A Simple Solution to the Six-Point Two-View Focal-Length Problem

Hongdong Li

RSISE, The Australian National University, National ICT Australia

Abstract. This paper presents a simple and practical solution to the 6-point 2-view focal-length estimation problem. Based on the *hidden-variable* technique we have derived a 15th degree polynomial in the unknown focal-length. During this course, a simple and constructive algorithm is established. To make use of multiple redundant measurements and then select the best solution, we suggest a kernel-voting scheme. The algorithm has been tested on both synthetic data and real images. Satisfactory results are obtained for both cases. For reference purpose we include our Matlab implementation in the paper, which is quite concise, consisting of 20 lines of code only. The result of this paper will make a small but useful module in many computer vision systems.

1 Introduction

This paper considers the problem of estimating a constant unknown focal-length from six corresponding points of two semi-calibrated views. By *semi-calibration* we mean that all camera intrinsic parameters but a fixed focal-length are known. This scenario is quite common (not restrictive) in daily camera use. For example, except for the case where the camera lens is allowed to zoom continuously, it is often practical to assume that its focal-length is constant across multiple views. In fact, all *other* camera intrinsic parameters (such as principal point and aspect ratio) can be considered fixed and known for a certain camera. In other words, the only user-adjustable (therefore variable) camera intrinsic parameter is the focal-length. Yet still, the focal-length is often kept constant over two successive image shoots [7][8].

It is well known that five points of two fully-calibrated views are possible to recover the essential matrix $\mathtt{E}$ between the two views. Since an essential matrix is a faithful representation of the camera motion (up to an unknown scale), namely, $\mathtt{E} = [t]_{\times}\mathtt{R}$, it therefore has five degrees of freedom. So, from five points it is possible to estimate the camera motion—this is exactly what the five-point algorithm does [11].

Now consider a semi-calibrated case where only a fixed focal-length f is unknown. For this case, it is shown that six points (in general position) are enough to estimate the camera motion as well as the unknown focal-length. This can be easily seen by the following reasoning. Compared to the fully-calibrated five-point case, the one extra point correspondence will provide one more constraint on the camera intrinsic matrix. Consequently, a single unknown focal-length, as well as the relative camera motion, can be computed from it.

The above conclusion can also be approached from the other direction. If the two camera views are uncalibrated, then seven points are the minimal requirement to compute a fundamental matrix $\mathtt{F}$. Since a fundamental matrix has seven degrees of freedom,

A. Leonardis, H. Bischof, and A. Pinz (Eds.): ECCV 2006, Part IV, LNCS 3954, pp. 200–213, 2006.

it provides two more constraints on the camera intrinsics, besides the camera motion encoded by an essential matrix. These two extra constraints are essentially the two Kruppa equations. Therefore, if the two views are only partially calibrated in that all camera intrinsics but *two* possibly different focal-lengths f and f' are known, then seven points are enough to estimate the relative camera motion and the two unknown focal-lengths [2]. Now, if we have only seven less one points, and the two focal-lengths are assumed identical, then it is possible to recover the single unknown focal-length as well as the camera motion from six corresponding points.

For the first time, Stewénius et al have proposed a concrete algorithm to solve the 6-point focal-length problem [1]. They have utilized a special mathematical tool—the Gröbner basis technique. The idea behind the Gröbner basis is to construct a complete and algebraically-closed polynomial system (an *ideal*) by adding in some newly *generated* compatible equations. By this tool they show that there are at most 15 solutions to the six-point algorithm. The Gröbner basis is a mathematically elegant technique for handling polynomial system. However, since it originates from a special mathematical field (i.e. *computational commutative algebra and algebraic geometry*), some readers may find it not fully-comfortable to follow, let alone to actually implement it and use it.

Why Six Points*?* Traditionally, the focal-length problem is solved through the fundamental matrix which itself can be computed from seven points. Moving from seven points to six points provides some benefits. The first benefit lies in its theoretic value. Compared with its non-minimal counterpart, the minimal algorithm offers a deeper theoretical understanding to the problem itself. For example, both the five-point algorithms [11] and the six point algorithm all better exploit the constraints provided by the epipolar equations and the Kruppa equations (cf. [14][15]); Secondly, effective techniques developed during the course of deriving the six-point algorithm are very useful for other similar vision problems too (e.g. [11]); Thirdly, for the task of focal-length estimation itself, it is demonstrated by experiments that the six-point algorithm sometimes offers even better performance than the seven-point algorithm; In addition, as shown in [11], six-point algorithm has less degenerate configurations than the seven-point algorithm; Moreover, when combine a minimal solver with the RANSAC scheme using six points (rather than seven) allows significant reduction in computation [5].

1.1 Main Contributions

This paper provides an alternative yet much simpler and practical solution to the 6-point focal-length problem, compared to the one originally proposed in [1].

We will show that to solve the 6-point problem there is *no* need to generate new equations. The original equations system, which includes the six epipolar conditions, one singularity condition and two Kruppa equations, already provides sufficient and algebraically-closed constraints to the problem. As a result, in the real domain $\mathbb{R}$ it is already enough to solve the six-point problem using 10 *rigidity equations*—equivalent to the above equations—without resorting to the Gröbner basis technique. For reference purposes we provide our implementation in the appendix of the paper, which is very concise and consists of 20 lines of general Matlab code only.

Paper [1] tested its algorithm mainly on noise-free simulation data. In this paper, we go beyond such an idealized scenario. We have tested the performance of our algorithm

on both synthetic and real images (with different levels of noise). We demonstrate our results by the accuracy of focal-length estimation *per se*, rather than by the errors in the reprojected fundamental matrix.

In the root-selection stage (whose purpose is to single-out the best root from multiple solutions), we propose a *kernel-voting* scheme, as an alternative to the conventionally adopted RANSAC. We show by experiments that our scheme is suitable for the particular problem context, and there is no need to wait until the reprojected fundamental matrix error is obtained.

2 Theoretic Backgrounds

Consider a camera, with constant intrinsic parameters denoted by a matrix $\mathtt{K} \in \mathbb{R}^{3\times3}$, observing a static scene. Two corresponding image points $\mathbf{m}$ and $\mathbf{m}'$ are related by a fundamental matrix $\mathtt{F} \in \mathbb{R}^{3\times3}$:

$$\mathbf{m}'^{\mathbf{T}}\mathtt{F}\mathbf{m} = 0. \tag{1}$$

A valid fundamental matrix must satisfy the following singularity condition:

$$\det(\mathtt{F}) = 0. \tag{2}$$

This is a cubic equation. Remember that the 3×3 fundamental matrix is only defined up to a scale, it therefore has 7 degrees of freedom in total. Consequently, seven corresponding points are sufficient to estimate the $\mathtt{F}$.

If the camera is fully-calibrated, then the fundamental matrix is reduced to an *essential matrix*, denoted by $\mathtt{E}$, and the relationship between them reads as:

$$\mathtt{K}^{-\mathbf{T}}\mathtt{E}\mathtt{K}^{-1} = \mathtt{F}. \tag{3}$$

Since an essential matrix $\mathtt{E}$ is a faithful representation of the relative camera motion (translation and rotation, up to a scale), it has only five degrees of freedom. Consequently, to be a valid essential matrix $\mathtt{E}$, it must further satisfy two more constraints, which are characterized by the following theorem.

Theorem-1: A real 3×3 *matrix* $\mathtt{E}$ is an essential matrix *if and only if* it satisfies the following condition

$$2\mathtt{E}\mathtt{E}^{\mathbf{T}}\mathtt{E} - \mathbf{tr}(\mathtt{E}\mathtt{E}^{\mathbf{T}})\mathtt{E} = 0. \tag{4}$$

This gives 9 equations in the elements of $\mathtt{E}$, but only two of them are algebraically independent. The above theorem, owing to many researchers (e.g, Kruppa, Demazure, Maybank, Huang, Trivedi, Faugeras, etc, just name a few, cf. [6][5][15]), is an important result in geometric vision.

For the semi-calibrated case considered here, since only one focal-length is unknown, without loss of generality we can assume the intrinsic camera matrix is: $\mathtt{K} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix}$, where f is the focal-length. Define a matrix $\mathtt{Q} = w^{-1}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & w \end{bmatrix}$, where $w = f^{-2}$. Write down the epipolar relations Eq. (1) for six points $\mathbf{m}_i$ and $\mathbf{m}'_i$,

$$\mathbf{m}_i'^{\mathbf{T}}\mathtt{F}\mathbf{m}_i = 0, \tag{5}$$

for $i = 1, \cdots, 6$. Using the six points we get a linear representation of the fundamental matrix:

$$\mathtt{F} = x\mathbf{F}_0 + y\mathbf{F}_1 + z\mathbf{F}_2, \tag{6}$$

where x, y, z are three unknown scalars to be estimated, and $\mathbf{F}_0, \mathbf{F}_1, \mathbf{F}_2$ are the bases of the null-space of the *epipolar design matrix*, which can be readily computed from the six points (cf. [5]).

Substituting this $\mathtt{F}$ into Eq.(3) and Eq.(4), we get the following equations in the unknown set $\{w, x, y, z\}$.

$$2\mathtt{F}\mathtt{Q}\mathtt{F}^{\mathbf{T}}\mathtt{Q}\mathtt{F} - \mathbf{tr}(\mathtt{F}\mathtt{Q}\mathtt{F}^{\mathbf{T}}\mathtt{Q})\mathtt{F} = 0. \tag{7}$$

This is a group of nine equations, and they provide sufficient conditions to find the unknown $\{w, x, y, z\}$ (up to an unknown scalar). If we somehow solve these equations, then the task of estimating the focal-length is accomplished. The above reasoning basically follows [1].

3 Review the Previous Algorithm

Stewénius et al proposed a clever algorithm based on the Gröbner basis technique [1]. More precisely, it is a variant of the classical Gröbner technique [4]. The key steps of their algorithm are briefly reviewed below.

Given six corresponding image points in general position, write down Eq.(7) and Eq.(2). Rearrange them in such a way that a 10×33 matrix equation $\mathtt{A}\mathbf{X} = 0$ is obtained, where $\mathtt{A}$ is a 10×33 coefficient matrix, and $\mathbf{X}$ a vector containing 33 terms of monomials of the unknowns. Now we have a polynomial system of 10 equations. This system is then ported into a finite field $\mathbb{Z}_p$ (p is a large *prime number*), and is solved using the Gröbner basis elimination procedure. This procedure is stopped when the whole system becomes an algebraically-closed *ideal generator set* of the original system. So far, a minimal solver (for $\mathbb{Z}_p$) has been built up.

The next step is to apply the same solver (i.e, the same sequence of elimination) to the original problem. One then obtains an enlarged polynomial system containing $n \times 33$ ($n > 10$) monomial terms. Finally, a generalized eigen-decomposition is employed to solve the polynomial system, for which there are 15 solutions. In order to improve numerical stability, a pivoted Gauss-Jordan elimination is used.

An important detail of the algorithm is that the arbitrary scale factor of the fundamental matrix is parameterized by setting one unknown to an arbitrary scalar. Thereby the number of unknowns is reduced by one, which simplifies the later derivation. By contrast, in this paper we avoid such scale parametrization in order to keep the homogeneity of some unknowns of the equation system. The reason will be explained later.

Limitations. The main mathematical device adopted by [1] is the Gröbner basis technique. The Gröbner basis is an elegant and powerful technique[3] [4]. Many commercial or free mathematical software packages include it as a standard module (for instance, in Maple and Mathematica etc). In many cases, to use it the user is not assumed to have specialized knowledge of it, and thus can simply apply it in a black-box manner, as also claimed by [1]. However, using a tool in a black-box manner is not always a safe way.

Whenever a program runs into trouble, it would be nicer if the user could understand its internal mechanism. Moreover, due to its special origin (*computational commutative algebra and algebraic geometry*), not every reader finds it easy to follow. Furthermore, paper [1] did not test its algorithm extensively on more realistic case. It experimented on perfect simulated data only. No result on real images was given there.

Finally, the *root-selection* procedure (i.e., single-out the best root from the possibly multiple solutions) is not addressed by paper [1], because it deals with simulated cases only, and thereby assumes the ground-truth data is available. However, in a real-world problem an efficient root-selection mechanism is necessary. It is in fact a common requirement for various minimal solvers (see for example [11] and [8]), where one often obtains multiple and maybe complex roots. The RANSAC is a good scheme to find the best solution from multiple candidates. In this paper, we propose an alternative *kernel-voting* scheme which is suitable for the particular context.

4 Our New Six-Point Algorithm

In this paper, we propose a new method for solving the six-point focal-length problem, using the *hidden variable* technique which is probably the best known technique for algebra elimination.

We claim that the recommended *hidden variable* technique is *not yet-another* specialized mathematical technique (which otherwise would be equally unfamiliar and uncomfortable to readers), but it follows very straightforward principle and procedures. It is so transparent and simple to the end-user that is almost self-explained. As will be described later, to better apply this technique to the problem, we introduce a small trick that is to keep the homogeneity of some unknowns of the equation system.

Hidden Variable Technique. The Hidden-Variable technique (also known as the *Dialytic Elimination*) is possibly one of the best known *resultant* techniques in algebraic geometry [4]. It is used to eliminate variables from a multivariate polynomial equation system. The basic idea is as follows.

Consider a system of M homogeneous polynomial equations in N variables, say, $p_i(x_1, x_2, ..., x_N) = 0$, for $i = 1, 2, ..., M$. If we treat one of the unknowns (for example, x_1) as a *parameter* (in the conventional terms, we *hide* the variable x_1), then by some simple algebra we can re-write the equation system as a matrix equation

$$\mathtt{C}(x_1)\mathbf{X} = 0,$$

where the coefficient matrix $\mathtt{C}$ will depend on the *hidden variable* x_1, and the $\mathbf{X}$ is a vector space consisting of the homogeneous monomial terms of all other N-1 variables (say, $x_2, x_3, \cdots, x_N$).

If the number of equations equals the number of monomial terms in the vector $\mathbf{X}$ (i.e. the matrix $\mathtt{C}$ is square), then one will have a *resultant equation* defined on x_1, say, $\det(C(x_1)) = 0$ *if and only if* the equation system has non-trivial solutions. By such procedure, one thus successfully eliminates N-1 variables from the equation system all at once. Solving the resulting resultant equation for x_1 and back-substituting it, one thus eventually solves the whole system.

4.1 Algorithm Derivation

Remember that Eq.(2) and Eq.(7) are the main equations we are to use. Notice that they are ten cubic equations in the four unknowns $\{w, x, y, z\}$. A careful analysis will show that within the real domain $\mathbb{R}$, Eq.(7) already implies Eq.(2). However, we would keep all these ten equations together in our derivation, and the reason will become clear soon.

Now we treat the unknown w as the hidden variable, and collect a coefficient matrix (denoted by $\mathtt{C}(w)$) with respect to the other three variables $\{x, y, z\}$. Here we do not replace one variable with an arbitrary scalar. Rather, we keep the homogeneous forms in the monomials formed by $[x, y, z]$. These are all cubic monomials which actually span a vector space:

$$\mathbf{X} = [xyz, x^2z, xy^2, xz^2, y^2z, yz^2, x^3, y^3, x^2y, z^3]^{\mathbf{T}} \tag{8}$$

To give a more close examination of the coefficient matrix $\mathtt{C}$, we list it element-wise:

	0	1	2	3	4	5	6	7	8	9
	xyz	x^2z	xy^2	xz^2	y^2z	yz^2	x^3	y^3	x^2y	z^3
0	s_0	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9
1	$[w]_{10}$	$[w]_{11}$	$[w]_{12}$	$[w]_{13}$	$[w]_{14}$	$[w]_{15}$	$[w]_{16}$	$[w]_{17}$	$[w]_{18}$	$[w]_{19}$
2	$[w]_{20}$	$[w]_{21}$	$[w]_{22}$	$[w]_{23}$	$[w]_{24}$	$[w]_{25}$	$[w]_{26}$	$[w]_{27}$	$[w]_{28}$	$[w]_{29}$
3	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$\cdots$	$\cdots$	$\cdots$	$[w]_{i-1,j}$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$\cdots$	$\cdots$	$[w]_{i,j-1}$	$[w]_{i,j}$	$[w]_{i,j+1}$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$\cdots$	$\cdots$	$\cdots$	$[w]_{i+1,j}$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
8	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
9	$[w]_{90}$	$[w]_{91}$	$[w]_{92}$	$[w]_{93}$	$[w]_{94}$	$[w]_{95}$	$[w]_{96}$	$[w]_{97}$	$[w]_{98}$	$[w]_{99}$

Here, elements in the first row are some scalars, $\mathtt{C}(i, j) = s_j$, for $i = 0$, computed from the singularity constraint Eq.(2). Elements of all other rows are quadratic in w, computed from the nine rigidity constraints Eq.(4). More precisely, it is in the form of $\mathtt{C}(i, j) = [w]_{ij} \doteq a_{ij}w^2 + b_{ij}w + c_{ij}$, for $1 \leqslant i \leqslant 9$.

As the monomial vector has been kept homogenous, the equation $\mathtt{C}(w)\mathbf{X} = 0$ will have non-trivial solutions of $\{x, y, z\}$ if and only if the determinant of the coefficient matrix vanishes. That is:

$$\boxed{\det(\mathtt{C}(w)) = 0.} \tag{9}$$

This determinant is better known as a *hidden-variable-resultant*, which is an univariate polynomial of the hidden variable, w.

By observing the elements of $\mathtt{C}$, one would expect that its determinant is an 18th degree polynomial. However, a more close inspection reveals that: it is actually a 15th degree polynomial, because terms of degree greater than 15 precisely cancel out. As a result, from a group of six points we will eventually obtain a **15th** degree polynomial in the single unknown w. More importantly, since the vector $\mathbf{X}$ is homogenous in $\{x, y, z\}$, and during the above construction we did not *generate* any extra equations besides the original ten, we are therefore safe to conclude that there are indeed at most 15 solutions to the six-point two-view focal-length problem. This result accords precisely with [1], but we achieve this via a different and more transparent approach.

Another benefit of keeping the homogeneity of $\mathbf{X}$ is that: estimating the fundamental matrix corresponding to the computed w is also made much easier than by [1]. Notice that the null-space $\mathbf{X} = \mathbf{null}(\mathtt{C}(w))$ is homogenous in $\{x, y, z\}$. Therefore, computing x, y, z is also made simple, thanks to the symmetric structure of the vector $\mathbf{X}$. As a result, the fundamental matrix $\mathtt{F}$ can be directly found using Eq. (6). By contrast, the back-substitution sequences used in [1] is more *ad hoc* and heuristic.

5 Implementation

5.1 Minimal Solver

Using the above construction, one can compute a hidden-variable resultant (i.e. a univariate polynomial equation) from every six points. Solving the hidden-variable resultant for the unknown w, one then finds the focal-length f. In general there are multiple *candidate* solutions. By *candidate* we mean that they have real positive values. To give a flavor we show below an example of such a 15th degree resultant polynomial and the corresponding real positive roots of f:

$$\begin{aligned} poly = &-9.3992319e^{-14}w^{15} - 4.7922208e^{-17}w^{14} + 8.7010404e^{-22}w^{13} \\ &+7.4700030e^{-25}w^{12} + 4.5342426e^{-29}w^{11} + 1.1095702e^{-33}w^{10} \\ &+9.3596284e^{-39}w^{9} - 9.8528519e^{-44}w^{8} - 1.3185972e^{-48}w^{7} \\ &+1.3420899e^{-53}w^{6} - 2.6216189e^{-59}w^{5} - 1.0514824e^{-64}w^{4} \\ &+5.5394855e^{-70}w^{3} - 9.1789042e^{-76}w^{2} + 6.0294511e^{-82}w - 1.2914421e^{-88} \\ f_{cand} = &\ 1/\sqrt{w} = [1536.38, 1104.52, 600.01, 593.99, 521.99, 83.74]. \end{aligned}$$

For this example we knew the ground-truth solution is $f_{true} = 600$ pixels.

There exist many (*global*) approaches to solving a univariate polynomial equation. Popular options include the *companion matrix* technique, or Sturm sequence bi-section technique [4]. The former can find *all roots* of a univariate polynomial, and the later can find *all real roots*. After solving the resultant equation, we only keep the real positive ones as the *candidate roots*, and then feed them into a second stage—root-selection.

5.2 Root Selection

From six point correspondences, one may get multiple candidate focal-lengths. Therefore a root-selection stage is required to single out a unique best root. In general, this stage is possible if only we have more than six points. In other words, it is the redundant measurements that provides extra information to resolve the multi-root ambiguity.

Paper [1] did not address the issue of root-selection, because it only deals with synthetic data with known ground-truth. RANSAC scheme is a good choice to fulfil such root-selection task. In the following we propose a kernel-voting scheme, which can be used as an alternative to the RANSAC, and which we think is suitable for the underlying problem.

6 Kernel Voting: Combining Multiple Measurements

To resolve the multiple roots ambiguity, the classical way is to use multiple measurements to eliminate those inconsistent solutions.

Given N ($N \gg 1$) groups of data, we will have a system of N simultaneous polynomial equations in one unknown. Any attempt to solve such an over-determined equation system strictly (exactly) is doomed to fail, because the inevitable noise in input will almost always make the equation system inconsistent (i.e, *co-prime*).

Alternatively, one could exploit the *least square* idea. For example, one might think of using a global cost function by summing up the square of each individual resultant equations Eq.(9) and then apply the bundle adjustment. However, experiments show that this simple idea does not work, because the summation has cancelled many of the convexities of the individual polynomials, leading to a cost function which is less likely to converge to global minima [13] [9].

RANSAC has been proposed as a successful approach to disambiguate the multiple roots problem, e.g, in *five-point relative-orientation* problem [11]. It is a good option here to resolve the ambiguity. However, in this paper, we suggest an alternative scheme based on the kernel-voting idea [10], which we believe is quite useful for certain situations.

The basic strategy is to keep the form of *minimal solver* for each individual data group (of six points), and use a *voting* scheme to choose the *best root* afterward. By the *best root* we mean that it is agreed by the majority of the the input measurements. This technique is therefore also immune to outliers.

6.1 Kernel Voting

The purpose of kernel-voting ([10]) is to single out the best real root from multiple candidate solutions. We fulfil this by a "soft" voting scheme, where the *votes* are the candidate roots that each polynomial Eq.(9) produces.

Because Eq.(9) is a high-degree polynomial, it is very sensitive to noise. A small perturbation in the input point coordinates may cause large changes in the polynomial coefficients. And this may significantly distort the resultant equation, as well as its roots. However, By experimentations we found: although noise affects the high-order basic equations *individually*, the obtained roots mostly surround the genuine root (this is also because a polynomial is a continuous function). The statistical distribution of all roots computed from the multiple measurements displays a peak shape. So long as a sufficiently large number of measurements, an asymptotically correct root will be eventually found. That is, the position that receives the maximal numbers of votes will eventually win.

In spirit, our voting scheme is similar to the Hough transform. However, their operations are differences. In the Hough transform the voting space is tessellated into discrete cells, while the voting space of ours is the continuous real axis. In addition, since we only receive votes at rather sparse and isolated positions (of the real roots), our search can be performed more efficiently. In order to smooth the voting space, we introduce a kernel density estimator (KDE) to estimate the distribution of the candidate roots. Then the peak position, corresponding to the (globally) maximal (peak) probability, is identified as the output of the best root. In this sense, it works like a Maximal-Likelihood Decision-Maker. Compared with the RANSAC, our kernel-voting scheme turns to make a *collective* decision, rather than depending on one *individual* decision.

Given multiple independent observations of a random variable, the KDE at point x is defined as

$$\hat{f}_h(x) = \sum_{i=1}^{n} \frac{\mathcal{K}(x_i - x)}{h} \tag{10}$$

where $\mathcal{K}()$ denotes the kernel function, and h the bandwidth. Here we choose a Gaussian Kernel with fixed bandwidth, and simply set the bandwidth as the estimation precision that we expect (e.g, 0.5%–1% of the focal-length).

7 Experiments

Thanks to the simplicity of the *hidden variable* technique, we implemented our six-point algorithm economically. The program language used is Matlab (with Symbolic-Math toolbox, which is essentially a subset of Maple). For reference purposes we include our Matlab program in the appendix of the paper. The central part of the program consists of 20 lines of code only, most being general (Matlab and Maple) functions. No hidden code is used. Only for demonstration purpose, a Maple function `solve` is applied in one step, which itself is indeed a long implementation. However, since that step is only used to solve a univariate equation, the reader can change it to any suitable solver.

We test our algorithm on both synthetic data (with various levels of noise and outliers) and real images. Some results are reported below.

7.1 Test on Synthetic Data

To resemble the real case, the size of synthetic image is 512×512. We tested different values for f, but found that they do not affect the final accuracy. So, in what follows we always use a ground-truth focal-length of 600 pixels. The camera motions between two views are drawn randomly. No special attention has been paid to avoid the *degenerate* motions (for focal-length estimation [7]). Gaussian noise was introduced to the raw image coordinates. It is noteworthy that the Hartley's normalization ([5]) is *not* essential for our six-point algorithm.

Our first experiment aimed at testing the focal-length estimation accuracy versus different image noises. From six points our algorithm is already able to output real focal length. However, in order to obtain a statistically robust estimation, fifty feature points were used to extract three 9-dimensional null-space vectors. After applying the procedures, we choose the best root as the nearest one to the ground-truth, and repeat this procedure 100 times. The following curves (fig-1) show the distribution of relative errors (percentage) in focal-length under different levels of noise. Our second experiment was used to test the performance of the root-selection based on the proposed kernel-voting scheme. From 50 point correspondences, we randomly drew 50 six-point data-groups, and apply our six-point algorithm to them. After performing a kernel-voting on all the obtained candidate focal-lengths, we plot the curves of root distribution and pick up the peak position. Some example curves are shown in figure-2. These curves show that the estimated focal-lengths are quite accurate.

We also test the cases where there are outliers in inputs and where there are errors in some of the camera intrinsics. From the voting curves shown in fig-3 we see that

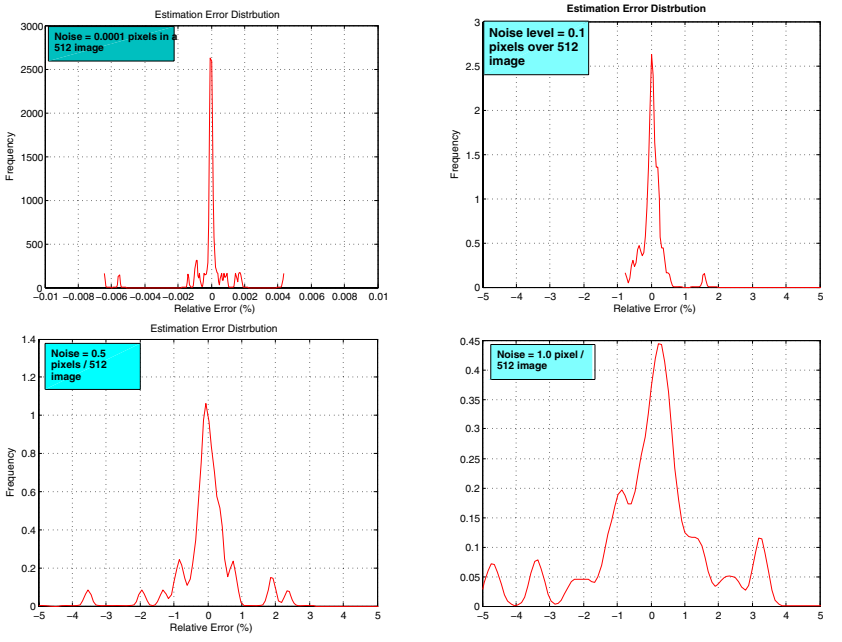


Fig. 1. Distribution of relative errors in focal-length estimation, for noise levels at (a) 0.0001 pixels, (b) 0.1 pixels and(c) 0.5 pixels (d) 1.0 pixels in a 512 size synthetic image. Note that even when the noise level is at 1.0 pixels, the relative errors are mainly less than 5%.

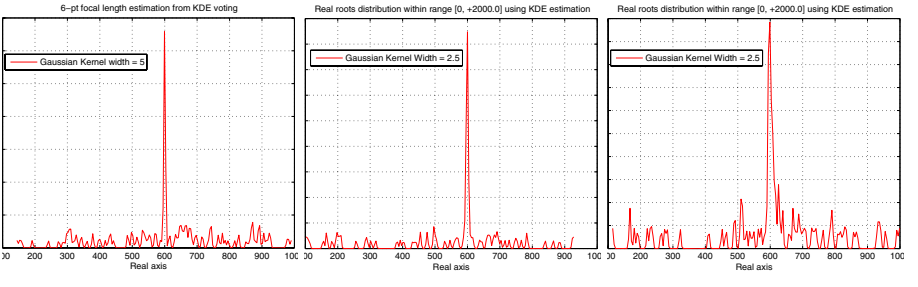


Fig. 2. Kernel-voting results, for noise level at (a)0.0001 pixels, (b)0.1 pixels and (c)1.0 pixels in a 512 size image. From the peak position we get the focal-length estimation $f \approx 600$.

the proposed method is robust to outliers, and not sensitive to the errors in some intrinsics. To quantitatively evaluate the estimation accuracy, we repeat the experiment 100 times, and plot the error-bar curve (mean value and standard deviation versus noise) in fig-4. Remember that the camera motions are drawn randomly (i.e. we did not intentionally avoid the degenerate motion). We also conducted experiments for comparing the numerical performances between our algorithm and ([1]), but no significant difference was found. This makes sense as both algorithms use essentially the same formulation.

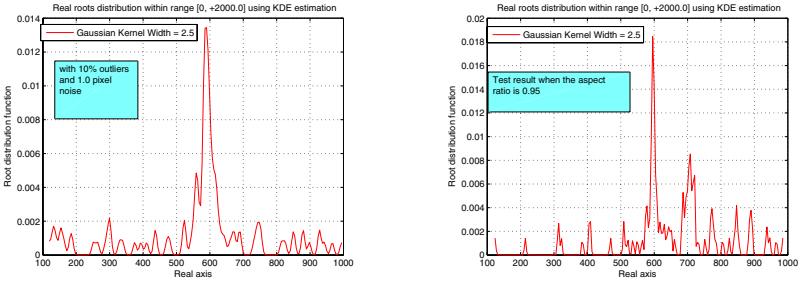


Fig. 3. (a)With 10% outliers (b)With some errors in the aspect ratio estimation: the true aspect-ratio is 0.95 but mis-use 1.00

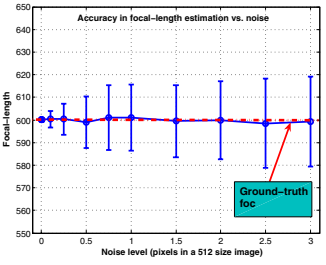
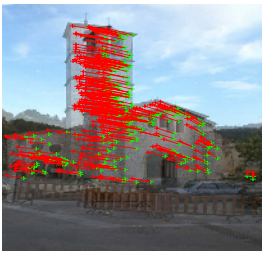


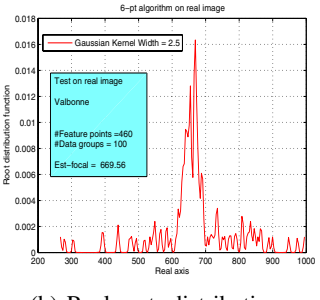
Fig. 4. Error bars (mean value and standard deviations) of focal-length estimation under different levels of noise

7.2 Test on Real Images

We have tested our new six-point algorithm on some real images. For example, we test the Valbonne sequences shown in fig-5(a). The two input images are partially calibrated using the calibration information provided by other authors[9]: $[\alpha, u_0, v_0] = [1.0, 258.8, 383.2]$. Then apply our algorithm, we get the following root distribution



(a) Two Valbonne images and some corresponding points.



(b) Real roots distribution.

Fig. 5. Test on real images: Valbonne sequences

curve shown in figure-5(b), from which we can read the focal-length is about 670 pixels, which is close to the estimation of 699 pixels given by [9]. We also test some other standard image sequences. We find that as long as the two cameras are not in degenerate configurations ([8]) the estimated focal-length is close to the ground-truth data (obtained from other calibration procedure).

8 Discussion

Even when there are more than six points available in image, there are still advantages of using the *six-point* algorithm. Indeed, it is a good strategy of keep using such a *minimal-solver* even when extra data are available. The reason is explained in [11] and [1], showing that minimal-solver often offers better performance than the non-minimal ones. At first sight this is a bit surprise. However, a careful analysis will reveal the reason. That is, the minimal solver has better exploited all available inherent constraints of the problem (both linear and nonlinear), while many other conventional algorithms (e.g, 8pt algorithm) only use the extra measurements to get a better linear null-space estimation [5].

Our algorithm will fail when the *degenerate cases*(for focal-length estimation, cf. [8]) is met, for example, when the two optical axes intersect at equal distances, or when the camera underwent a pure translation. As this is a general difficulty for any focal-length algorithm, we do not intend to overcome it here. However, it is our conjecture that because the six-point algorithm has better exploited the nonlinear constraints it might have better conditioning near some degenerate configurations (including the critical surfaces and singular motions). To justify this, more critical experiments and theoretical analysis are yet to be done. The author believes that how to mitigate the degenerate surfaces problem in motion-and-structure computation is a topic worth researching.

9 Conclusions

We have provided a practical algorithm to solve the six-point focal-length problem. The most appealing feature is its simplicity and transparency. Besides its theoretical contribution, we hope the six-point algorithm will make a small and useful module in many vision systems.

We believe that the proposed algorithm is not an individual success of the powerful hidden-variable technique. It can have wider applications in similar problems, for example, five-point relative-orientation and three-point absolute-orientation etc. These can be future work.

Acknowledgments. NICTA is funded through the Australian Government's Backing Australia's Ability Initiative, in part through the ARC. The author wishes to thank Richard (H) for inspiration, guidance and invaluable support, to Fredrik (K) for many illuminating discussions, to Fred (S) for introducing me to UAG [4] and for the mug, and to the three anonymous reviewers for their suggestions that much improve the paper.

References

1. H.Stewénius, D.Nistér, F.Kahl, F.Schaffalitzky, A minimal solution for relative pose with unknown focal length, in Proc. IEEE-CVPR-2005, 2005.
2. R.I.Hartley, Estimation of Relative Camera Positions for Uncalibrated Cameras, In Proc.2nd ECCV, 1992.
3. S.Petitjean, Algebraic geometry and computer vision: Polynomial systems, real and complex roots, Journal of Mathematical Imaging and Vision,10:191-220,1999.
4. D.Cox, J.Little and D.O'shea, *Using Algebraic Geometry*, 2nd Edition, Springer, 2005.
5. R.Hartley, A.Zisserman, Mutiview Geometry in computer vision, 2nd Edition, Cambridge University Press, 2004.
6. O.Faugeras, Q.Luong, *The geometry of multiple images*, The MIT Press, 2001.
7. P.Sturm, On focal-length calibration from two views, In Proc.CVPR-2001, December, 2001.
8. P.Sturm,et al, Focal length calibration from two views:method and alaysis of singular cases, Computer Vision and Image Understanding,Vol 99, No.1,2005.
9. A.Fusiello, et al, Globally convergent autocalibration using interval analysis, IEEE T-PAMI 26(12), 2004.
10. Hongdong Li, Richard Hartley, A Non-iterative Method for Correcting Lens Distortion from Nine-Point Correspondences, In Proc. OmniVision'05, ICCV-workshop,2005.
11. D.Nistér, An efficient solution to the five-point relative pose problem, in Proc. IEEE CVPR-2003,Vol-2, pp. 195-202, 2003.
12. R.Hartley, F.Schaffalitzky, L-inf minimization in geometric reconstruction problems, In Proc. CVPR-2004, 2004.
13. F.Kahl, D.Henrion, Globally optimal estimes for geometric reconstruction problems, In Proc. ICCV-2005, Beijing, 2005.
14. O. Faugeras and S. Maybank, Motion from Point Matches:Multiplicity of Solutions, IJCV,vol.4,pp.225-246,1990.
15. A. Heyden, and G. Sparr, Reconstruction from Calibrated Cameras A New Proof of the Kruppa-Demazure Theorem, J. Math. Imag. and Vis., vol.10, pp.1-20, 1999.

Appendix

Program 1 . The six-point focal-length algorithm

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% This is a simple 6-pt focal-length algorithm.          %%%
%% Use Matlab-7.0(6.5)with SymbolicMath Toolbox.          %%%
%% The "Matches" is a 6x4 matrix containing six points.   %%%
%% For example,                                           %%%
Matches = [ 93.3053,   59.9312, -420.3770, -773.9141;
          -141.9589,  -50.1980, -386.7602, -471.0662;
          -174.0883, -157.0080, -489.9528, -259.9091;
          -57.6271 , -12.2055 , -394.5345, -466.4747;
          -115.7769,  154.4320, -172.2640, -461.6882;
           134.6858,   -4.0822, -575.1835, -855.5145]
%% For this example the ground truth is foc = 600.  %%%%%%%%%
%% Output: all computed focal-lengths in foc. %%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function foc = SixPtFocal(Matches)

  syms F f x y z w equ Res Q C
  Q = [1, 0 ,0 ; 0 ,1 ,0; 0, 0, w];
  q  =  [ Matches(:,1), Matches(:,2)] ;
  qp =  [ Matches(:,3), Matches(:,4)] ;
  M = [qp(:,1).*q(:,1), qp(:,1).*q(:,2), qp(:,1), ...
       qp(:,2).*q(:,1), qp(:,2).*q(:,2), qp(:,2), ...
        q(:,1), q(:,2), ones(6,1)] ;

  N = null(M) %%% compute the null-space
  f = x*N(:,1) + y*N(:,2) + z*N(:,3); %% form the FM
  F = transpose(reshape(f,3,3));
  FT =transpose(F);
  equ(1)    = det(F);
  equ(2:10) = expand(2*F*Q*FT*Q*F-trace(F*Q*FT*Q)*F);

  for i =1:10
   %Note:Be careful with MATLAB delimiter for string, 'or'?
    equ(i) = maple(`collect`,equ(i),`[x,y,z]`,`distributed`);
    for j =1:10
       oper = maple(`op`, j, equ(i)) ;
       C(i,j) = maple(`op`,1,oper);
    end
  end
  disp(`Compute Det(C),need a while,please wait,,,`);
  Res = maple(`evalf`, det(C))%%Hidden-variable resultant
  foc = 1.0./sqrt(double([solve( Res)]))
  disp(`Ground-truth focal-length = 600.0000`);
```