encodes the Euclidean structure in projective coordinates. Under projective transformations it transforms contravariantly (*i.e.* like a point) in each of its two indices so that the dot product between plane normals is invariant: $\boldsymbol{\Omega} \rightarrow \mathbf{T} \, \boldsymbol{\Omega} \, \mathbf{T}^\top$ and $\mathbf{p}_i \rightarrow \mathbf{p}_i \, \mathbf{T}^{-1}$, so $\mathbf{p}_1 \, \boldsymbol{\Omega} \, \mathbf{p}_2^\top = \boldsymbol{n}_1 \cdot \boldsymbol{n}_2$ is constant. $\boldsymbol{\Omega}$ is invariant under Euclidean transformations, but in a general projective frame it loses its diagonal form and becomes an arbitrary symmetric positive semidefinite rank k matrix. In any frame, the Euclidean angle between two hyperplanes is $\cos\theta = (\mathbf{p} \, \boldsymbol{\Omega} \, \mathbf{p}'^\top)/\sqrt{(\mathbf{p} \, \boldsymbol{\Omega} \, \mathbf{p}^\top)(\mathbf{p}' \, \boldsymbol{\Omega} \, \mathbf{p}'^\top)}$, and the plane at infinity is $\boldsymbol{\Omega}$'s unique null vector: $\mathbf{p}_\infty \, \boldsymbol{\Omega} = \mathbf{0}$. When restricted to coordinates on $\mathbf{p}_\infty$, $\boldsymbol{\Omega}$ becomes nonsingular and can be dualized (inverted) to give the $k \times k$ symmetric positive definite **absolute (direction) conic** $\boldsymbol{\Omega}^*$. This measures dot products between displacement vectors, just as $\boldsymbol{\Omega}$ measures them between hyperplane normals. $\boldsymbol{\Omega}^*$ is defined *only* on direction vectors, not on finite points, and unlike $\boldsymbol{\Omega}$ it has no unique canonical form in terms of the *un*restricted coordinates. (Anything of the form $\left(\begin{smallmatrix} \boldsymbol{I} & \boldsymbol{x} \\ \boldsymbol{x}^\top & y \end{smallmatrix}\right)$ can be used, for arbitrary $\boldsymbol{x}, y$).

Direction bases: In Euclidean coordinates, $\boldsymbol{\Omega}$ can be decomposed as a sum of outer products of any orthonormal (in terms of $\boldsymbol{\Omega}^*$) basis of displacement vectors: $\boldsymbol{\Omega} = \sum_{i=1}^{k} \boldsymbol{x}_i \boldsymbol{x}_i^\top$ where $\boldsymbol{x}_i \, \boldsymbol{\Omega}^* \, \boldsymbol{x}_j = \delta_{ij}$. For example in 2D $\boldsymbol{\Omega} = \left(\begin{smallmatrix} \boldsymbol{I}_{2\times 2} & \boldsymbol{0} \\ \boldsymbol{0} & 0 \end{smallmatrix}\right) = \hat{\boldsymbol{x}}\hat{\boldsymbol{x}}^\top + \hat{\boldsymbol{y}}\hat{\boldsymbol{y}}^\top$ where $\hat{\boldsymbol{x}} = (1,0,0), \hat{\boldsymbol{y}} = (0,1,0)$, are the usual unit direction vectors. Gathering the basis vectors into the columns of a $(k+1) \times k$ orthonormal rank k matrix $\mathbf{U}$ we have $\boldsymbol{\Omega} = \mathbf{U} \, \mathbf{U}^\top$, $\mathbf{p}_\infty \, \mathbf{U} = \mathbf{0}$ and $\mathbf{U}^\top \boldsymbol{\Omega}^* \mathbf{U} = \boldsymbol{I}_{k\times k}$. The columns of $\mathbf{U}$ span $\mathbf{p}_\infty$. All of these relations remain valid in an arbitrary projective frame $\mathbf{T}$ and with an arbitrary choice of representative for $\boldsymbol{\Omega}^*$, except that $\mathbf{U} \rightarrow \mathbf{T} \, \mathbf{U}$ ceases to be orthonormal.

$\mathbf{U}$ is defined only up to an arbitrary $k \times k$ orthogonal mixing of its columns (redefinition of the direction basis) $\mathbf{U} \rightarrow \mathbf{U} \, \boldsymbol{R}_{k\times k}$. Even in a projective frame where $\mathbf{U}$ itself is not orthonormal, this mixing freedom remains orthogonal. In a Euclidean frame $\mathbf{U} = \left(\begin{smallmatrix} \boldsymbol{V} \\ 0 \end{smallmatrix}\right)$ for some $k \times k$ rotation matrix $\boldsymbol{V}$, so the effect of a Euclidean space transformation is $\mathbf{U} \rightarrow \left(\begin{smallmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{smallmatrix}\right) \mathbf{U} = \mathbf{U} \, \boldsymbol{R}'$ where $\boldsymbol{R}' = \boldsymbol{V}^\top \boldsymbol{R} \boldsymbol{V}$ is the conjugate rotation: Euclidean transformations of direction bases (*i.e.* on the left) are equivalent to orthogonal re-mixings of them (*i.e.* on the right). This remains true in an arbitrary projective frame, even though $\mathbf{U}$ and the transformation no longer *look* Euclidean. This mixing freedom can be used to choose a direction basis in which $\mathbf{U}$ is orthonormal up to a diagonal rescaling: simply take the SVD $\mathbf{U}' \boldsymbol{D} \, \boldsymbol{V}^\top$ of $\mathbf{U}$ and discard the mixing rotation $\boldsymbol{V}^\top$. Equivalently, the eigenvectors and square roots of eigenvalues of $\boldsymbol{\Omega}$ can be used. Such orthogonal parametrizations of $\mathbf{U}$ make good numerical sense, and we will use them below.

Circular points: Given any two orthonormal direction vectors $\mathbf{x}, \mathbf{y}$, the complex conjugate vectors $\mathbf{x}_\pm \equiv \frac{1}{\sqrt{2}}(\mathbf{x} \pm i\mathbf{y})$ satisfy $\mathbf{x}_\pm \, \boldsymbol{\Omega}^* \, \mathbf{x}_\pm^\top = 0$. Abstractly, these complex directions "lie on the absolute conic", and it is easy to check that any complex projective point which does so can be decomposed into two orthogonal direction vectors, its real and imaginary parts. In the 2D case there is only one such conjugate pair up to complex phase, and these **circular points** characterize the Euclidean structure of the plane. However for numerical purposes, it is usually easier to avoid

[2] Abstractly, $\boldsymbol{\Omega}$ can be viewed as a cone (degenerate quadric hypersurface) with no real points in complex projective hyperplane space. But it is usually simpler just to think of it concretely as a symmetric matrix with certain properties.

complex numbers by using the real and imaginary parts $\mathbf{x}$ and $\mathbf{y}$ rather than $\mathbf{x}_\pm$. The phase freedom in $\mathbf{x}_\pm$ corresponds to the 2×2 orthogonal mixing freedom of $\mathbf{x}$ and $\mathbf{y}$.

Theoretically, the above parametrizations of Euclidean structure are equivalent. Which is practically best depends on the problem. $\mathbf{\Omega}$ is easy to use, except that constrained optimization is required to handle the rank k constraint $\det \mathbf{\Omega} = 0$. Direction bases $\mathbf{U}$ eliminate this constraint at the cost of numerical code to handle their $k \times k$ orthogonal gauge freedom. The absolute conic $\mathbf{\Omega}^*$ has neither constraint nor gauge freedom, but has significantly more complicated image projection properties and can only be defined once the plane at infinity $\mathbf{p}_\infty$ is known and a projective coordinate system on it has been chosen (*e.g.* by induction from one of the images). It is also possible to parametrize Euclidean structure by non-orthogonal Choleski-like decompositions $\mathbf{\Omega} = \mathbf{L}\,\mathbf{L}^\top$ (*i.e.* the L part of the LQ decomposition of $\mathbf{U}$), but this introduces singularities at maximally non-Euclidean frames unless pivoting is also used.

Image Projections: Since the columns of a 3D direction basis matrix $\mathbf{U}$ are *bona fide* 3D direction vectors, its image projection is simply $\boldsymbol{P}\,\mathbf{U}$, where $\boldsymbol{P}$ is the usual 3×4 point projection matrix. Hence, the projection of $\mathbf{\Omega} = \mathbf{U}\,\mathbf{U}^\top$ is the 3×3 symmetric positive definite contravariant image matrix $\boldsymbol{\omega} = \boldsymbol{P}\,\mathbf{\Omega}\,\boldsymbol{P}^\top$. Abstractly, this is the image line quadric dual to the image of the absolute conic. Concretely, given any two image lines $\boldsymbol{l}_1, \boldsymbol{l}_2$, $\boldsymbol{\omega}$ encodes the 3D dot product between their 3D visual planes $\mathbf{p}_i = \boldsymbol{l}_i \boldsymbol{P}$: $\mathbf{p}_1\,\mathbf{\Omega}\,\mathbf{p}_2^\top = \boldsymbol{l}_1 \boldsymbol{P}\,\mathbf{\Omega}\,\boldsymbol{P}^\top \boldsymbol{l}_2^\top = \boldsymbol{l}_1\,\boldsymbol{\omega}\,\boldsymbol{l}_2^\top$. With the traditional Euclidean decomposition $\boldsymbol{K}\boldsymbol{R}\,(\boldsymbol{I} \mid -\boldsymbol{t})$ of $\boldsymbol{P}$ into an upper triangular **internal calibration matrix** $\boldsymbol{K}$, a 3×3 **camera orientation** (rotation) $\boldsymbol{R}$ and an **optical centre** $\boldsymbol{t}$, $\boldsymbol{\omega}$ becomes simply $\boldsymbol{K}\,\boldsymbol{K}^\top$. Since $\mathbf{\Omega}$ is invariant under Euclidean motions, $\boldsymbol{\omega}$ is invariant under camera displacements so long as $\boldsymbol{K}$ remains constant. $\boldsymbol{K}$ can be recovered from $\boldsymbol{\omega}$ by Choleski decomposition, and similarly the Euclidean scene structure (in the form of a 'rectifying' projective transformation) can be recovered from $\mathbf{\Omega}$. The upper triangular **inverse calibration matrix** $\boldsymbol{C} = \boldsymbol{K}^{-1}$ converts homogeneous pixel coordinates to optical ray directions in the Euclidean camera frame. $\boldsymbol{\omega}^{-1} = \boldsymbol{C}^\top \boldsymbol{C}$ is the image of the absolute conic.

Autocalibration: Given several images taken with projection matrices $\boldsymbol{P}_i = \boldsymbol{K}_i \boldsymbol{R}_i (\boldsymbol{I} \mid -\boldsymbol{t}_i)$, and (in the same Euclidean frame) a orthogonal direction basis $\mathbf{U} = \left(\begin{smallmatrix} \boldsymbol{V} \\ \boldsymbol{0} \end{smallmatrix}\right)$, we find that

$$\boldsymbol{C}_i\,\boldsymbol{P}_i\,\mathbf{U} \;=\; \boldsymbol{R}'_i \tag{1}$$

where $\boldsymbol{C}_i = \boldsymbol{K}_i^{-1}$ and $\boldsymbol{R}'_i = \boldsymbol{R}_i\,\boldsymbol{V}$ is a rotation matrix depending on the camera pose. This is perhaps the most basic form of the autocalibration constraint. It says that the calibrated images (*i.e.* 3D directions in the camera frame) of an orthogonal direction basis must remain orthogonal. It remains true in arbitrary projective 3D and image frames, as the projective deformations of $\mathbf{U}$ *vs.* $\boldsymbol{P}_i$ and $\boldsymbol{P}_i$ *vs.* $\boldsymbol{C}_i$ cancel each other out. However, it is not usually possible to choose the scale factors of projectively reconstructed projections *a priori*, in a manner consistent with those of their unknown Euclidean parents. So in practice this constraint can only be applied up to an unknown scale factor for each image: $\boldsymbol{C}_i\,\boldsymbol{P}_i\,\mathbf{U} \;\sim\; \boldsymbol{R}'_i$. As always, the direction basis $\mathbf{U}$ is defined only up to an arbitrary 3×3 orthogonal mixing $\mathbf{U} \to \mathbf{U}\,\boldsymbol{R}$.

2.1 Autocalibration for Non-Planar Scenes

The simplest approaches to autocalibration for non-planar scenes are based on the consistency equation (1), an intermediate projective reconstruction $\boldsymbol{P}_i$, and some sort of knowledge about the $\boldsymbol{C}_i$

(*e.g.* classically that they are all the same: $\boldsymbol{C}_i = \boldsymbol{C}$ for some unknown $\boldsymbol{C}$). Nonlinear optimization or algebraic elimination are used to estimate the Euclidean structure $\boldsymbol{\Omega}$ or $\mathbf{U}$, and the free parameters of the $\boldsymbol{C}_i$. Multiplying (1) either on the left or on the right by its transpose to eliminate the unknown rotation, and optionally moving the $\boldsymbol{C}$'s to the right hand side, gives several equivalent symmetric 3×3 constraints linking $\boldsymbol{\Omega}$ or $\mathbf{U}$ to $\boldsymbol{\omega}_i$, $\boldsymbol{K}_i$ or $\boldsymbol{C}_i$

$$\mathbf{U}^\top \boldsymbol{P}_i^\top \boldsymbol{\omega}_i^{-1} \boldsymbol{P}_i \mathbf{U} \sim \boldsymbol{I}_{3\times 3} \tag{2}$$

$$\boldsymbol{C}_i \boldsymbol{P}_i \boldsymbol{\Omega} \boldsymbol{P}_i^\top \boldsymbol{C}_i^\top \sim \boldsymbol{I}_{3\times 3} \tag{3}$$

$$\boldsymbol{P}_i \boldsymbol{\Omega} \boldsymbol{P}_i^\top \sim \boldsymbol{\omega}_i = \boldsymbol{K}_i \boldsymbol{K}_i^\top \tag{4}$$

In each case there are 5 independent constraints per image on the 8 non-Euclidean d.o.f. of the 3D projective structure[3] and the 5 (or fewer) d.o.f. of the internal calibration. For example, three images in general position suffice for classical constant-$\boldsymbol{C}$ autocalibration. In each case, the unknown scale factors can be eliminated by treating the symmetric 3×3 left and right hand side matrices as $3 \cdot 4/2 = 6$ component vectors, and either (*i*) projecting (say) the left hand sides orthogonally to the right hand ones (hence deleting the proportional components and focusing on the constraint-violating non-proportional ones), or (*ii*) cross-multiplying in the usual way:

$$\begin{array}{c} \mathbf{u}_i \cdot \mathbf{v}_i = \mathbf{u}_i \cdot \mathbf{w}_i = \mathbf{v}_i \cdot \mathbf{w}_i = 0 \\ \|\mathbf{u}_i\|^2 = \|\mathbf{v}_i\|^2 = \|\mathbf{w}_i\|^2 \end{array} \quad \text{where} \quad (\mathbf{u}_i, \mathbf{v}_i, \mathbf{w}_i) \equiv \boldsymbol{C}_i \boldsymbol{P}_i \mathbf{U} \tag{5}$$

$$\begin{array}{c} (\boldsymbol{C}_i \boldsymbol{P}_i \boldsymbol{\Omega} \boldsymbol{P}_i^\top \boldsymbol{C}_i^\top)^{AA} = (\boldsymbol{C}_i \boldsymbol{P}_i \boldsymbol{\Omega} \boldsymbol{P}_i^\top \boldsymbol{C}_i^\top)^{BB} \\ (\boldsymbol{C}_i \boldsymbol{P}_i \boldsymbol{\Omega} \boldsymbol{P}_i^\top \boldsymbol{C}_i^\top)^{AB} = 0 \end{array} \quad \text{where} \quad A < B = 1 \ldots 3 \tag{6}$$

$$(\boldsymbol{P}_i \boldsymbol{\Omega} \boldsymbol{P}_i^\top)^{AB} \; (\boldsymbol{\omega})^{CD} = (\boldsymbol{\omega})^{AB} \; (\boldsymbol{P}_i \boldsymbol{\Omega} \boldsymbol{P}_i^\top)^{CD} \quad \text{where} \quad A \leq B, C \leq D = 1 \ldots 3 \tag{7}$$

Several recent autocalibration methods for 3D scenes (*e.g.* [22, 9]) are based implicitly on these constraints, parametrized by $\boldsymbol{K}$ or $\boldsymbol{\omega}$ and by something equivalent[4] to $\boldsymbol{\Omega}$ or $\mathbf{U}$. All of these methods seem to work well provided the intrinsic degeneracies of the autocalibration problem [18] are avoided.

In contrast, methods based on the Kruppa equations [14, 3, 26] can not be recommended for general use, because they add a serious additional singularity to the already-restrictive ones intrinsic to the problem. If any 3D point projects to the same pixel and is viewed from the same distance in each image, a 'zoom' parameter can not be recovered from the Kruppa equations. In particular, for a camera moving around an origin and fixating it at the image centre, the focal length can not be recovered[5]. Sturm [19] gives a geometric argument for this, but it is also easy to see algebraically. Let $\boldsymbol{x}$ be the fixed image point, $\boldsymbol{F}$ the fundamental matrix between images 1 and 2, $\boldsymbol{e}$

[3]These can be counted as follows: 15 for a 3D projective transformation modulo 7 for a scaled Euclidean one; or 12 for a 4×3 $\mathbf{U}$ matrix modulo 1 scale and 3 d.o.f. for a 3×3 orthogonal mixing; or $4 \cdot 5/2 = 10$ d.o.f. for a 4×4 symmetric quadric matrix $\boldsymbol{\Omega}$ modulo 1 scale and 1 d.o.f. for the rank 3 constraint $\det \boldsymbol{\Omega} = 0$; or 3 d.o.f. for $\mathbf{p}_\infty$ and 5 for the $3 \cdot 4/2 = 6$ components of $\boldsymbol{\Omega}^*$ modulo 1 scale.

[4]If the first camera projection is taken to be $(\boldsymbol{I} \,|\boldsymbol{0})$ [5, 9], $\mathbf{U}$ can be chosen to have the form $\begin{pmatrix} \boldsymbol{I} \\ -\boldsymbol{p}^\top \end{pmatrix} \boldsymbol{K}$ where $\mathbf{p}_\infty \sim (\boldsymbol{p}^\top, 1)$, whence $\boldsymbol{\Omega} \sim \begin{pmatrix} \boldsymbol{\omega} & -\boldsymbol{\omega}\boldsymbol{p} \\ -\boldsymbol{p}^\top\boldsymbol{\omega} & \boldsymbol{p}^\top\boldsymbol{\omega}\boldsymbol{p} \end{pmatrix}$ and $\begin{pmatrix} \boldsymbol{C} & \boldsymbol{0} \\ \boldsymbol{p}^\top & 1 \end{pmatrix}$ is a Euclideanizing projectivity.

[5]For most other autocalibration methods, this case is ambiguous only if the fixed point is at infinity (rotation about a fixed axis + arbitrary translation).

the epipole of image 2 in image 1, and $\boldsymbol{\omega}$ the constant dual absolute image quadric. Choosing appropriate scale factors for $\boldsymbol{e}$ and $\boldsymbol{F}$, the Kruppa constraint can be written as $\boldsymbol{F}\,\boldsymbol{\omega}\,\boldsymbol{F}^\top = [\,\boldsymbol{e}\,]_\times \boldsymbol{\omega}\,[\,\boldsymbol{e}\,]_\times^\top$. Since $\boldsymbol{x}$ is fixed, $\boldsymbol{x}^\top \boldsymbol{F}\,\boldsymbol{x} = 0$ and by the projective depth recovery relations [20] $\boldsymbol{F}\,\boldsymbol{x} = \lambda\,[\,\boldsymbol{e}\,]_\times \boldsymbol{x}$ where λ is the relative projective depth (projective scale factor) of $\boldsymbol{x}$ in the two images. Hence $\boldsymbol{F}(\boldsymbol{\omega} + \mu\,\boldsymbol{x}\,\boldsymbol{x}^\top)\boldsymbol{F}^\top = [\,\boldsymbol{e}\,]_\times(\boldsymbol{\omega} + \mu\,\lambda\,\boldsymbol{x}\,\boldsymbol{x}^\top)[\,\boldsymbol{e}\,]_\times^\top$. With these normalizations of $\boldsymbol{e}$ and $\boldsymbol{F}$, $\lambda = 1$ iff the *Euclidean* depth of $\boldsymbol{x}$ is the same in each image. If this is the case for all of the images we see that if $\boldsymbol{\omega}$ is a solution of the Kruppa equations, so is $\boldsymbol{\omega} + \mu\,\boldsymbol{x}\,\boldsymbol{x}^\top$ for any μ. This means that the calibration can only be recovered up to a zoom centred on $\boldsymbol{x}$. Numerical experience suggests that Kruppa-based autocalibration remains ill-conditioned even quite far from this singularity. This is hardly surprising given that in any case the distinction between zooms and closes depends on fairly subtle 2^{nd}-order perspective effects, so that the recovery of focal lengths is never simple. (Conversely, the effects of an inaccurate zoom-close calibration on image measurements or local object-centred 3D ones are relatively minor).

2.2 Autocalibration from Planar Scenes

Now consider autocalibration from *planar* scenes. Everything above remains valid, except that no intermediate 3D projective reconstruction is available from which to bootstrap the process. However we will see that by using the inter-image homographies, autocalibration is still possible.

The Euclidean structure of the scene plane is given by any one of (*i*) a 3×3 rank 2 absolute line quadric $\boldsymbol{Q}$; (*ii*) a 3 component line at infinity $\boldsymbol{l}_\infty$ and its associated 2×2 absolute (direction) conic matrix; (*iii*) a 3×2 direction basis matrix $\boldsymbol{U} = (\boldsymbol{x}\;\boldsymbol{y})$; (*iv*) two complex conjugate circular points $\boldsymbol{x}_\pm = \frac{1}{\sqrt{2}}(\boldsymbol{x} \pm i\boldsymbol{y})$ which are also the two roots of the absolute conic on $\boldsymbol{l}_\infty$ and the factors of the absolute line quadric $\boldsymbol{Q} = \boldsymbol{x}\,\boldsymbol{x}^\top + \boldsymbol{y}\,\boldsymbol{y}^\top = \boldsymbol{x}_+\boldsymbol{x}_-^\top + \boldsymbol{x}_-\boldsymbol{x}_+^\top$. In each case the structure is the natural restriction of the corresponding 3D one, re-expressed in the planar coordinate system. In each case it projects isomorphically into each image, either by the usual 3×4 3D projection matrix (using 3D coordinates), or by the corresponding 3×3 world-plane to image homography $\boldsymbol{H}$ (using scene plane coordinates). Hence, each image inherits a pair of circular points $\boldsymbol{H}_i\,\boldsymbol{x}_\pm$ and the corresponding direction basis $\boldsymbol{H}_i\,(\boldsymbol{x}\;\boldsymbol{y})$, line at infinity $\boldsymbol{l}_\infty\,\boldsymbol{H}_i^{-1}$ and 3×3 rank 2 absolute line quadric $\boldsymbol{H}_i\,\boldsymbol{Q}\,\boldsymbol{H}_i^\top$. As the columns of the planar $\boldsymbol{U}$ matrix represent *bona fide* 3D direction vectors (albeit expressed in the planar coordinate system), their images still satisfy the autocalibration constraints (1):

$$\boldsymbol{C}_i\,\boldsymbol{H}_i\,\boldsymbol{U} \sim \boldsymbol{R}_{3\times 2} \tag{8}$$

where $\boldsymbol{R}_{3\times 2}$ contains the first two columns of a 3×3 rotation matrix. Multiplying on the left by the transpose to eliminate the unknown rotation coefficients gives (*c.f.* (2)):

$$\boldsymbol{U}^\top \boldsymbol{H}_i^\top\,\boldsymbol{\omega}_i^{-1}\,\boldsymbol{H}_i\,\boldsymbol{U} \;\sim\; \boldsymbol{I}_{2\times 2} \tag{9}$$

Splitting this into components gives the form of the constraints used by our planar autocalibration algorithm:

$$\|\boldsymbol{u}_i\|^2 = \|\boldsymbol{v}_i\|^2, \quad 2\,\boldsymbol{u}_i \cdot \boldsymbol{v}_i = 0 \qquad \text{where} \qquad (\boldsymbol{u}_i, \boldsymbol{v}_i) \;\equiv\; \boldsymbol{C}_i\,\boldsymbol{H}_i\,(\boldsymbol{x}, \boldsymbol{y}) \tag{10}$$

These constraints say that any two orthonormal direction vectors in the world plane project under the calibrated world-plane to image homography $\boldsymbol{C}_i\,\boldsymbol{H}_i$ to two orthonormal vectors in the camera

frame. Equivalently, the (calibrated) images of the circular points $\boldsymbol{x}_\pm = \frac{1}{\sqrt{2}}(\boldsymbol{x} \pm i\boldsymbol{y})$ lie on the image of the (calibrated) absolute conic:

$$(\boldsymbol{H}_i \boldsymbol{x}_\pm)^\top \boldsymbol{\omega}^{-1} (\boldsymbol{H}_i \boldsymbol{x}_\pm) \;=\; \|\boldsymbol{u}_{i\pm}\|^2 \;=\; 0 \qquad \text{where} \qquad \boldsymbol{u}_{i\pm} \;\equiv\; \boldsymbol{C}_i \boldsymbol{H}_i \boldsymbol{x}_\pm \tag{11}$$

All of the above constraints are valid in arbitrary projective image and world-plane frames, except that $(\boldsymbol{x}, \boldsymbol{y})$ are no longer orthonormal. As always, $(\boldsymbol{x}, \boldsymbol{y})$ are defined only up to a 2×2 orthogonal mixing, and we can use this gauge freedom to require that $\boldsymbol{x} \cdot \boldsymbol{y} = 0$.

Our planar autocalibration method is based on direct numerical minimization of the residual error in the constraints (10) from several images, over the unknown direction basis $(\boldsymbol{x}, \boldsymbol{y})$ and any combination of the five intrinsic calibration parameters f, a, s, u_0 and v_0. The input data is the set of world plane to image homographies $\boldsymbol{H}_i$ for the images, expressed with respect to an arbitrary projective frame for the world plane. In particular, if the plane is coordinatized by its projection into some key image (say image 1), the inter-image homographies $\boldsymbol{H}_{i1}$ can be used as input.

Four independent parameters are required to specify the Euclidean structure of a projective plane: the 6 components of $(\boldsymbol{x}, \boldsymbol{y})$ modulo scale and the single d.o.f. of a 2×2 rotation; or the $3 \cdot 4/2 = 6$ components of a 3×3 absolute line quadric $\boldsymbol{Q}$ modulo scale and the rank 2 constraint $\det \boldsymbol{Q} = 0$; or the 2 d.o.f. of the plane's line at infinity, plus the 2 d.o.f. of two circular points on it. Since equations (9), (10) or (11) give two independent constraints for each image, $\lceil \frac{n+4}{2} \rceil$ images are required to estimate the Euclidean structure of the plane and n intrinsic calibration parameters. Two images suffice to recover the structure if the calibration is known, three are required if the focal length is also estimated, four for the perspective f, u_0, v_0 model, and five if all 5 intrinsic parameters are unknown.

2.3 Camera Parametrization

We have not yet made the camera parametrization explicit, beyond saying that it is given by the upper triangular matrices $\boldsymbol{K}$ or $\boldsymbol{C} = \boldsymbol{K}^{-1}$. For autocalibration methods which fix some parameters while varying others, it makes a difference which parametrization is used. I prefer the following form motivated by a zoom lens followed by an affine image-plane coordinatization:

$$\boldsymbol{K} = \begin{pmatrix} f & f\,s & u_0 \\ 0 & f\,a & v_0 \\ 0 & 0 & 1 \end{pmatrix} \qquad\qquad \boldsymbol{C} = \boldsymbol{K}^{-1} = \frac{1}{f\,a} \begin{pmatrix} a & -s & s\,v_0 - a\,u_0 \\ 0 & 1 & -v_0 \\ 0 & 0 & f\,a \end{pmatrix}$$

Here, if standard pixel coordinates are used, $f = \alpha_u$ is the focal length in u-pixels, $s = -\tan\theta_{\text{skew}}$ is the dimensionless geometric skew, $a = \alpha_v/(\alpha_u \cos\theta_{\text{skew}})$ is the dimensionless $v : u$ aspect ratio, and (u_0, v_0) are the pixel coordinates of the principal point. However pixel coordinates are *not* used in the optimization routine below. Instead, a nominal calibration is used to standardize the parameters to nominal values $f = a = 1$, $s = u_0 = v_0 = 0$, and all subsequent fitting is done using the above model with respect to these values.

3 Algebraic *vs.* Statistical Error

Many vision problems reduce to minimizing the residual violation of some vector of nonlinear constraints $\mathbf{e}(\mathbf{x}, \boldsymbol{\mu}) \approx \mathbf{0}$ over parameters $\boldsymbol{\mu}$, given fixed noisy measurements $\mathbf{x}$ with known covariance $\mathbf{V}_\mathbf{x}$. Often, heuristic error metrics such as the **algebraic error** $\|\mathbf{e}(\mathbf{x}, \boldsymbol{\mu})\|^2$ are taken as the

target for minimization. However, such approaches are statistically sub-optimal and if used uncritically can lead to (*i*) very significant bias in the results and (*ii*) severe constriction of the domain of convergence of the optimization method. Appropriate **balancing** or **preconditioning** (numerical scaling of the variables and constraints, *e.g.* as advocated in [7, 8] or any numerical optimization text) is the first step towards eliminating such problems, but it is not the whole story. In any case it begs the question of what *is* "balanced". It is *not* always appropriate to scale all variables to $\mathcal{O}(1)$. In fact, in the context of parameter estimation, "balanced" simply means "close to the underlying **statistical error measure**"[6]

$$\chi^2_{\mathbf{e}} \;\approx\; \mathbf{e}^\top \mathbf{V}_{\mathbf{e}}^{-1} \mathbf{e} \quad \text{where} \quad \mathbf{V}_{\mathbf{e}} \;\approx\; \tfrac{\mathbf{De}}{\mathbf{Dx}} \mathbf{V}_{\mathbf{x}} \tfrac{\mathbf{De}}{\mathbf{Dx}}^\top \quad \text{is the covariance of } \mathbf{e}$$

Ideally one would like to optimize the statistical cost (*i.e.* log likelihood). Unfortunately, this is often rather complicated owing to the matrix products and (pseudo-)inverse, and simplifying assumptions are often in order. I feel that this pragmatic approach is the *only* acceptable way to introduce algebraic error measures — as explicit, controlled approximations to the underlying statistical metric. Given that the extra computation required for a suitable approximation is usually minimal, while the results can be substantially more accurate, it makes little sense to iteratively minimize an algebraic error without such a validation step.

One very useful simplification is to ignore the dependence of $\mathbf{V}_{\mathbf{e}}^{-1}$ on $\boldsymbol{\mu}$ in cost function derivatives. This gives **self-consistent** or **iterative re-weighting** schemes (*e.g.* [12]), where $\mathbf{V}_{\mathbf{e}}$ is treated as a constant within each optimization step, but updated at the end of it. One can show that the missing terms effectively displace the cost derivative evaluation point from the measured $\mathbf{x}$ to a first order estimate of the true underlying value $\mathbf{x}_0$ [21]. For the most part this makes little difference unless the constraints are strongly curved on the scale of $\mathbf{V}_{\mathbf{x}}$.

For our autocalibration method, the statistical error splits into independent terms for each image[7]. For want of a more specific error model, we assume that the components of the data $\mathbf{x}$ (here, the $\boldsymbol{H}_i$ in nominally calibrated coordinates) are i.i.d.: $E\left[\Delta \boldsymbol{H}^A_B \, \Delta \boldsymbol{H}^C_D\right] \approx \epsilon \cdot \delta^{AC} \delta_{BD}$ where ϵ is a noise level[8]. From here it is straightforward to find and invert the constraint covariance. For the planar autocalibration constraint (10), and assuming that we have enforced the gauge constraint $\boldsymbol{x} \cdot \boldsymbol{y} = 0$, the constraint covariance is

$$4\epsilon \cdot \begin{pmatrix} \boldsymbol{x}^2 \, \boldsymbol{a}_i^2 + \boldsymbol{y}^2 \, \boldsymbol{b}_i^2 & (\boldsymbol{x}^2 - \boldsymbol{y}^2) \, \boldsymbol{a}_i \cdot \boldsymbol{b}_i \\ (\boldsymbol{x}^2 - \boldsymbol{y}^2) \, \boldsymbol{a}_i \cdot \boldsymbol{b}_i & \boldsymbol{x}^2 \, \boldsymbol{b}_i^2 + \boldsymbol{y}^2 \, \boldsymbol{a}_i^2 \end{pmatrix} \qquad \text{where} \quad (\boldsymbol{a}_i, \boldsymbol{b}_i) \;\equiv\; \boldsymbol{C}_i^\top \, (\boldsymbol{u}_i, \boldsymbol{v}_i) \;=\; \boldsymbol{\omega}_i^{-1} \boldsymbol{H}_i \, (\boldsymbol{x}, \boldsymbol{y})$$

In this case, numerical experience indicates that the off-diagonal term is seldom more than a few percent of the diagonal ones, which themselves are approximately equal for each image, but differ

[6] $\mathbf{e}$ is a random variable through its dependence on $\mathbf{x}$. Assuming that the uncertainty is small enough to allow linearization and that $\mathbf{x}$ is centred on some underlying $\mathbf{x}_0$ satisfying $\mathbf{e}(\mathbf{x}_0, \boldsymbol{\mu}_0) = 0$ for some parameter value $\boldsymbol{\mu}_0$, $\mathbf{e}(\mathbf{x}, \boldsymbol{\mu}_0)$ has mean $\mathbf{0}$ and the above covariance. It follows that $\mathbf{e}^\top \mathbf{V}_{\mathbf{e}}^{-1} \mathbf{e}$ is approximately a $\chi^2_{\text{rank}(\mathbf{e})}$ variable near $\boldsymbol{\mu}_0$, which can be minimized to find a maximum likelihood estimate of $\boldsymbol{\mu}$.

[7] We (perhaps unwisely) ignore the fact that the $\boldsymbol{H}_i$ are correlated through their mutual dependence on the base image. The base image is treated just like any other in the sum.

[8] This model is undoubtedly over-simplistic. Balancing should make their variances similar, but in reality the components are most unlikely to be independent. We should at very least subtract a diagonal term $\boldsymbol{H}^A_B \boldsymbol{H}^C_D / \|\boldsymbol{H}^A_B\|^2$, as variations proportional to $\boldsymbol{H}$ make no projective difference. However this makes no difference here, as when contracted with $\nabla \mathbf{e}$'s it just gives back $\mathbf{e}(\mathbf{x}_0)$'s which vanish. This *had* to happen: correctly weighted error terms must be insensitive to projective scale factors, and hence have total homogeneity 0 in their projective-homogeneous variables.

by as much as a factor of 2–3 between images[9]. Hence, we drop the off-diagonal term to give an autocalibration method based on self-consistent optimization of the diagonal cost function

$$\sum_{i=1}^{m} \left(\frac{(\|\boldsymbol{u}_i\|^2 - \|\boldsymbol{v}_i\|^2)^2/4}{\boldsymbol{x}^2 \|\boldsymbol{C}_i^\top \boldsymbol{u}_i\|^2 + \boldsymbol{y}^2 \|\boldsymbol{C}_i^\top \boldsymbol{v}_i\|^2} + \frac{(\boldsymbol{u}_i \cdot \boldsymbol{v}_i)^2}{\boldsymbol{x}^2 \|\boldsymbol{C}_i^\top \boldsymbol{v}_i\|^2 + \boldsymbol{y}^2 \|\boldsymbol{C}_i^\top \boldsymbol{u}_i\|^2} \right) \quad \text{where } (\boldsymbol{u}_i, \boldsymbol{v}_i) \equiv \boldsymbol{C}_i \boldsymbol{H}_i (\boldsymbol{x}, \boldsymbol{y}) \tag{12}$$

In our synthetic experiments, this statistically motivated cost function uniformly reduced the ground-truth standard deviation of the final estimates by about 10% as compared to the best carefully normalized algebraic error measures. This is a modest but useful improvement, obtained without any measurable increase in run time. The improvement would have been much larger if the error model had been less uniform in the standardized coordinates. Perhaps most importantly, the statistical cost is almost completely immune to mis-scaling of the variables, which is certainly *not* true of the algebraic ones which deteriorated very rapidly for mis-scaling factors greater than about 3.

4 Planar Autocalibration Algorithm

Numerical Method: Our planar autocalibration algorithm is based on direct numerical minimization of the m-image cost function (12), with respect to the direction basis $\{\boldsymbol{x}, \boldsymbol{y}\}$ and any subset of the 5 internal calibration parameters focal length f, aspect ratio a, skew s, and principal point (u_0, v_0). There are 4 d.o.f. in $\{\boldsymbol{x}, \boldsymbol{y}\}$ — 6 components defined up to an overall mutual rescaling and a 2×2 orthogonal mixing — so the optimization is over 5–9 parameters in all. Numerically, the 6 component $(\boldsymbol{x}, \boldsymbol{y})$ vector is locally projected onto the subspace orthogonal to its current scaling and mixing d.o.f. by Householder reduction (*i.e.* effectively a mini QR decomposition). As mentioned in section 2, the mixing freedom allows us to enforce the gauge condition $\boldsymbol{x} \cdot \boldsymbol{y} = 0$. Although not essential, this costs very little (one Jacobi rotation) and we do it at each iteration as an aid to numerical stability.

A fairly conventional nonlinear least squares optimization method is used: Gauss-Newton iteration based on Choleski decomposition of the normal equations. As always, forming the normal equations gives a fast, relatively simple method but effectively squares the condition number of the constraint Jacobian. This is not a problem so long as intermediate results are stored at sufficiently high precision: double precision has proved more than adequate for this application.

As with any numerical method, care is needed to ensure stability should the numerical conditioning become poor. Our parametrization of the problem guarantees that all variables are of $\mathcal{O}(1)$ and fairly well decoupled, so preconditioning is not necessary. The Choleski routine uses diagonal pivoting and Gill & Murray's [4] minimum-diagonal-value regularization to provide local stability. The regularizer is also manipulated in much the same way as a Levenberg-Marquardt parameter to ensure that each step actually reduces the cost function. We also limit the maximum step size for each variable, relatively for the positive, multiplicative parameters f and a and absolutely for the others. Both the regularizer and the step size limits are activated fairly often in practice, the regularizer at any time, and the step limit usually only during the first 1–2 iterations. The method

[9] This was to be expected, since we chose everything to be well-scaled except that the $\boldsymbol{H}$ normalizations may differ somewhat from their 'correct' Euclidean ones, and our noise model is uniform in an approximately calibrated frame. If any of these conditions were violated the differences would be *much* greater.

terminates when the step size converges to zero, with additional heuristics to detect thrashing. Convergence within 5–10 iterations is typical.

Prior over Calibrations: We also allow for a simple user-defined prior distribution on the calibration parameters. Even if there is no very strong prior knowledge, it is often advisable to include a weak prior in statistical estimation problems as a form of regularization. If there are unobservable parameter combinations (*i.e.* that make little or no difference to the fit), optimal, unbiased estimates of these are almost always extremely sensitive to noise. Adding a weak prior makes little difference to strong estimates, but significantly reduces the variability of weak ones by biasing them towards reasonable default values. A desire to "keep the results unbiased" is understandable, but limiting the impact of large fluctuations on the rest of the system may be more important in practice.

Default priors are also useful to ensure that parameters retain physically meaningful values. For example, we use heuristic priors of the form $(x/x_0 - x_0/x)^2$ for f and a, to ensure that they stay within their physically meaningful range $(0, \infty)$. This is particularly important for autocalibration problems, where degenerate motions occur frequently. In such cases the calibration can not be recovered uniquely. Instead there is a one or more parameter family of possible solutions, usually including physically unrealizable ones. A numerical method (if it converges at all) will converge to an arbitrary one of these solutions, and for sanity it pays to ensure that this is a physically feasible one not too far from the plausible range of values. A weak default prior is an effective means of achieving this, and seems no more unprincipled than any other method. This is not to say that such degeneracies should be left unflagged, but simply that whatever cleaning up needs to be done will be easier if it starts from reasonable default values.

Initialization: The domain of convergence of the numerical optimization is reasonably large and for many applications it will probably be sufficient to initialize it from fixed default values. The most critical parameters are the focal length f and the number and angular spread of the views. For example, if f can only be guessed within a factor of 2 and all 5 parameters f, a, s, u_0, v_0 are left free, about 9–10 images spread by more than about 10° seem to be required for reliable convergence to the true solution. Indeed, with 5 free parameters and the theoretical minimum of only 5–6 images, even an *exact* initialization is not always sufficient to eliminate false solutions (*i.e.* with slightly smaller residuals than the true one).

These figures assume that the direction basis $\boldsymbol{x}, \boldsymbol{y}$ is completely unknown. Information about this is potentially very valuable and should be used if available. Knowledge of the world-plane's horizon (line at infinity) removes 2 d.o.f. from $\boldsymbol{x}, \boldsymbol{y}$ and hence reduces the number of images required by one, and knowledge of its Euclidean structure (but not the positions of points on it) eliminates another image. Even if not directly visible, horizons can be recovered from known 3D parallelism or texture gradients, or bounded by the fact that visible points on the plane must lie inside them. We will not consider these types of constraints further here.

If a default initialization is insufficient to guarantee convergence, several strategies are possible. One quite effective technique is simply to use a preliminary optimization over $\boldsymbol{x}, \boldsymbol{y}$ or $\boldsymbol{x}, \boldsymbol{y}, f$ to initialize a full one over all parameters. More generally, some sort of initialization search over f, $\boldsymbol{x}$ and $\boldsymbol{y}$ is required. Perhaps the easiest way to approach this is to fix nominal values for all the calibration parameters except f, and to recover estimates for $\boldsymbol{x}, \boldsymbol{y}$ as a function of f from a single pair of images as f varies. These values can then be substituted into the autocalibration constraints for the other images, and the overall most consistent set of values chosen to initialize the optimization routine. The estimation of $\boldsymbol{x}(f), \boldsymbol{y}(f)$ reduces to the classical photogrammetric

problem of the relative orientation of two calibrated cameras from a planar scene, as the Euclidean structure is easily recovered once the camera poses are known. In theory this problem could be solved in closed form (the most difficult step being a 3×3 eigendecomposition) and optimized over f analytically. But in practice this would be rather messy and I have preferred to implement a coarse numerical search over f. The search uses a new SVD-based planar relative orientation method (see appendix 1) related to Wunderlich's eigendecomposition approach [25]. The camera pose and planar structure are recovered directly from the SVD of the inter-image homography. As always with planar relative orientation, there is a two-fold ambiguity in the solution, so both solutions are tested. In the implemented routine, the solutions for each image against the first one, and for each f in a geometric progression, are substituted into the constraints from all the other images, and the most consistent overall values are chosen.

If the full 5 parameter camera model is to be fitted, Hartley's 'rotating camera' method [6] can also be used for initialization. It works well *provided* (*i*) the camera translations are smaller than or comparable to the distance to the plane; (*ii*) no point on the plane is nearly fixated from a constant distance. (For such a point $\boldsymbol{x}$, $\boldsymbol{\omega} + \mu \boldsymbol{x}\boldsymbol{x}^\top$ is an approximate solution of Hartley's equation $\boldsymbol{H}\boldsymbol{\omega}\boldsymbol{H}^\top = \boldsymbol{\omega}$ for any μ, *i.e.* $\boldsymbol{\omega}$ can not be estimated uniquely, even for small translations).

5 Experiments

Synthetic data: The method has been implemented in C and tested on both real and synthetic images. For the synthetic experiments, the camera roughly fixates a point on the plane from a constant distance, from randomly generated orientations varying by (by default) $\pm 30^\circ$ in each of the three axes. The camera calibration varies randomly about a nominal focal length of 1024 pixels and unit aspect ratio, by $\pm 30\%$ in focal length f, $\pm 10\%$ in aspect ratio a, ± 0.01 in dimensionless skew s, and ± 50 pixels in principal point (u_0, v_0). (These values are standard deviations of log-normal distributions for f, a and normal ones for s, u_0, v_0). The scene plane contains by default 40 visible points, projected into the 512×512 images with a Gaussian noise of ± 1 pixel. Before the homographies are estimated and the method is run, the pixel coordinates are centred and scaled to a nominal focal length of 1: $(u, v) \rightarrow (u - 256, v - 256)/1024$. The output is classed as a 'success' or 'failure' according to fixed thresholds on the size of its deviation from the true value. Only successes count towards the accuracy estimates. The usual mode of failure is convergence to a false solution with extremely short focal length (say < 50 pixels). However when the angular spread of the views is small or there are only a few images, random fluctuations sometimes take a "correct" but highly variable solution outside the (generously set) thresholds. Conversely, there is occasionally convergence to a false solution within the threshold. Thus, when the failure rate is high, neither it nor the corresponding error measure (nor, for that matter, the results!) are accurate. The optimization typically converges within 5–10 iterations, although more may be needed for degenerate problems. The run time is negligible: on a Pentium 133, about 0.5 milliseconds per image if the default initialization is used, or 2.0 with a fairly fine initialization search over f.

Figure 1 gives some illustrative accuracy and reliability results, concentrating on the estimation of focal length f. First consider the plots where all 5 calibration parameters are estimated. The error scales roughly linearly with noise and inversely with the angular spread of the views. It drops rapidly as the first few images are added, but levels off after about 10 images. The failure rate increases rapidly for more than about 2–3 pixels noise, and is also unacceptably high for near-minimal numbers of images (within 1–2 of the minimum) and small angular spreads (less than

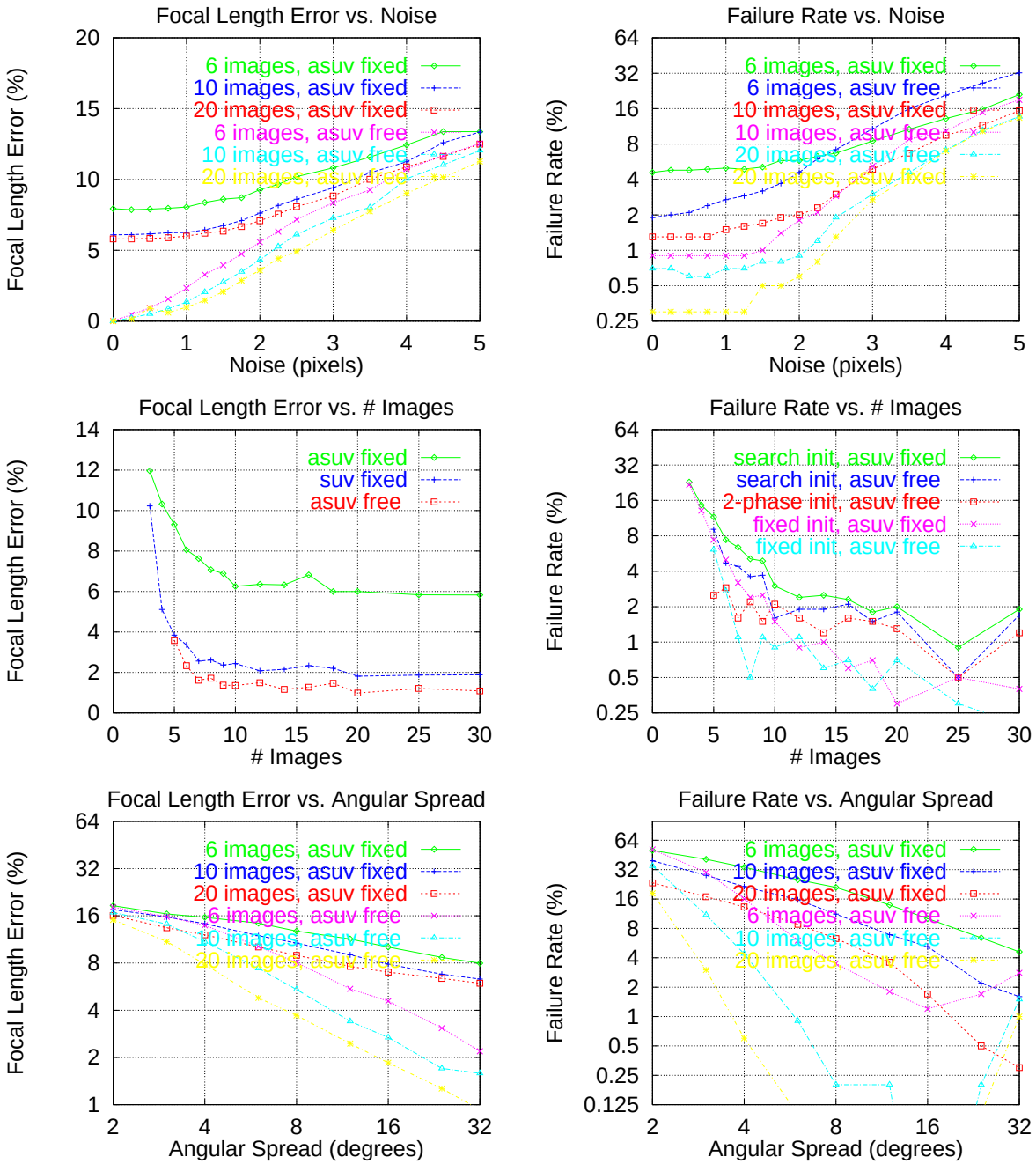

Figure 1: Error in estimated focal length f and failure rate *vs.* image noise, number of images and angular spread of cameras. Each value is the average of 1000 trials. The aspect ratio a, skew s, and principal point (u_0, v_0) are either fixed at their nominal values, or allowed to vary freely, as indicated. The method is initialized from the nominal calibration, except that in the failure *vs.* images plot we also show the results for initialization by numerical search over f, and by a preliminary fit over f alone ('2-phase').

about 10°). however, it decreases rapidly as each of these variables is increased. It seems to be difficult to get much below about 1% failure rate with the current setup. Some of these failures are probably the result of degeneracies in the randomly generated problems, but most of them are caused by convergence to a false solution with implausible parameters, either very small f (less

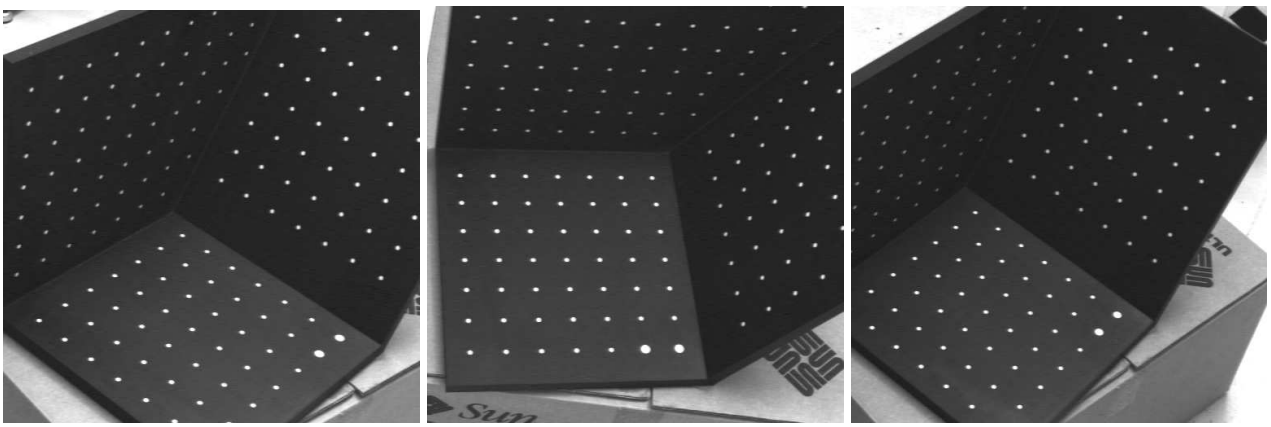

Figure 2: Several images from our calibration sequence.

than about 50) or a far from 1. The initialization method has little impact on the reliability. In fact, in these experiments the default initialization proved more reliable than either numerical search over f, or an initial optimization over f alone. The reason is simply that we do not assume prior knowledge of *any* of the calibration parameters. An initialization search over f must fix a, s, u_0, v_0 at their inaccurate nominal values, and this is sometimes enough to make it miss the true solution entirely. This also explains the poor performance of the methods which hold a, s, u_0, v_0 fixed and estimate f alone. As the graphs of error *vs.* noise and number of images show, errors in a, s, u_0, v_0 lead to a significant bias in f, but most of this can be eliminated by estimating a as well as f. The initialization search over f also becomes much more reliable (*e.g.* 0.05% failure rate for 10 images, 30° spread and 1 pixel noise) if a and s are accurate to within a few percent. Here and elsewhere, it is only worthwhile to fix parameters if they are reliably known to an accuracy better than their measured variabilities, *e.g.* here for 1 pixel noise and 10 images, to about 0.003 for a, s or 20 pixels for u_0, v_0.

For conventional calibration, f is often said the most difficult parameter to estimate, and also the least likely to be known *a priori*. In contrast, a and s are said to be estimated quite accurately, while u_0 and v_0 — although variable — are felt to have little effect on the overall results. A more critical, quantitative view is to compare the *relative* accuracy $|\Delta f/f|$ to the dimensionless quantities $|\Delta a|$, $|\Delta s|$, $|\Delta u_0/f|$ and $|\Delta v_0/f|$. Errors in these contribute about equally to the overall geometric accuracy (*e.g.* reconstruction errors of 3D visual ray directions). Conversely, other things being equal, geometric constraints such as the autocalibration ones typically constrain each of these quantities to about the same extent. Hence a good rule of thumb is that for autocalibration (and many other types of calibration) $|\Delta u_0/f|$ and $|\Delta v_0/f|$ are of the same order of magnitude as $|\Delta f/f|$, while $|\Delta a|$ and $|\Delta s|$ are usually somewhat smaller if there is cyclotorsion or other aspect ratio constraints, but larger if there are none (*e.g.* if the rotation axis direction is almost constant). These rules are well borne out in all the experiments reported here: we always find $|\Delta u_0| \approx |\Delta v_0| \approx |\Delta f|$, while $|\Delta a|$ and $|\Delta s|$ are respectively about one fifth, one half, and one tenth of $|\Delta f/f|$ for the synthetic experiments, the real experiments below, and the Faugeras-Toscani calibration used in the real experiments.

Real data: We have run the method on several non-overlapping segments of a sequence of about 40 real images of a calibration grid (see fig. 2). Only the 49 (at most) points on the base plane of the grid are used. (It would be straightforward to extend the algorithm to handle several

planes, but there seems little point as a non-planar autocalibration method could be used in this case). The motion was intended to be general within the limits of the 5 d.o.f. robot used to produce it, but is fairly uniform within each subsequence. Visibility considerations limited the total angular displacement to about 40°, and significantly less within each subsequence. The sample means and standard deviations over a few non-overlapping subsequences for (*i*) f alone, and (*ii*) all 5 parameters, are as follows (the errors are observed sample scatters, *not* estimates of absolute accuracy):

	f only	f	a	s	u_0	v_0
calibration	-	1515 ± 4	0.9968 ± 0.0002	-	271 ± 3	264 ± 4
6 images	1584 ± 63	1595 ± 63	0.9934 ± 0.0055	0.000 ± 0.001	268 ± 10	271 ± 22
8 images	1619 ± 25	1614 ± 42	0.9890 ± 0.0058	-0.005 ± 0.005	289 ± 3	320 ± 26
10 images	1612 ± 19	1565 ± 41	1.0159 ± 0.0518	-0.004 ± 0.006	273 ± 5	286 ± 27

The 'calibrated' values are the averaged results of several single-image Faugeras-Toscani calibrations using all visible points on the grid. Looking at the table, the results of the autocalibration method seem usable but not quite as good as I would have expected on the basis of the synthetic experiments. This may just be the effect of the small angular range within each subsequence, but the estimates of f seem suspiciously high and it may be that some small systematic error has occurred during the processing. Further work is required to check this. Note that in this case, fixing a, s, u_0, v_0 appears to have the desired effect of decreasing the variability of the estimated f without perturbing its value very much.

6 Summary

In summary, we have shown how autocalibration problems can be approached using a projective representation of orthogonal 3D direction frames, and used this to derive a practical numerical algorithm for the autocalibration of a moving projective camera viewing a planar scene. The method is based on the 'rectification' of inter-image homographies. It requires a minimum of 3 images if only the focal length is estimated, or 5 for all five internal parameters. Adding further images significantly increases both the reliability and the accuracy, up to a total of about 9–10. An angular spread between the cameras of at least 10–20° is recommended.

The priorities for future work are the initialization problem and the detection of false solutions (or possibly the production of multiple ones). Although the current numerical method is stable even for degenerate motions (and hence gives *a* possible solution), it does not attempt to detect and flag the degeneracy. This could be done, *e.g.* by extracting the null space of the estimated covariance matrix. It would also be useful to have autocalibration methods that could estimate lens distortion. This should be relatively simple in the planar case, as distortion can be handled during homography estimation.

Appendix 1: Relative Orientation from Planar Scenes

This appendix describes a simple method for the relative orientation of two calibrated cameras from an unknown but planar scene (*e.g.* 4 or more coplanar 3D points), used to initialize our planar

autocalibration method by a numerical search over focal length f with the remaining calibration parameters fixed at their nominal values. The procedure is similar to Wunderlich's method [25], but is based on Singular Value Decomposition (SVD) of the inter-image homography $\boldsymbol{H}$ rather than eigen-decomposition of $\boldsymbol{H}^\top\boldsymbol{H}$. This makes it more direct and a little stabler. As with any plane based method, even if 'twisted pair' solutions are eliminated, there are always two possible solutions for the orientation and 3D structure. Both are visible, internally self-consistent and have small epipolar residual, although extra information such as the horizon line may help to disambiguate them.

We use homogeneous, calibrated, lens-distortion-corrected image coordinates, so $\boldsymbol{H} = \boldsymbol{K}_2^{-1}\boldsymbol{H}'\boldsymbol{K}_1$ where $\boldsymbol{H}'$ is uncalibrated homography and $\boldsymbol{K}_i$ are the two camera calibration matrices. The sign of $\boldsymbol{H}$ should be chosen so that $\boldsymbol{x}_2{}^\top\boldsymbol{H}\boldsymbol{x}_1 > 0$ for corresponding image points $\boldsymbol{x}_1, \boldsymbol{x}_2$. Let $\boldsymbol{H} = \mathrm{U}\boldsymbol{S}\boldsymbol{V}^\top$ be the SVD of $\boldsymbol{H}$, where U and $\boldsymbol{V}$ are 3×3 rotation matrices and $\boldsymbol{S} = \mathrm{diag}(s_1, s_2, s_3)$ is positive decreasing diagonal $s_1 \geq s_2 \geq s_3 \geq 0$. Denote the associated columns of U and $\boldsymbol{V}$ by $\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3$ and $\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3$.

In the coordinate frame of the first camera, let the 3D plane be $\boldsymbol{n} \cdot \boldsymbol{x} = 1/\zeta$, where $\boldsymbol{n}$ is the *outward* (away from camera) normal. $\zeta = 1/z > 0$ is the inverse distance to the plane. In this frame the first camera has 3×4 projection matrix $\boldsymbol{P}_1 = (\boldsymbol{I}_{3\times3}|\boldsymbol{0})$. Let the matrix of the second camera be $\boldsymbol{P}_2 = \boldsymbol{R}(\boldsymbol{I}_{3\times3}| - \boldsymbol{t})$ where $\boldsymbol{t}$ is the inter-camera translation (*i.e.* the second camera's optical centre) and $\boldsymbol{R}$ the inter-camera rotation. Then the homography from image 1 to image 2 is $\boldsymbol{H} = \boldsymbol{R}\boldsymbol{H}_1$ where $\boldsymbol{H}_1 = \boldsymbol{I}_{3\times3} - \zeta\boldsymbol{t}\boldsymbol{n}^\top$. (For a 3D point $\boldsymbol{x}$ on the plane $\boldsymbol{H}\boldsymbol{x} = \boldsymbol{R}(\boldsymbol{x} - \zeta\boldsymbol{t}\boldsymbol{n}^\top\boldsymbol{x}) = \boldsymbol{R}(\boldsymbol{x} - \boldsymbol{t}) \sim \boldsymbol{P}_2\boldsymbol{x}$, since $\zeta\boldsymbol{n}^\top\boldsymbol{x} = 1$ there. Treating $\boldsymbol{x}$ as a point in image 1 changes only the overall scale factor). Only the product $\zeta\boldsymbol{t}\boldsymbol{n}^\top$ is recoverable, so we normalize to $\|\boldsymbol{t}\| = \|\boldsymbol{n}\| = 1$ (*i.e.* the plane distance $1/\zeta$ is measured in units of the baseline $\|\boldsymbol{t}\|$) and use visibility tests to work out the allowable signs.

The SVD's of $\boldsymbol{H} = \mathrm{U}\boldsymbol{S}\boldsymbol{V}$ and $\boldsymbol{H}_1 = \mathrm{U}_1\boldsymbol{S}\boldsymbol{V}$ are identical up to a factor of $\boldsymbol{R}$: $\mathrm{U} = \boldsymbol{R}\mathrm{U}_1$. $\boldsymbol{H}_1$ leaves the cross-product vector $\boldsymbol{t} \wedge \boldsymbol{n}$ invariant. If the singular values are distinct, $\boldsymbol{t} \wedge \boldsymbol{n}$ must correspond to a singular vector. It turns out to be the *middle* one $\boldsymbol{v}_2$, so the 'correct' normalization for $\boldsymbol{H}$ is $\boldsymbol{H} := \boldsymbol{H}/s_2$, $\boldsymbol{S} := \boldsymbol{S}/s_2$. Replace (s_1, s_2, s_3) by $(s_1/s_2, 1, s_3/s_2)$.

Given that $\boldsymbol{t} \wedge \boldsymbol{n}$ corresponds to $\boldsymbol{v}_2$ in the image 1 frame, the $\{\boldsymbol{t}, \boldsymbol{n}\}$ subspace must be spanned by $\{\boldsymbol{v}_1, \boldsymbol{v}_3\}$, say $\boldsymbol{n} = \beta\,\boldsymbol{v}_1 - \alpha\,\boldsymbol{v}_3$, $\boldsymbol{n} \wedge (\boldsymbol{t} \wedge \boldsymbol{n}) \sim \alpha\,\boldsymbol{v}_1 + \beta\,\boldsymbol{v}_3$ for some $\alpha^2 + \beta^2 = 1$. Any direction orthogonal to $\boldsymbol{n}$ — in particular $\boldsymbol{n} \wedge (\boldsymbol{t} \wedge \boldsymbol{n})$ — has norm unchanged by $\boldsymbol{H}$ or $\boldsymbol{H}_1$, whence $(\alpha\,s_1)^2 + (\beta\,s_3)^2 = \alpha^2 + \beta^2$ or $(\alpha, \beta) \sim (\pm\sqrt{1 - s_3^2}, \pm\sqrt{s_1^2 - 1})$. Had we taken $\boldsymbol{t} \wedge \boldsymbol{n}$ to correspond to $\boldsymbol{v}_1$ or $\boldsymbol{v}_3$ above, there would have been no real solution here, so $\boldsymbol{v}_2$ is the only possibility.

Exactly the same argument on the left shows that $\boldsymbol{R}\,\boldsymbol{t} = -(\beta\,\boldsymbol{u}_1 + \alpha\,\boldsymbol{u}_3)$. As $\boldsymbol{t}$ is an eigenvector of $\boldsymbol{H}_1$ with eigenvalue $1 - \zeta\,\boldsymbol{n}\cdot\boldsymbol{t}$, we have $\boldsymbol{H}\,\boldsymbol{t} = (1 - \zeta\,\boldsymbol{n}\cdot\boldsymbol{t})\boldsymbol{R}\,\boldsymbol{t}$, whence $\boldsymbol{t} \sim \boldsymbol{H}^{-1}(\boldsymbol{R}\,\boldsymbol{t}) \sim \beta/s_1\,\boldsymbol{v}_1 + \alpha/s_3\,\boldsymbol{v}_3$ and (after simplification) $\zeta = s_1 - s_3$. The left singular vectors of $\boldsymbol{H}_1$ (columns $\boldsymbol{u}'_1, \boldsymbol{u}'_2, \boldsymbol{u}'_3$ of U_1) can be recovered by noting that $\boldsymbol{u}'_2 = \boldsymbol{v}_2$ and requiring that $\boldsymbol{t}$ be an eigenvector of $\boldsymbol{H}_1$, whence $\boldsymbol{u}'_1 = \gamma\,\boldsymbol{v}_1 + \delta\,\boldsymbol{v}_3, \boldsymbol{u}'_3 = \delta\,\boldsymbol{v}_1 - \gamma\,\boldsymbol{v}_3$ with (simplifying) $(\gamma, \delta) \sim (1 + s_1\,s_3, \pm\alpha\,\beta)$ and hence

$$\boldsymbol{R} = \mathrm{U}\,\mathrm{U}_1^\top = \mathrm{U}\begin{pmatrix} \gamma & 0 & \delta \\ 0 & 1 & 0 \\ -\delta & 0 & \gamma \end{pmatrix}\boldsymbol{V}^\top$$

In summary, the following OCTAVE/MATLAB routine generates the two possible orientation solutions given homography $\boldsymbol{H}$ (`unitize()` scales two variables so that their sum of squares is 1):

```
function [R1,t1,n1, R2,t2,n2, zeta] = homog_to_Rt(H)
   [U,S,V] = svd(H);
   s1 = S(1,1)/S(2,2);
   s3 = S(3,3)/S(2,2);
   zeta = s1-s3;
   a1 = sqrt(1-s3^2);
   b1 = sqrt(s1^2-1);
   [a,b] = unitize(a1,b1);
   [c,d] = unitize( 1+s1*s3, a1*b1 );
   [e,f] = unitize( -b/s1, -a/s3 );
   v1 = V(:,1); v3 = V(:,3);
   n1 = b*v1-a*v3;
   n2 = b*v1+a*v3;
   R1 = U*[c,0,d; 0,1,0; -d,0,c]*V';
   R2 = U*[c,0,-d; 0,1,0; d,0,c]*V';
   t1 = e*v1+f*v3;
   t2 = e*v1-f*v3;
   if (n1(3)<0)  t1 = -t1; n1 = -n1; end;
   if (n2(3)<0)  t2 = -t2; n2 = -n2; end;
end;
```

Although the above derivation uses $\boldsymbol{t} \wedge \boldsymbol{n}$ and friends, provided that $0 \ll \zeta = s_1 - s_3 \ll \infty$ (*i.e.* the plane is not too far away compared to the baseline $\boldsymbol{t}$, and the cameras stay on one side of it), the given routine is stable even if $\boldsymbol{t}$ is: (*i*) parallel to $\boldsymbol{n}$ ($\boldsymbol{t} \wedge \boldsymbol{n} \rightarrow \boldsymbol{0}$, $s_1 = s_2 = 1 > s_3$, $(\alpha, \beta) \sim (1, 0)$, $\boldsymbol{t} \sim \boldsymbol{n} \sim \boldsymbol{v}_3$, and $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ can not be clearly separated); (*ii*) anti-parallel to $\boldsymbol{n}$ (similar, with $s_1 > s_2 = s_3 = 1$); or (*iii*) orthogonal to $\boldsymbol{n}$ ($s_1 > s_2 = s_3 = 1$, $(\alpha, \beta) \sim (0, 1)$, $\boldsymbol{n} \sim \boldsymbol{v}_1$, $\boldsymbol{Rt} \sim \boldsymbol{u}_1$, but $\boldsymbol{v}_2, \boldsymbol{v}_3$ and $\boldsymbol{u}_2, \boldsymbol{u}_3$ can not be separated). For a distant plane $\zeta \rightarrow 0$, $\boldsymbol{S} \rightarrow \boldsymbol{I}_{3\times3}$ and α, β, $\boldsymbol{n}$, and $\boldsymbol{t}$ become unreliable but $\boldsymbol{R}$ should still be accurate. Of course, if the points used to estimate $\boldsymbol{H}$ are not well spread in the images, $\boldsymbol{H}$ and everything else can become inaccurate.

For our autocalibration purposes, we only need the two orthogonal directions $\boldsymbol{v}_2 \sim (\boldsymbol{t} \wedge \boldsymbol{n})$ and $\alpha\, \boldsymbol{v}_1 \pm \beta\, \boldsymbol{v}_3 \sim \boldsymbol{n} \wedge (\boldsymbol{t} \wedge \boldsymbol{n})$, so the routine is even simpler.

Appendix 2: Homography Factorization

Our planar autocalibration approach is based on scene plane to image homographies $\boldsymbol{H}_i$. In practice we can not estimate these directly, only the inter-image homographies $\boldsymbol{H}_{ij} = \boldsymbol{H}_i \boldsymbol{H}_j^{-1}$ induced by them. In theory this is not a problem as the formalism is invariant to projective deformations of the input frame, so we can choose scene plane coordinates derived from a key image (say image 1) and use the $\boldsymbol{H}_{i1}$ in place of the $\boldsymbol{H}_i$ (*i.e.* the unknown direction vectors $\boldsymbol{x}, \boldsymbol{y}$ are parametrized by their coordinates in the key image). This works reasonably well in practice, but there is a risk that inaccurate measurements or poor conditioning in the key image will have an undue influence on the overall numerical accuracy or stability of the method, since they potentially contribute coherently to all the $\boldsymbol{H}$'s. It would be useful to find a homography representation that does not single out a specific key image, but instead averages the uncertainty over all of them. This can be achieved by

a factorization method analogous to factorization-based projective structure and motion [20, 22][10].

This appendix describes the homography factorization algorithm. However note that it is *not* used in the final planar autocalibration routine as it turns out to give slightly *worse* results in practice. I am not sure why this happens. It may be that the scaling required for the homographies induces less than ideal error averaging, or that the resulting frame is in some way less well adapted to the calibration problem. In any case, it suggests that the use of a key image does not introduce too much bias in the calibration. Despite this, I have included a description of the factorization method here as I still think it is potentially useful for other applications.

Suppose we have estimated inter-image homographies $\boldsymbol{H}_{ij}$ between each pair of m images of a plane. In terms of some coordinate system on the plane which induces plane to image homographies $\boldsymbol{H}_i$ we have $\lambda_{ij}\boldsymbol{H}_{ij} \approx \boldsymbol{H}_i\boldsymbol{H}_j^{-1} + \text{noise}$, where the λ_{ij} are unknown scale factors. Write this as a big $(3m) \times (3m)$ rank 3 matrix equation

$$\begin{pmatrix} \lambda_{11}\boldsymbol{H}_{11} & \lambda_{12}\boldsymbol{H}_{12} & \cdots & \lambda_{1m}\boldsymbol{H}_{1m} \\ \lambda_{21}\boldsymbol{H}_{21} & \lambda_{22}\boldsymbol{H}_{22} & \cdots & \lambda_{2m}\boldsymbol{H}_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{m1}\boldsymbol{H}_{m1} & \lambda_{m2}\boldsymbol{H}_{m2} & \cdots & \lambda_{mm}\boldsymbol{H}_{mm} \end{pmatrix} \approx \begin{pmatrix} \boldsymbol{H}_1 \\ \boldsymbol{H}_2 \\ \vdots \\ \boldsymbol{H}_m \end{pmatrix} \begin{pmatrix} \boldsymbol{H}_1^{-1} & \boldsymbol{H}_2^{-1} & \cdots & \boldsymbol{H}_m^{-1} \end{pmatrix} + \text{noise}$$

As in the projective structure case, if we can recover a self-consistent set of scale factors λ_{ij}, the left hand side can be factorized to rank 3 using (*e.g.*) SVD or a fixed-rank power iteration method: $\mathbf{H}_{3m\times 3m} = \mathbf{U}_{3m\times 3}\mathbf{V}_{3\times 3m}$. Any such rank 3 factorization has the required noise-averaging properties and represents some 'numerically reasonable' choice of projective coordinates on the plane. For our purposes we need not insist that the 3×3 submatrices of $\mathbf{U}$ are exactly the inverses of those of $\mathbf{V}$, although — given that $\boldsymbol{H}_{ii} = \boldsymbol{I}$ — the inverse property is always approximately satisfied up to scale.

A suitable set of scale factors λ_{ij} can be found very simply by choosing a key image 1 and noting that up to scale $\boldsymbol{H}_{ij} \approx \boldsymbol{H}_{i1}\boldsymbol{H}_{1j}$. Resolving this approximate matrix proportionality by projecting it along $\boldsymbol{H}_{ij}$, we find that the quantities

$$\lambda_{ij} \equiv \frac{\text{Trace}\left((\boldsymbol{H}_{i1}\boldsymbol{H}_{1j}) \cdot \boldsymbol{H}_{ij}^{\top}\right)}{\text{Trace}(\boldsymbol{H}_{ij} \cdot \boldsymbol{H}_{ij}^{\top})}$$

are an approximately self-consistent set of scale factors. As in the projective structure case, the matrix of scale factors λ_{ij} is only defined up to independent overall rescalings of each row and each column. Numerically, it is highly advisable to balance the matrix so that all its elements are of order $\mathcal{O}(1)$ before applying it to the $\boldsymbol{H}_{ij}$'s and factorizing. Our balancing algorithm proceeds by alternate row and column normalizations as in the projective structure case [20], and converges within 2-3 iterations.

It may seem that using a key image to find the scale factors is likely to spoil the noise averaging properties of the factorization, but this is not so. Perturbations of the scales of $\boldsymbol{H}_{i1}$ and $\boldsymbol{H}_{1j}$ introduce no inconsistency, while other perturbations of order $\mathcal{O}(\epsilon)$ introduce errors only at $\mathcal{O}(\epsilon^2)$ in the projection of $\boldsymbol{H}_{i1}\boldsymbol{H}_{1j}$ along $\boldsymbol{H}_{ij}$ — and hence in the scale factors — as these matrices are proportional up to noise. At normal noise levels $\epsilon \ll \frac{1}{m}$, these errors are swamped by the $\mathcal{O}(\epsilon)$ ones arising from the explicit $\boldsymbol{H}_{i1}$ and $\boldsymbol{H}_{1j}$ terms in the factorization, so each image has roughly the

[10] Analogous methods also exist for 3D homographies (projective structure alignment, rank=4) and, more interestingly, for finding coherent sets of fundamental matrices or line projections (rank=6).

same total influence on the result (*provided* the λ_{ij} have been balanced appropriately). The same phenomenon is observed in the projective structure method: errors in the fundamental matrices and epipoles used to estimate the scales have very little effect.

References

[1] M. Armstrong, A. Zisserman, and R. Hartley. Self-calibration from image triplets. In B. Buxton and R. Cipolla, editors, *European Conf. Computer Vision*, pages 3–16, Cambridge, U.K., April 1996.

[2] O. Faugeras. Stratification of 3-d vision: Projective, affine, and metric representations. *J. Optical Society of America*, A 12(3):465–84, March 1995.

[3] O. Faugeras, Q.-T. Luong, and S. J. Maybank. Camera self calibration: Theory and experiments. In *European Conf. Computer Vision*, Santa Margherita Ligure, Italy, May 1992. Springer-Verlag.

[4] P. Gill, W. Murray, and M. Wright. *Practical Optimization*. Academic Press, 1981.

[5] R. Hartley. Euclidean reconstruction from multiple views. In 2^{nd} *Europe-U.S. Workshop on Invariance*, pages 237–56, Ponta Delgada, Azores, October 1993.

[6] R. Hartley. Self-calibration from multiple views with a rotating camera. In *European Conf. Computer Vision*, pages 471–8. Springer-Verlag, 1994.

[7] R. Hartley. In defence of the 8-point algorithm. In E. Grimson, editor, *IEEE Int. Conf. Computer Vision*, pages 1064–70, Cambridge, MA, June 1995.

[8] R. Hartley. Minimizing algebraic error. In *IEEE Int. Conf. Computer Vision*, Bombay, January 1998.

[9] A. Heyden and K. Åström. Euclidean reconstruction from constant intrinsic parameters. In *Int. Conf. Pattern Recognition*, pages 339–43, Vienna, 1996.

[10] A. Heyden and K. Åström. Euclidean reconstruction from image sequences with varying and unknown focal length and principal point. In *IEEE Conf. Computer Vision & Pattern Recognition*, Puerto Rico, 1997.

[11] R. Horaud and G. Csurka. Self-calibration and euclidean reconstruction using motions of a stereo rig. In *IEEE Int. Conf. Computer Vision*, January 1998.

[12] K. Kanatani. *Statistical Optimization for Geometric Computation: Theory and Practice*. Elsevier Science, Amsterdam, 1996.

[13] Q.-T. Luong and T. Viéville. Canonic representations for the geometries of multiple projective views. Technical Report UCB/CSD-93-772, Dept. EECS, Berkeley, California, 1993.

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

[15] M. Pollefeys and L Van Gool. A stratified approach to metric self-calibration. In *IEEE Conf. Computer Vision & Pattern Recognition*, pages 407–12, San Juan, Puerto Rico, June 1997.

[16] M. Pollefeys, L Van Gool, and M. Proesmans. Euclidean 3d reconstruction from image sequences with variable focal length. In B. Buxton and R. Cipolla, editors, *European Conf. Computer Vision*, pages 31–42, Cambridge, U.K., April 1996.

[17] J. G. Semple and G. T. Kneebone. *Algebraic Projective Geometry*. Oxford University Press, 1952.

[18] P. Sturm. Critical motion sequences for monocular self-calibration and uncalibrated Euclidean reconstruction. In *IEEE Conf. Computer Vision & Pattern Recognition*, Puerto Rico, 1997.

[19] P. Sturm. *Vision 3D non calibrée : contributions à la reconstruction projective et étude des mouvements critiques pour l'auto-calibrage*. Ph.D. Thesis, Institut National Polytechnique de Grenoble, December 1997.

[20] P. Sturm and B. Triggs. A factorization based algorithm for multi-image projective structure and motion. In *European Conf. Computer Vision*, pages 709–20, Cambridge, U.K., 1996. Springer-Verlag.

[21] B. Triggs. A new approach to geometric fitting. Available from http://www.inrialpes.fr/movi/people/Triggs.

[22] B. Triggs. Autocalibration and the absolute quadric. In *IEEE Conf. Computer Vision & Pattern Recognition*, Puerto Rico, 1997.

[23] B. Triggs. Autocalibration from planar scenes. Submitted to 1998 European Conference on Computer Vision, June 1998.

[24] T. Viéville and O. Faugeras. Motion analysis with a camera with unknown and possibly varying intrinsic parameters. In E. Grimson, editor, *IEEE Int. Conf. Computer Vision*, pages 750–6, Cambridge, MA, June 1995.

[25] W. Wunderlich. Rechnerische rekonstruktion eines ebenen objekts aus zwei photographien. In *Mittsilungen Geodät. Inst. TU Gras,* Folge 40 (festschrift K. Rimmer sum 70. Geburstag), pages 365–77, 1982.

[26] C. Zeller and O. Faugeras. Camera self-calibration from video sequences: the Kruppa equations revisited. Technical Report 2793, INRIA, INRIA Sophia-Antipolis, France, 1996.

[27] Z. Zhang, Q.-T. Luong, and O. Faugeras. Motion of an uncalibrated stereo rig: Self-calibration and metric reconstruction. Technical Report 2079, INRIA, Sophia-Antipolis, France, 1993.