Numerically Stable Optimization of Polynomial Solvers for Minimal Problems

Yubin Kuang and Kalle Åström

Centre for Mathematical Sciences
Lund University, Sweden
{yubin,kalle}@maths.lth.se

Abstract. Numerous geometric problems in computer vision involve the solution of systems of polynomial equations. This is particularly true for so called minimal problems, but also for finding stationary points for overdetermined problems. The state-of-the-art is based on the use of numerical linear algebra on the large but sparse coefficient matrix that represents the original equations multiplied with a set of monomials. The key observation in this paper is that the speed and numerical stability of the solver depends heavily on (i) what multiplication monomials are used and (ii) the set of so called permissible monomials from which numerical linear algebra routines choose the basis of a certain quotient ring. In the paper we show that optimizing with respect to these two factors can give both significant improvements to numerical stability as compared to the state of the art, as well as highly compact solvers, while still retaining numerical stability. The methods are validated on several minimal problems that have previously been shown to be challenging with improvement over the current state of the art.

1 Introduction

Many problems in computer vision can be rephrased as several polynomials in several unknowns. This is particularly true for so called minimal structure and motion problems. Solutions to minimal structure and motion problems are essential for RANSAC algorithms to find inliers in noisy data [13, 24, 25]. For such applications one needs to efficiently solve a large number of minimal structure and motion problems in order to find the best set of inliers. Once enough inliers have been found it is common to use non-linear optimization *e.g.* to find the best fit to all data in a least squares sense. Here fast and numerical stable solutions for the polynomials systems of the minimal problems are crucial for generate initial parameters for the non-linear optimization. Another area of recent interest is global optimization used *e.g.* for optimal triangulation, resectioning and fundamental matrix estimation. While global optimization is promising, it is a difficult pursuit and various techniques has been tried, *e.g.* branch and bound [1], L_∞-norm methods [16] and methods using linear matrix inequalities (LMIs) [17]. An alternative way to find the global optimum is to calculate stationary points directly, usually by solving some polynomial equation system [14, 23]. So far, this has been an approach of limited applicability since calculation of stationary points is numerically difficult for larger problems. A third area are methods for finding the optimal set of inliers [11, 12, 21], which are based on solving certain geometrical problems, which involve solving systems of polynomial equations.

A. Fitzgibbon et al. (Eds.): ECCV 2012, Part III, LNCS 7574, pp. 100–113, 2012.

Recent applications of polynomial techniques in computer vision include solving for fundamental and essential matrices with radial distortion [19], panoramic stitching [2] and pose with unknown focal length [3]. To solve such geometric problems by solving systems of polynomials in several unknowns is often not trivial. First of all one has model the problem and choose parameters in order to get as simple equations as possible. Then efficient and numerical stable techniques are applied to solve the resulting system of equations.

Related Works. Traditionally, researchers have hand-coded elimination schemes in order to solve systems of polynomial equations. Recently, however, new techniques based on algebraic geometry and numerical linear algebra have been used to find all solutions, *cf*. [22]. In [7] several techniques for improving the numerical stability by allowing a larger set of so called permissible monomials from which to choose the basis of $\mathbb{C}[\mathbf{x}]/I$. By clever use of *e.g*. QR factorization with column-pivoting the numerical stability can be improved by several orders of magnitude. In [18] a system for automatic generation of minimal solvers is developed. The script first calls upon algebraic geometry packages to determine the number of solutions and to determine a basis for $\mathbb{C}[\mathbf{x}]/I$. Then repeated numerical tries with increasingly number of equations are tried until a large enough set is found. Then the equations are trimmed down automatically to give a compact set of equations and monomials. Finally these are hardcoded in a solver. Similar techniques for automatic trimming of equations has been developed in [20], where equations (rows) and monomials (columns) are removed if they have no numerical effects on the construction of the action matrix. Another recent work [4] focuses on speed improvement for polynomial solvers with techniques that avoid direct action matrix eigenvalue computation.

In this paper, we focus on optimizing polynomial solvers with respect to their size as well as numerical accuracy. The goal is to generate a solver that have better numerical accuracy with as compact set of equations as possible. This is unlike [18] and [20], where the main goals are to generate compact solvers. The common criterion is to remove equations and monomials that do not affect the overall solvability of the solvers. While such compact solvers are generally faster, they tend to have inferior numerical accuracy. We propose to use the numerical accuracy of the solvers as the criterion, and optimize the set of equations in the template, as well as the permissible set. We apply local search on this combinatorial problem and generate solvers for minimal problems with better numerical accuracy and smaller template.

2 Action Matrix Method

In this section we review some of the classical theory of multivariate polynomials. We consider the following problem:

Problem 1. *Given a set of m polynomials $f_i(\mathbf{x})$ in s variables $\mathbf{x} = (x_1, \ldots, x_s)$, determine the complete set of solutions to*

$$f_1(\mathbf{x}) = 0, \ldots, f_m(\mathbf{x}) = 0. \tag{1}$$

We denote by V the zero set of (1). In general V need not be finite, but in this paper we will only consider zero dimensional V, *i.e*. V is a point set.

The general field of study of multivariate polynomials is algebraic geometry. See [10] and [9] for an introduction to the field. In the language of algebraic geometry, V is an affine algebraic variety and the polynomials f_i generate an *ideal* $I = \{g \in \mathbb{C}[\mathbf{x}] : g = \Sigma_i h_i(\mathbf{x}) f_i(\mathbf{x})\}$, where $h_i \in \mathbb{C}[\mathbf{x}]$ are any polynomials and $\mathbb{C}[\mathbf{x}]$ denotes the set of all polynomials in $\mathbf{x}$ over the complex numbers.

The motivation for studying the ideal I is that it is a generalization of the set of equations (1). A point $\mathbf{x}$ is a zero of (1) iff it is a zero of I. Being even more general, we could ask for the complete set of polynomials vanishing on V. If I is equal to this set, then I is called a radical ideal.

We say that two polynomials f, g are equivalent modulo I iff $f - g \in I$ and denote this by $f \sim g$. With this definition we get the quotient space $\mathbb{C}[\mathbf{x}]/I$ of all equivalence classes modulo I. Further, we let $[\cdot]$ denote the natural projection $\mathbb{C}[\mathbf{x}] \to \mathbb{C}[\mathbf{x}]/I$, *i.e.* by $[f_i]$ we mean the set $\{g_j : f_i - g_j \in I\}$ of polynomials equivalent to f_i modulo I.

A related structure is $\mathbb{C}[V]$, the set of equivalence classes of polynomial functions on V. We say that a function F is polynomial on V if there is a polynomial f such that $F(\mathbf{x}) = f(\mathbf{x})$ for $\mathbf{x} \in V$ and equivalence here means equality on V. If two polynomials are equivalent modulo I, then they are obviously also equal on V. If I is radical, then conversely two polynomials which are equal on V must also be equivalent modulo I. This means that for radical ideals, $\mathbb{C}[\mathbf{x}]/I$ and $\mathbb{C}[V]$ are isomorphic. Now, if V is a point set, then any function on V can be identified with a $|V|$-dimensional vector and since the unisolvence theorem for polynomials guarantees that any function on a discrete set of points can be interpolated exactly by a polynomial, we get that $\mathbb{C}[V]$ is isomorphic to $\mathbb{C}^r$, where $r = |V|$.

3 Revisit Basis Selection Technique

In this section, we review in detail the basis selection method introduced in [6]. We further discuss the key components of this method and motivate why potential improvement can be achieved.

The goal is to construct the action matrix for the linear mapping $T_{x_k} : p(x) \mapsto x_k p(x)$ in the r-dimensional space $\mathbb{C}[\mathbf{x}]/I$ in numerically stable manner. Specifically, given the equation system, for a linear basis of monomials $\mathcal{B} = \{\mathbf{x}^{\alpha_1}, \ldots, \mathbf{x}^{\alpha_r}\}$ for $\mathbb{C}[\mathbf{x}]/I$ and the action variable x_k, we want to express the image $x_k \mathbf{x}^{\alpha_i}$ ($\forall \mathbf{x}^{\alpha_i} \in \mathcal{B}$) in terms of the basis monomials, *i.e.* $x_k \mathbf{x}^{\alpha_i} \sim \sum_j A_{ij} \mathbf{x}^{\alpha_j}$. The matrix $A = \{A_{ij}\}^T$ is called the *action matrix*. In previous methods [7, 18], the so-called single elimination template is applied to enhance numerical stability. It starts by multiplying the equations (1) by a set of monomials producing an equivalent (but larger) set of equations. There is no general criterion how to choose such set of monomials, which are usually selected as the minimal set of monomials that make the problem solvable or numerically stable. We will discuss later that the choices of multiplication monomials affect the overall numerical accuracy the solvers. By stacking the coefficients of the new equations in an expanded coefficient matrix $\mathbf{C}_{\text{exp}}$, we have

$$\mathbf{C}_{\text{exp}} \mathbf{X}_{\text{exp}} = 0. \tag{2}$$

This expanded coefficient matrix is usually called *elimination template*.

Using the notation in [7], we partition the set of all monomials $\mathcal{M}$ occurring in the expanded set of equations as $\mathcal{M} = \mathcal{E} \bigcup \mathcal{R} \bigcup \mathcal{P}$. We use $\mathcal{P}$ ($\mathcal{P}$ for permissible) to denote the set (or a subset) of monomials that stay in $\mathcal{M}$ after the multiplication of the action variable. And we denote the reducible set $x_k \mathbf{x}^{\alpha_i} \notin \mathcal{P}$ for $\mathbf{x}^{\alpha_i} \in \mathcal{P}$ as $\mathcal{R}$. The monomials $\mathcal{E}$ ($\mathcal{E}$ for excessive) are simply the monomials which are neither in $\mathcal{R}$ nor in $\mathcal{P}$. We then order them so that $\mathcal{E} > \mathcal{R} > \mathcal{P}$ holds for all monomials in their respective sets. After applying the corresponding reordering of the columns of $\mathbf{C}_{\text{exp}}$, we yield

$$\begin{bmatrix} \mathbf{C}_{\mathcal{E}} & \mathbf{C}_{\mathcal{R}} & \mathbf{C}_{\mathcal{P}} \end{bmatrix} \begin{bmatrix} \mathbf{X}_{\mathcal{E}} \\ \mathbf{X}_{\mathcal{R}} \\ \mathbf{X}_{\mathcal{P}} \end{bmatrix} = 0. \tag{3}$$

Giving a linear system as in (3), we know that if we choose from $\mathcal{P}$ a linear basis $\mathcal{B} = \{\mathbf{x}^{\alpha_1}, \ldots, \mathbf{x}^{\alpha_r}\}$, we are able to use numerical linear algebra to construct corresponding action matrix. Traditionally, the Gröbner bases are used to generate $\mathcal{B}$ for some ordering on the monomials (*e.g. grevlex* [18]). In this case, the permissible set is the same as basis set. The key observation of [6] is that by allowing adaptive selection of $\mathcal{B}$ from a large permissible set ($|\mathcal{P}| \geq r$), one can improve the numerical accuracy greatly in most cases. To achieve this, we first eliminate the $\mathcal{E}$-monomials, since they are not in the basis and do not need to be reduced. By making $\mathbf{C}_{\text{exp}}$ triangular through either LU or QR factorization, we have

$$\begin{bmatrix} \mathbf{U}_{\mathcal{E}_1} & \mathbf{C}_{\mathcal{R}_1} & \mathbf{C}_{\mathcal{P}_1} \\ 0 & \mathbf{U}_{\mathcal{R}_2} & \mathbf{C}_{\mathcal{P}_2} \\ 0 & 0 & \mathbf{C}_{\mathcal{P}_3} \end{bmatrix} \begin{bmatrix} \mathbf{X}_{\mathcal{E}} \\ \mathbf{X}_{\mathcal{R}} \\ \mathbf{X}_{\mathcal{P}} \end{bmatrix} = 0, \tag{4}$$

where $\mathbf{U}_{\mathcal{E}_1}$ and $\mathbf{U}_{\mathcal{R}_2}$ are upper triangular. The top rows of the coefficient matrix involving the excessives can be removed,

$$\begin{bmatrix} \mathbf{U}_{\mathcal{R}_2} & \mathbf{C}_{\mathcal{P}_2} \\ 0 & \mathbf{C}_{\mathcal{P}_3} \end{bmatrix} \begin{bmatrix} \mathbf{X}_{\mathcal{R}} \\ \mathbf{X}_{\mathcal{P}} \end{bmatrix} = 0. \tag{5}$$

We can further reduce $\mathbf{C}_{\mathcal{P}_3}$ into upper triangular matrix. In [6], the column-pivoting QR is utilized to try to minimize the condition number of the first $|\mathcal{P}| - r$ columns of $\mathbf{C}_{\mathcal{P}_3} \Pi$, where Π is a permutation matrix. This will enhance the overall numerical stability of the extraction of action matrix. The basis is thus selected as the last r monomials after the reordering of Π i.e. $\begin{bmatrix} \mathbf{X}_{\mathcal{P}'} & \mathbf{X}_{\mathcal{B}} \end{bmatrix}^t$. This gives us

$$\begin{bmatrix} \mathbf{U}_{\mathcal{R}_2} & \mathbf{C}_{\mathcal{P}_4} & \mathbf{C}_{\mathcal{B}_1} \\ \mathbf{0} & \mathbf{U}_{\mathcal{P}_3} & \mathbf{C}_{\mathcal{B}_2} \end{bmatrix} \begin{bmatrix} \mathbf{X}_{\mathcal{R}} \\ \mathbf{X}_{\mathcal{P}'} \\ \mathbf{X}_{\mathcal{B}} \end{bmatrix} = 0. \tag{6}$$

From this, we can express monomials in $\mathcal{R}$ and $\mathcal{P}'$ as linear combination of $\mathcal{B}$:

$$\begin{bmatrix} \mathbf{X}_{\mathcal{R}} \\ \mathbf{X}_{\mathcal{P}'} \end{bmatrix} = - \begin{bmatrix} \mathbf{U}_{\mathcal{R}_2} & \mathbf{C}_{\mathcal{P}_4} \\ \mathbf{0} & \mathbf{U}_{\mathcal{P}_3} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{C}_{\mathcal{B}_1} \\ \mathbf{C}_{\mathcal{B}_2} \end{bmatrix} \mathbf{X}_{\mathcal{B}}. \tag{7}$$

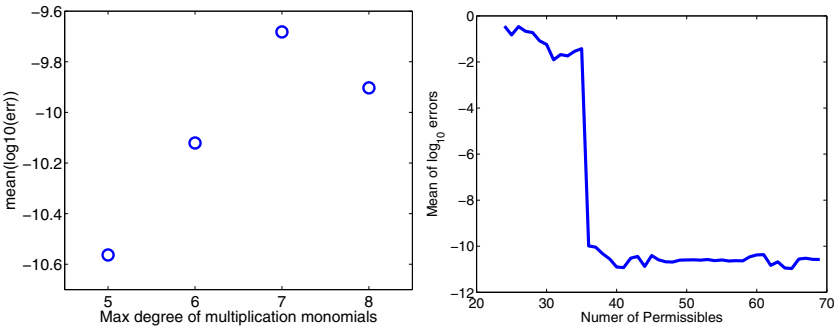

Fig. 1. Numerical accuracy (mean of log_{10}(errors)) of the solver in [8] for uncalibrated camera with radial distortion. Left: With multiplication monomials of degree 5 (393×390), 6 (680×595), 7 (1096×865) and 8 (1675×1212). The numbers in the parenthesis are the sizes of corresponding C_{exp}. Right: Vary the number of permissible monomials. Note the minimal size of the permissible set is 24 which is the number of solutions.

Since by construction $x_k\mathbf{x}^{\alpha_i} \in \mathcal{R}\bigcup\mathcal{P}'\bigcup\mathcal{B}$, we can construct the action matrix from the linear mapping in (7) by finding corresponding indices of $x_k\mathbf{x}^{\alpha_i}$ ($\forall\mathbf{x}^{\alpha_i} \in \mathcal{B}$).

This method above gives generally good performance concerning speed and numerical accuracy for most minimal problems in computer vision [7]. However, there are several parameters that are chosen manually for different problems (1) the set of multiplication monomials and (2) the choices of the permissible set. For different problem, in [6, 7], the general guideline is to use as few equations as possible (this is usually done by fine tuning the set of equations and see if the solver is still numerical stable). For the permissible set, it was suggested that one should choose it as large as possible, since this in theory should provide better chance getting stable partial-pivoting QR. One may ask, will we gain more numerical stabilities by adding more equations or using a smaller set of permissible monomials?

We present here an example to show the effects of such factors on current state of the art polynomial solvers. The problem we study is the estimation of fundamental matrix for uncalibrated cameras with radial distortion [8], which will be discussed in more details in Section 5.1. In [8], the original equations are multiplied with set of monomials so that the highest degree of resulting equations is 5. To shed light on the effects of more equations, we allow the highest degree to be 8. We note that adding equations by simply increased degrees will actually hurt the overall numerical accuracy (Figure 1, Left). On the other hand, we also vary the size of permissible set used in the solver, where we choose the last few monomials in *grevlex* order similar to [8]. We can see in Figure 1 (Right), by reducing the size of permissible set in this way, the solver retains numerical stability for permissible set of large size, while lose its accuracy for smaller permissible set (in this case when size is smaller than 36). We will show in the next few sections that one can actually gain numerical accuracy by adding equation if we carefully choose the multiplication monomials. Similarly, we can also achieve similar or even improved accuracy with smaller permissible set.

4 Optimizing Size and Accuracy for Solvers

In this section, we describe our method for optimizing numerical accuracy of polynomial solvers. We focus on improving the numerical accuracy of the basis selection method in [6] with respect to the permissible set and the equations of the template.

4.1 Permissible Selection

As discussed in 3, the permissible set is the set of monomials where the optimal basis can be selected from. In [6], it is suggested to choose the permissible set as large as possible and it is usually chosen as last few monomials (with *grevlex* order). In general, larger permissible set will gives better accuracy since it gives high freedom in choosing the optimal basis. However, for some problems, it is possible to obtain slightly better numerical accuracy with smaller set of permissible as indicated by the local minima in Figure 1 (Right) . However, it is not clear how the size of permissible set will affect the overall numerical accuracy of the final solvers. On the other hand, it should also be noted that the larger the permissible set is, we need in theory more equations for the template to be solvable. This is related to the fact that we need at least $m = |\mathcal{P}|+|\mathcal{R}|-r$ equations to yield the system in (6). Therefore, to generate slimmer solvers, one step to go is to select a compact set of permissibles while keeping the solvers numerically stable.

Choosing the optimal permissible set of size K is a combinatorial problem, and it is difficult to fully access the optimality of such sets with respect to numerical accuracy. Our approach here is to start with a large set of permissible. We then run the solvers as in [6] with basis selection for random problem examples. We collect for each run the basis selected for the specific problem in $\mathbf{B} = \{\mathcal{B}_1, \ldots, \mathcal{B}_i, \ldots, \mathcal{B}_N\}$, where N is number of random runs. In this way, we have information of what monomials are selected as basis. Typically, in our experiments, we find no unique optimal set of monomials among the permissibles, which are selected by all the random problems. One criterion for selecting a smaller permissible set of size K is to choose the most selected K monomials. The drawback of this is that such monomials as a set might have never been selected as basis in any random runs. To avoid such cases, we formulate this as a optimization problem. Essentially, a subset $\mathcal{P}^*$ of $\mathcal{P}$ would retain numerical accuracy if it is a superset of as many as basis sets as possible. Therefore, for optimal $\mathcal{P}^*$ of size K, we have the following:

Problem 2. *Given a set $\mathcal{P}$ and $\mathbf{B} = \{\mathcal{B}_1, \ldots, \mathcal{B}_i, \ldots, \mathcal{B}_N\}$ where $B_i \subseteq \mathcal{P}$ for $i = \{1, \ldots N\}$, find $\mathcal{P}^* \subseteq \mathcal{P}$,*

$$\begin{aligned} &\max_{\mathcal{P}^*} \sum_{i=1}^{N} \delta(\mathcal{B}_i \subseteq \mathcal{P}^*), \\ &s.t.\ \ |\mathcal{P}^*| = K, \end{aligned} \tag{8}$$

where $\delta(.)$ is 1 for true boolean expression and 0 otherwise.

This is a hard problem itself. However, since $|\mathcal{P}|$ is generally small for most minimal problems, we can solve it with branch and bound in reasonable time.

4.2 Equation Removal

In all previous solvers applying single elimination, the equations in the template are expanded with redundancy. This will both affect the speed and the overall numerical accuracy of the solvers. This is because all previous methods involve steps of generating upper triangular form of partial or full template, with LU or QR factorization or Gauss-Jordan elimination. For large template with many equations, this step is slow and might be numerically unstable. For [6–8], the multiplication monomials are manually tuned for different minimal problems. Both [18] and [20] investigated the possibility of automatic removing equations while keeping the solvers numerical stable. They showed that one can remove equations along with corresponding monomials without losing numerical stability and even gain better numerical performance with single precision for certain minimal problem.

In additional to numerical stability, we here try to optimize the numerical accuracy by automatically removing equations. To achieve this, we apply a simple greedy search on what equations to remove while keeping the numerical accuracy of solvers as good as possible. Given a redundant set of equations, we try to find the best equation to remove and then iterate until no equations can be removed *i.e.* when removing any of the equations makes the template unsolvable. Specifically, we first remove a candidate equation from the original template. We then measure the mean log_{10}-errors of the resulting template on a random set of problem examples (as training data). After running through all equations, we remove the equation without which gives the lowest errors. Generally, we can use any polynomial solver to evaluate the errors at each removal step. We have chosen the solver with basis selection based on column pivoting [8] since it generally gives better numerical accuracy.

By exploiting the fact that the excessive monomials do not contribute to the construction of the action matrix, we can remove certain equations without any local search and maintain the numerically stability of template. Specifically, when expanding the equations, there can be some excessive monomials that appear only in one of the equations. We call these excessive monomials as *singular excessive monomials*. For the first elimination in (4), if an equation contains singular monomials, removing it will not hurt the overall numerical condition of the LU or QR with proper pivoting. Therefore, all equations containing singular excessive can be safely removed. One can always apply this trimming step with each local search step to speed up the removal process.

It might be also good to start with more equations which might introduce better numerical conditioning for the first elimination in (4). We then rely on our equation removal step to select equations to improve the overall numerical accuracy. For monomial removal, we have not exploited the numerical routines to remove columns as in [20]. Here, we simply remove columns that become all zeros after certain equations are removed.

4.3 Optimization Scheme

Given the tools introduced in the previous sections, we should be able to optimize the polynomial solvers by finding the best combination of permissible set and equations for a specific problem. This is a difficult combinatorial problem. Therefore, as a first

attempt, we here first optimize the permissible set by fixing the set of equations. We select the optimal permissible sets of different sizes on the original template (no equation removal). Then we apply greedy search to remove equations with these optimal permissible sets. All these training are done with a fixed set of random problem examples. By investigating the error distributions of all these solvers, the size of template (number of equations, number of monomials), we will be able to choose these optimized solvers with trade-off in accuracy and speed. Generally, we have the following steps for optimizing polynomial solver for a minimal problem:

Input: Expanded coefficient matrix $\mathbf{C}_{exp}$, initial permissible set $\mathcal{P}$, a training set of problem examples $\mathbf{T}$ and threshold parameter ϵ :

1. Perform permissible selection to choose $\mathcal{P}_K^*$ for $r \leq K \leq |\mathcal{P}|$ as in Problem 2.
2. For each $\mathcal{P}_K^*$, perform equation removal, intialize $\mathbf{C}_{slim} = \mathbf{C}_{exp}$,
   (a) Remove equation(s) containing singular excessive monomials
   (b) For each remaining rows i in $\mathbf{C}_{slim}$, evaluate the average errors of the solver with $\mathbf{C}_{slim/i}$ on a random subset of $\mathbf{T}$
   (c) Update C_{slim} by removing the row i^* that gives lowest mean error
   (d) Go back to (a) if the lowest average error in (c) is less than ϵ or a predefined number of iterations is reached.

Note here if the solver fail, we set the error to be infinite. Therefore, if we choose ϵ to be large enough, it will be similar to using solvability as removal criterion in [18].

5 Experimental Validation

In this section, we apply our method on different minimal geometry problems in computer vision. The first two problems are the estimation of fundamental or essential matrix with radial distortion for uncalibrated or calibrated cameras [8, 19]. The third problem is three-point stitching with radial distortion [5, 15, 20]. All these problems has been studied before and were shown to be challenging to obtain numerically stable solvers. We compare our method with previous methods with respect to both numerical accuracy as well as the size of template. The state-of-the-art method is the column-pivoting basis selection [7] which is also the building block of our method. We also compare with other two methods [18, 20] with template reduction procedures. In our implementation of a general solver with column-pivoting basis selection, we have used QR factorization to perform all the elimination steps. Our method involves a training stage where we perform equation removal and permissible selection on a fixed set of random problem examples. For all the comparisons, we test the resulting solvers on a independent set of random problems.

5.1 Nine-Point Uncalibrated Radial Distortion

The problem of estimating fundamental matrix with radial distortion is studied by [19]. Numerically stable solutions were provided by [8, 18]. The minimal case for this problem is 9-point correspondences. In general, there are 10 equations and 10 unknowns.

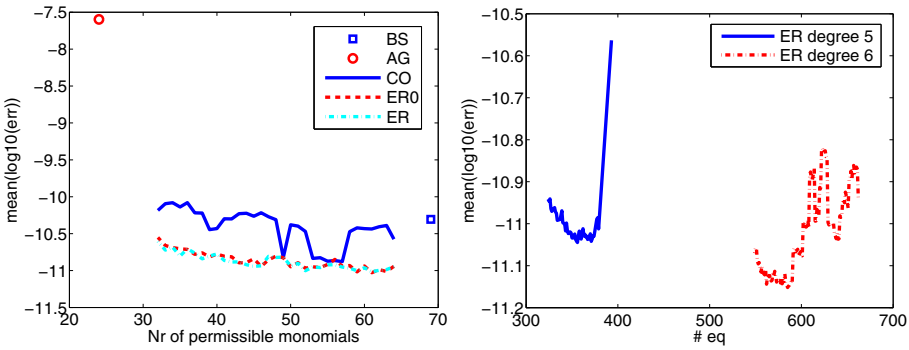

Fig. 2. Nine-point uncalibrated cameras with radial distortion. Left: Effects of permissible selection and equation removal on numerical accuracy of the polynomials solver. BS - basis selection with column-pivoting [6] , AG - automatic generator [18]), CO - permissible selection with combinatorial optimization, ER0 - ours equation removal without removing equations with singular excessives, ER - same as ER0 but with extra removal on singular excessive equations. Right: Effects of removing equations with fixed permissible (a) ER - degree 5, 393 $\times$ 390 template and (b) ER - degree 6, 680 $\times$ 595 template.

By linear elimination, one can reduced the system to 4 equations and 4 unknowns [19]. The reduced system has 24 solutions. In [18], the equations are first expanded to 497 equations which are then reduced to 179 equations and 203 monomials. There is no basis selection in this method and the basis set is chosen from the output of Macaulay2, which is of size 24. On the other hand, in [8], a fine-tuned template of 393 equations and 390 monomials are used. In this case, we use the template in [8] as our starting point. We call this as the original template. For the set of permissible monomials, we choose also exactly the same set as in [8] which is of size 69. We note that further increasing the set of permissible makes the template unsolvable.

Selecting Permissibles. Following the steps in Section 4.1, we first run the solver for 393 equations and 390 monomials, and with the set of permissible of size 69 on 5000 random problem examples. First of all, we notice that there are monomials that are never selected as basis by any problem examples. Ideally, we can easily remove these monomials from the permissible set. However, we need to force $\{x_1, x_2, x_3, x_4, 1\}$ to be in the permissible set even if they are never selected. This is because without these monomials we can not extract the solutions of the unknowns $\{x_1, x_2, x_3, x_4\}$ with the method in [7][1]. Therefore, excluding $\{x_1, x_2, x_3, x_4, 1\}$, we are able to remove 8 monomials that are never selected from the permissible set. To this end, we have the possible permissible set of size 61. We then continue to solve the problem in (8) with varying K. We manage to solve the problem for K up to 27 with branch and bound. We can see the effects of permissible selection in Figure 2 (Right, blue - solid), the solver is still stable using only smallest permissible set of size 32 (we force the 5 monomials to

[1] Note that in general, this is not required for action matrix method e.g. see [9].

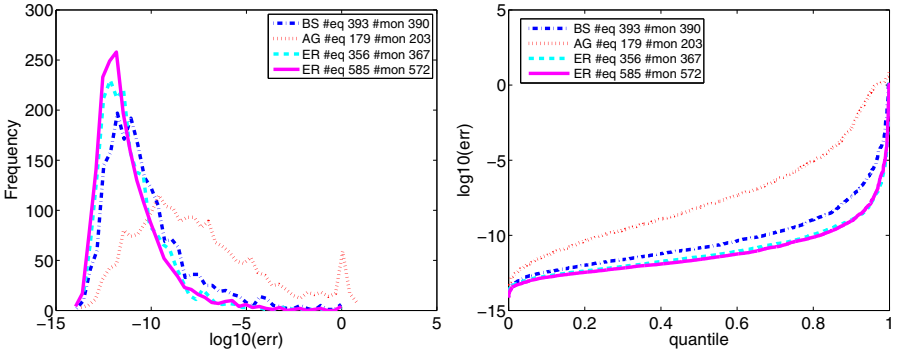

Fig. 3. Histogram (left) and quantiles (right) of log_{10} (errors) of different solvers for the uncalibrated radial distortion problem BS - [6], AG - [18] and ER - equation removal (ours) with different initial templates.

be included) which is better than without basis selection (Figure 1). The improvement gained by permissible selection is more significant for permissible set of smaller size. We also note that permissible selection alone fails to improve the accuracy of the solver consistently.

Removing Equations. We first study the interplay between equation removal and permissible selection. To do this, we perform equation removal on the template with the optimal permissible sets of different sizes from branch and bound. We can see that by performing several steps of equation removal (10 steps in this case), one get consistent improvement of solvers for different optimal permissible sets ($ER0$ in Figure 2). This suggests that one need to combine permissible selection and equation removal to get compact and numerical stable solvers. We also show that removing equations containing singular excessive monomials have little effect on the numerical stability (ER in Figure 2). Therefore, we can always apply this to speed up the removal process.

To further understand the mechanism of removing equations, we fix the permissible to be the large set of size 69. We work with both template T_1 (393×390) and the further expanded template T_2 (680×595 up to degree 6). We then proceed to remove equations using local search described in Section 4.2. For each removal step, we evaluate the mean log-errors on 100 random samples and we remove the equation without which gives the best numerical accuracy. We can see that the greedy search improves both templates at the beginning, and the templates get worse when more equations are removed (Figure 2,right)[2] . Ideally, by optimization, the large template should be converge to better template with smaller size. This is because the nature of greedy optimization as well as the randomness involved in the training data. Nevertheless, we can see that by removing equations using the greedy search, we can improve significantly the numerical behavior of T_2. Note that the initial mean log_{10}(errors) of T_2 (Figure 1, left, degree 6) is around -10.10. We can reduce the size T_2 to 585×572 (local minima in Figure 2, right) while

[2] The initial errors (see Figure 1 Left) are omitted for better visualization.

in the meantime improve the accuracy to -11.15. This is even slightly better than the best template reduced from T_1 (356×367). This indicates that one could improve numerical accuracy with large template if one carefully optimize the set of equations. We also show the distribution of test errors for solvers producing the lowest errors from our training comparing to state of the art in Figure 3. We can see that both our optimized solvers improve over [6] on accuracy. From the iteration of optimization, one get a spectrum of solvers which can be chosen with preference to speed or accuracy.

5.2 Six-Point Calibrated Radial Distortion

For estimation of essential matrix for calibrated cameras with radial distortion, the minimal case is 6 points [19]. This minimal problem consist of 16 equations in 9 unknowns. It can be reduced to 11 equations in 4 unknown with linear elimination, which has 52 solutions. In [18], a solver with template of 238 equations and 290 monomials is automatically generated. While in [8], a template with 320 equation and 363 monomials is used after fine tuning the degree of monomials multiplied with. For this problem, we do not find the the tuning parameter for the smaller template (320 by 363). Therefore, we work with a larger template reported in the paper (356 by 378, with monomials up to degree eight).

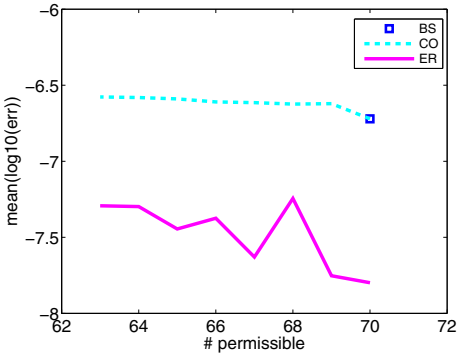

Fig. 4. Six-point uncalibrated cameras with radial distortion. Effects of permissible selection and equation removal. BS - [6], CO - permissible selection with combinatorial optimization, ER - 5 steps of equation removal with permissibles from CO.

We notice in Figure (4) that the initial template (356 by 378) is fairly unstable compared to the one reported in [8]. Here we perform first permissible selection on the template. We observe that reducing the permissible size hurts the numerical accuracy even with permissible selection. The equation removal step gain consistent improvement on all permissible sizes. We illustrate this example here to show that the local method can be applied to numerically unstable template and still gives improvements.

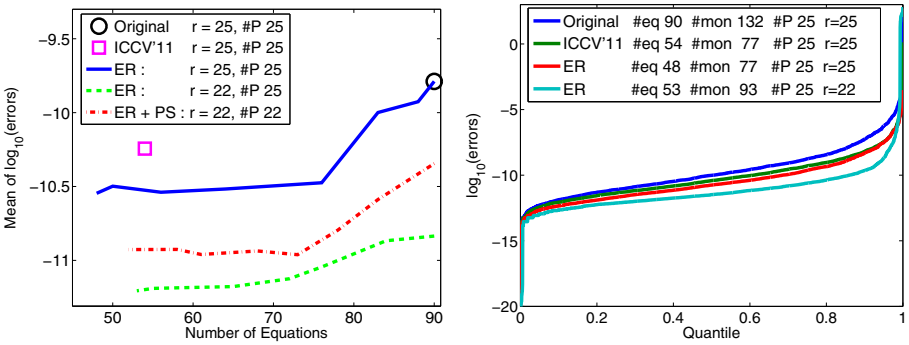

Fig. 5. Performance of different polynomials solvers on three-point stitching problem. Left: log_{10} (errors) with respect to number of equations. Original - 90×132 template as in [5], ICCV'11-optimization with sequential equation and monomial removal [20] , ER - our equation removal and PS - permissible selection. Right: (best view in colors) Quantile for the log_{10}(errors).

5.3 Three-Point Stitching

For cameras with common focal point, we can solve for the radial distortion and focal length (assuming also the same). The minimal case for this problem is 3 point correspondences [15]. A numerically stable solver based on action matrix method was proposed in [5]. The polynomial system is 2 equations in 2 unknowns with 18 solutions. By multiplying the equations with monomials up to degree eight, the expanded template consist of 90 equations in 132 monomials. For this problem, a truncation technique [7] is also applied to obtain numerical solver. It is essential a way to gain numerical stability by allowing false solutions. Specifically. instead of 18 (number of solutions), the size of the basis r is chosen to 25. In this case, the size of permissible set is also 25. To further improve numerical stability and speed, in [20], a reduction procedure is performed on the same template which is trimmed down to 54 equation in 77 monomials. The smaller template tends to give more stable solvers that are also faster.

Removing Equations. For original solver in [5], the permissible set used was of the same size of the basis (both are 25). Therefore, we do equation removal without permissible selection. We will explore in the next section on possible ways for permissible selection. We can see that one can actually remove fairly many equations and also gain numerical accuracy compared to the original template. Note that our optimized template is also smaller (48×77 for the smallest one) than the one in [20] with slightly better numerical accuracy (mean log-errors -10.56 v.s. -10.27). For fair comparison, similar to [20], we only modify the publicly available solver from [5] with corresponding equations and monomials removed.

For the solver in [5], it turns out the choice for dimension of the basis also affects the numerical accuracy. We studied this by running solvers for different pairs of $|\mathcal{P}| = 25$ and r. We find that the best combination of $|P|$ and r is using 25 permissible with $r = 22$. It is shown that by simply reducing r, we gain almost one order of improvement

on accuracy (Figure 5 , green - dash line) . Note that this is a local search itself and there might exist better combination. We leave this as future research direction. We can further reduce the size of the template by removing equations. In this case, the smallest and actually also the best template we get is of 53 equations in 93 monomials. The equation removal step again improves the numerical accuracy of the solver further.

Permissible Selection. Given that we have a working solver with $|P| > r$, ($|\mathcal{P}| = 25$ and r = 22), we can investigate the potential of permissible selection. By solving Problem (2) with $K = 22, 23, 24$, we find that having a large permissible set is always better for this problem. It is here of interest to see what numerical accuracy can we get with smallest set of permissible monomials. Using 22 permissible selected by branch and bound and equation removal step, the resulting solver (Figure 5, red - dotted line) is not as good as the best solver of size 53 with 25 permissibles and r = 22. It is a slightly slimmer solver (52×84) with trade-off in increased errors.

6 Conclusions

Previous state-of-the-art has given us methods for making the coefficient matrix as compact as possible for improving the speed of the algorithm. Other results exploit a larger set of so called permissible monomials from which the numerical linear algebra routine can choose the basis for the quotient ring. In this paper we have made several observations on the stability of polynomial equation solving using the so called action matrix method. First it is shown that adding more equations can improve numerical accuracy, but it only does so up to a point. Adding too many equations can actually decrease numerical accuracy. Secondly it is shown that the choice of so called permissible monomials also affects the numerical precision.

We present optimization algorithms that exploit these observations. Thus we are able to produce solvers that range from very compact, while still retaining good numerical accuracy to solvers that involve larger set of multiplication monomials and larger set of permissible monomials to optimize for numerical accuracy. Our method is easy to implement and is general for different problems. Therefore, it can serve as an initial tool for improving minimal problem involving large templates.

There are several interesting avenues of future research. First, the interplay between numerical linear algebra routines used and the choice of multiplication monomials and permissible monomials should be better understood. Second, the increased understanding of the mechanisms for numerical accuracy could open up for further improvements.

Acknowledgment. The research leading to these results has received funding from the strategic research projects ELLIIT and eSSENCE, and Swedish Foundation for Strategic Research projects ENGROSS and VINST(grant no. RIT08-0043).

References

1. Agarwal, S., Chandraker, M.K., Kahl, F., Kriegman, D.J., Belongie, S.: Practical Global Optimization for Multiview Geometry. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006, Part I. LNCS, vol. 3951, pp. 592–605. Springer, Heidelberg (2006)

2. Brown, M., Hartley, R., Nister, D.: Minimal solutions for panoramic stitching. In: Proc. Conf. on Computer Vision and Pattern Recognition (CVPR 2007), Minneapolis (June 2007)
3. Bujnak, M., Kukelova, Z., Pajdla, T.: A general solution to the p4p problem for camera with unknown focal length. In: Proc. Conf. Computer Vision and Pattern Recognition, Anchorage, USA (2008)
4. Bujnak, M., Kukelova, Z., Pajdla, T.: Making Minimal Solvers Fast. In: Proc. Conf. Computer Vision and Pattern Recognition (2012)
5. Byröd, M., Brown, M., Åström, K.: Minimal solutions for panoramic stitching with radial distortion. In: Proc. British Machine Vision Conference, London, United Kingdom (2009)
6. Byröd, M., Josephson, K., Åström, K.: A Column-Pivoting Based Strategy for Monomial Ordering in Numerical Gröbner Basis Calculations. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part IV. LNCS, vol. 5305, pp. 130–143. Springer, Heidelberg (2008)
7. Byröd, M., Josephson, K., Åström, K.: Fast and stable polynomial equation solving and its application to computer vision. Int. Journal of Computer Vision 84(3), 237–255 (2009)
8. Byröd, M., Kukelova, Z., Josephson, K., Pajdla, T., Åström, K.: Fast and robust numerical solutions to minimal problems for cameras with radial distortion. In: Proc. Conf. Computer Vision and Pattern Recognition, Anchorage, USA (2008)
9. Cox, D., Little, J., O'Shea, D.: Using Algebraic Geometry. Springer (1998)
10. Cox, D., Little, J., O'Shea, D.: Ideals, Varieties, and Algorithms. Springer (2007)
11. Enqvist, O., Ask, E., Kahl, F., Åström, K.: Robust Fitting for Multiple View Geometry. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part I. LNCS, vol. 7572, pp. 738–751. Springer, Heidelberg (2012)
12. Enqvist, O., Josephson, K., Kahl, F.: Optimal correspondences from pairwise constraints. In: Proc. 12th Int. Conf. on Computer Vision, Kyoto, Japan (2009)
13. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 24(6), 381–395 (1981)
14. Hartley, R., Sturm, P.: Triangulation. Computer Vision and Image Understanding 68, 146–157 (1997)
15. Jin, H.: A three-point minimal solution for panoramic stitching with lens distortion. In: Computer Vision and Pattern Recognition, Anchorage, Alaska, USA, June 24-26 (2008)
16. Kahl, F.: Multiple view geometry and the l_∞-norm. In: ICCV, pp. 1002–1009 (2005)
17. Kahl, F., Henrion, D.: Globally optimal estimates for geometric reconstruction problems. In: Proc. 10th Int. Conf. on Computer Vision, Bejing, China, pp. 978–985 (2005)
18. Kukelova, Z., Bujnak, M., Pajdla, T.: Automatic Generator of Minimal Problem Solvers. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part III. LNCS, vol. 5304, pp. 302–315. Springer, Heidelberg (2008)
19. Kukelova, Z., Pajdla, T.: Two minimal problems for cameras with radial distortion. In: OMNIVIS (2007)
20. Naroditsky, O., Daniilidis, K.: Optimizing polynomial solvers for minimal geometry problems. In: ICCV, pp. 975–982 (2011)
21. Olsson, C., Enqvist, O., Kahl, F.: A polynomial-time bound for matching and registration with outliers. In: Proc. Conf. on Computer Vision and Pattern Recognition (2008)
22. Stewénius, H.: Gröbner Basis Methods for Minimal Problems in Computer Vision. PhD thesis, Lund University (April 2005)
23. Stewénius, H., Schaffalitzky, F., Nistér, D.: How hard is three-view triangulation really? In: Proc. Int. Conf. on Computer Vision, Beijing, China, pp. 686–693 (2005)
24. Torr, P., Zisserman, A.: Robust parameterization and computation of the trifocal tensor. Image and Vision Computing 15(8), 591–605 (1997)
25. Torr, P., Zisserman, A.: Robust computation and parametrization of multiple view relations. In: Proc. 6th Int. Conf. on Computer Vision, Mumbai, India, pp. 727–732 (1998)