A New Algorithm for Computing Comprehensive Gröbner Systems *

Deepak Kapur
Dept. of Computer Science
University of New Mexico
Albuquerque, NM, USA
kapur@cs.unm.edu

Yao Sun
Key Laboratory of
Mathematics Mechanization
Academy of Mathematics and
Systems Science, CAS
Beijing, China
sunyao@amss.ac.cn

Dingkang Wang
Key Laboratory of
Mathematics Mechanization
Academy of Mathematics and
Systems Science, CAS
Beijing, China
dwang@mmrc.iss.ac.cn

ABSTRACT

A new algorithm for computing a comprehensive Gröbner system of a parametric polynomial ideal over $k[U][X]$ is presented. This algorithm generates fewer branches (segments) compared to Suzuki and Sato's algorithm as well as Nabeshima's algorithm, resulting in considerable efficiency. As a result, the algorithm is able to compute comprehensive Gröbner systems of parametric polynomial ideals arising from applications which have been beyond the reach of other well known algorithms. The starting point of the new algorithm is Weispfenning's algorithm with a key insight by Suzuki and Sato who proposed computing first a Gröbner basis of an ideal over $k[U, X]$ before performing any branches based on parametric constraints. Based on Kalkbrener's results about stability and specialization of Gröbner basis of ideals, the proposed algorithm exploits the result that along any branch in a tree corresponding to a comprehensive Gröbner system, it is only necessary to consider one polynomial for each nondivisible leading power product in $k(U)[X]$ with the condition that the product of their leading coefficients is not 0; other branches correspond to the cases where this product is 0. In addition, for dealing with a disequality parametric constraint, a probabilistic check is employed for radical membership test of an ideal of parametric constraints. This is in contrast to a general expensive check based on Rabinovitch's trick using a new variable as in Nabeshima's algorithm. The proposed algorithm has been implemented in Magma and experimented with a number of examples from different applications. Its performance (vis a vie number of branches and execution timings) has been compared with the Suzuki-Sato's algorithm and Nabeshima's speed-up algorithm. The algorithm has been successfully used to solve the famous P3P problem from computer vision.

*The first author is supported by the National Science Foundation award CCF-0729097 and the last two authors are supported by NSFC 10971217, 10771206 60821002/F02.

ISSAC 2010, 25–28 July 2010, Munich, Germany.

Categories and Subject Descriptors

I.1.2 [**Symbolic and Algebraic Manipulation**]

General Terms

Algorithms

Keywords

Gröbner basis, comprehensive Gröbner system, radical ideal membership, probabilistic check.

1. INTRODUCTION

A new algorithm for computing a comprehensive Gröbner system (CGS), as defined by Weispfenning [18] for parametric ideals (see also [7] where a related concept of parametric Gröbner system was introduced) is proposed. The main advantage of the proposed algorithm is that it generates fewer branches (segments) compared to other related algorithms; as a result, the algorithm is able to compute comprehensive Gröbner systems for many problems from different application domains which could not be done previously. In the rest of this section, we provide some motivations for comprehensive Gröbner systems and approaches used for computing them.

Many engineering problems are parameterized and have to be repeatedly solved for different values of parameters [4]. A case in point is the problem of finding solutions of a parameterized polynomial system. One is interested in finding for what parameter values, the polynomial system has a common solution; more specifically, if there are solutions, one is also interested in finding out the structure of the solution space (finitely many, infinitely many, in which their dimension, etc.). One recent application of comprehensive Gröbner systems is in automated geometry theorem proving [2] and automated geometry theorem discovery [11]. In the former, the goal is to consider all possible cases arising from an ambiguous problem formulation to determine whether the conjecture is generic enough to be valid in all cases, or certain cases have to be ruled out. In the latter, one is interested in identifying different relationship among geometric entities for different parameter values. Another recent application is in the automatic generation of loop invariants and inductive assertions of programs operating on numbers using quantifier elimination methods as proposed in [8]. The main idea is to hypothesize invariants/assertions to have a template like structure (such as a polynomial in which

the degree of every variable is ≤ 2, or a polynomial with a predetermined support), in which the presence/coefficient of a power product is parameterized. Verification conditions from the program are then generated which are formulas involving parameterized polynomial equations. The objective is to generate conditions on parameters which make these verification conditions to be valid. See [8] for more details.

Let k be a field, R be the polynomial ring $k[U]$ in the parameters $U = \{u_1, \cdots, u_m\}$, and $R[X]$ be the polynomial ring over the parameter ring R in the variables $X = \{x_1, \cdots, x_n\}$ and $X \cap U = \emptyset$, i.e., X and U are disjoint sets.

Given a polynomial set $F \subset R[X]$, we are interested in identifying conditions on parameters U such that the solution structure of the specialized polynomial system F for the values of U satisfying these conditions is different from other parameter values. One way to do this is to compute a comprehensive Gröbner system as introduced by Weispfenning, which is a finite set of triples of the form (E_i, N_i, G_i), where E_i, N_i are finite sets of polynomials in $k[U]$ and $\sigma(G_i)$ is a Gröbner basis of $\sigma(F)$, for every specialization σ such that for every $e_i \in E_i$, e_i vanishes and for at least one $n_i \in N_i$, n_i does not vanish; we will say that in that case σ satisfies the parametric constraints. Furthermore, for every specialization, there is at least one triple whose parametric constraints satisfy it. We will call each triple as a *branch* (also called a segment) in a comprehensive Gröbner system.

In 1992, Weispfenning [18] gave an algorithm for computing a comprehensive Gröbner system but it suffered from the problem of too many branches, many of which leading to the Gröbner basis $\{1\}$.[1] Since then, many improvements have been made to improve these algorithms to make them useful for different applications; see [10, 14, 15, 9]. A major breakthrough was an algorithm proposed by Suzuki and Sato [16] (henceforth called the *SS* algorithm) in which they showed how traditional implementations of Gröbner basis algorithms for polynomial rings over a field could be exploited for computing a comprehensive Gröbner basis system.

The main idea of the SS algorithm is to compute a Gröbner basis G from the parametric ideal basis in $k[U, X]$ using the block ordering in which $U \ll X$. In case G has polynomials purely in the parameters U, there are branches corresponding to each such polynomial being not equal to 0 in which case the Gröbner basis is $\{1\}$ for the specialization. For the branch when all these polynomials are 0, the Gröbner basis is G minus these polynomials under the additional condition that the leading coefficient of each polynomial is nonzero. In addition, there are branches corresponding to the cases when each of these leading coefficients is 0.

Nabeshima's speed-up algorithm [12] improves upon the SS algorithm by using the fact that (i) for every leading power product, only one coefficient needs to be made nonzero, and (ii) Rabinovitch's trick of introducing a new variable can be used to make that polynomial monic. Nabeshima reported that these tricks led to fewer branches of the SS-algorithm for most examples.

The algorithm proposed in this paper uses ideas from the construction proposed by Weispfenning[19] for computing a canonical comprehensive Gröbner basis of a parametric ideal as the starting point. The proposed algorithm integrates the ideas about essential and inessential specializations from Weispfenning's construction with the key insight in the Suzuki-Sato (SS) algorithm based on Kalkbrener's results about specialization of ideals and stability of their Gröbner bases.

First, let G be the the reduced Gröbner basis of a parametric ideal $\langle F \rangle \subset k[U, X]$ w.r.t. $\prec_{X,U}$, and let $G_r = G \cap k[U]$, the polynomials in parameters only in G. A noncomparable set G_m, which is defined in section 4, is extracted from $G \backslash G_r$, consisting only of polynomials with nondivisible powerproducts in X in G. Let h be the product of the leading coefficients of the polynomials in G_m. $(G_r, \{h\}, G_m)$ is one of the branches of the comprehensive Gröbner system of F. Based on case analysis over the leading coefficients of the polynomials in G_m, it is possible to compute the remaining branches of a comprehensive Gröbner system.

For computing a Gröbner basis for specializations along many branches, it is useful to perform radical membership check of a parametric constraint in an ideal of other parametric constraints for checking consistency. Instead of using Rabinovitch's trick of introducing a new variable for radical membership check as proposed in Nabeshima's speed-up version of the SS algorithm, we have developed a collection of useful heuristics for this check based on case analysis on whether the ideal whose radical membership is being checked, is 0-dimensional or not. In case of a positive dimensional ideal, a probabilistic check is employed after randomly specializing the independent variables of the ideal. The general check is performed as a last resort.

The paper is organized as follows. Section 2 gives notations and definitions used. Section 3 briefly reviews the Suzuki-Sato algorithm. Section 4 is the discussion of the key insights needed for the proposed algorithm; the new algorithm is presented there as well. Section 5 discusses heuristics for checking radical membership of an ideal. Section 6 illustrates the proposed algorithm on a simple example. Empirical data and comparison with the SS-algorithm and Nabeshima's speed-up algorithm are presented in Section 7. Concluding remarks follow in Section 8.

2. NOTATIONS AND DEFINITIONS

Let k be a field, R be the polynomial ring $k[U]$ in the parameters $U = \{u_1, \cdots, u_m\}$, and $R[X]$ be the polynomial ring over R in the variables $X = \{x_1, \cdots, x_n\}$ and $X \cap U = \emptyset$.

Let $PP(X)$, $PP(U)$ and $PP(U, X)$ be the sets of power products of X, U and $U \cup X$ respectively. $\prec_{X,U}$ is an admissible block term order on $PP(U, X)$ such that $U \ll X$. $\prec_X$ and $\prec_U$ is the restriction of $\prec_{X,U}$ on $PP(X)$ and $PP(U)$, respectively.

For a polynomial $f \in R[X] = k[U][X]$, the leading power product, leading coefficient and leading monomial of f w.r.t. the order $\prec_X$ are denoted by $\mathrm{lpp}_X(f)$, $\mathrm{lc}_X(f)$ and $\mathrm{lm}_X(f)$ respectively. Since f can also be regarded as an element of $k[U, X]$, in this case, the leading power product, leading coefficient and leading monomial of f w.r.t. the order $\prec_{X,U}$ are denoted by $\mathrm{lpp}_{X,U}(f)$, $\mathrm{lc}_{X,U}(f)$ and $\mathrm{lm}_{X,U}(f)$ respectively.

Given a field L, a specialization of R is a homomorphism $\sigma : R \longrightarrow L$. In this paper, we assume L to be the algebraic closure of k, and consider the specializations induced by the elements in L^m. That is, for $\bar{a} \in L^m$, the induced homomorphism $\sigma_{\bar{a}}$ is denoted as $\sigma_{\bar{a}} : f \longrightarrow f(\bar{a})$, $f \in R$. Every specialization $\sigma : R \longrightarrow L$ extends canonically to a homomorphism $\sigma : R[X] \longrightarrow L[X]$ by applying σ coefficient-wise.

DEFINITION 2.1. *Let F be a subset of $R[X]$, $A_1, \cdots, A_l$*

[1] Kapur's algorithm for parametric Gröbner bases suffered from similar weaknesses.

be algebraically constructible subsets of L^m and $G_1, \cdots, G_l$ be subsets of $R[X]$, and S be a subset of L^m such that $S \subseteq A_1 \cup \cdots \cup A_l$. A finite set $\mathcal{G} = \{(A_1, G_1), \cdots, (A_l, G_l)\}$ is called a comprehensive Gröbner system on S for F if $\sigma_{\bar{a}}(G_i)$ is a Gröbner basis of the ideal $\langle \sigma_{\bar{a}}(F) \rangle \subset L[X]$ for $\bar{a} \in A_i$ and $i = 1, \cdots, l$. Each (A_i, G_i) is called a branch of $\mathcal{G}$. If $S = L^m$, $\mathcal{G}$ is called a comprehensive Gröbner system for F.

DEFINITION 2.2. *A comprehensive Gröbner system $\mathcal{G} = \{(A_1, G_1), \cdots, (A_l, G_l)\}$ on S for F is said to be* minimal *if for every $i = 1, \cdots, l$, (i) for each $g \in G_i$, $\sigma_{\bar{a}}(\mathrm{lc}_X(g)) \neq 0$ for any $\bar{a} \in A_i$, (ii) $\sigma_{\bar{a}}(G_i)$ is a minimal Gröbner basis of the ideal $\langle \sigma_{\bar{a}}(F) \rangle \subset L[X]$ for $\bar{a} \in A_i$, and (iii) $A_i \neq \emptyset$, and furthermore, for each $i, j = 1 \cdots l$, $A_i \cap A_j = \emptyset$ whenever $i \neq j$.*

For an $F \subset R = k[U]$, the variety defined by F in L^m is denoted by $V(F)$. In this paper, the constructible set A_i always has the form: $A_i = V(E_i) \setminus V(N_i)$ where E_i, N_i are subsets of $k[U]$. If $A_i = V(E_i) \setminus V(N_i)$ is empty, the branch (A_i, G_i) is redundant.

DEFINITION 2.3. *For $E, N \subset R = k[U]$, a pair (E, N) is called a* parametric constraint. *A parametric constraint (E, N) is said to be consistent if the set $V(E) \setminus V(N)$ is not empty. Otherwise, (E, N) is called inconsistent.*

It is easy to see that the consistency of (E, N) can be checked by ensuring that at least one $f \in N$ is not in the radical of $\langle E \rangle$.

3. THE SUZUKI-SATO ALGORITHM

In this section, we briefly review the key ideas of the Suzuki-Sato algorithm [16]. The following two lemmas serve as the basis of the SS algorithm. The first lemma is a corollary of the Theorem 3.1 given by Kalkbrener in [6].

LEMMA 3.1. *Let G be a Gröbner basis of the ideal $\langle F \rangle \subset k[U, X]$ w.r.t. the order $\prec_{X,U}$. For any $\bar{a} \in L^m$, let $G_1 = \{g \in G | \sigma_{\bar{a}}(\mathrm{lc}_X(g)) \neq 0\}$. Then $\sigma_{\bar{a}}(G_1) = \{\sigma_{\bar{a}}(g) | g \in G_1\}$ is a Gröbner bases of $\langle \sigma_{\bar{a}}(F) \rangle$ in $L[X]$ w.r.t. $\prec_X$ if and only if $\sigma_{\bar{a}}(g)$ reduces to 0 modulo $\sigma_{\bar{a}}(G_1)$ for every $g \in G$.*

The next lemma, which follows from the first lemma, plays the key role in the design of the SS algorithm.

LEMMA 3.2. *Let G be a Gröbner basis of the ideal $\langle F \rangle \subset k[U, X]$ w.r.t. the order $\prec_{X,U}$. If $\sigma_{\bar{a}}(\mathrm{lc}_X(g)) \neq 0$ for each $g \in G \setminus (G \cap R)$, then $\sigma_{\bar{a}}(G)$ is a Gröbner basis of $\langle \sigma_{\bar{a}}(F) \rangle$ in $L[X]$ w.r.t. $\prec_X$ for any $\bar{a} \in V(G \cap R)$.*

The main idea of the SS algorithm is to first compute a reduced Gröbner basis, say G, of $\langle F \rangle \subset k[U, X]$ w.r.t. $\prec_{X,U}$, which is also a Gröbner basis of the ideal $\langle F \rangle \subset k[U][X]$ w.r.t. $\prec_X$. Let $\{h_1, \cdots, h_l\} = \{\mathrm{lc}_X(g) \mid g \in G \setminus R\} \subset R$. By the above lemma, $(G \cap k[U], V(h_1) \cup \cdots \cup V(h_l), G)$ forms a branch of the comprehensive Gröbner system for F. That is, for any $\bar{a} \in V(G \cap k[U]) \setminus (V(h_1) \cup \cdots \cup V(h_l))$, $\sigma_{\bar{a}}(G)$ is a Gröbner basis of $\langle \sigma_{\bar{a}}(F) \rangle$ in $L[X]$ w.r.t. $\prec_X$. To compute other branches corresponding to the specialization $\bar{a} \in V(h_1) \cup \cdots \cup V(h_l)$, Lemma 3.2 is used for each $F \cup \{h_i\}$, the above steps are repeated. Since $h_i \notin \langle F \rangle$, the algorithm terminates in finitely many steps.

As stated earlier, this algorithm can be easily implemented in most of the computer algebra systems already supporting an efficient implementation of a Gröbner basis algorithm over a polynomial ring over a field. It has very good performance since it can take advantage of well-known fast implementations for computing Gröbner bases.

The algorithm however suffers from certain weaknesses. The algorithm does not check whether $V(G \cap R) \setminus V(h)$ is empty; as a result, many redundant/unnecessary branches may be produced. In [16], an improved version of the algorithm is reported which removes redundant branches. To reduce the number of branches generated from the SS algorithm, Nabeshima proposed a speed-up algorithm in [12]. The main idea of that algorithm is to exploit disequality parametric constraints for simplification. For every leading power product in $G \setminus R$ that is a nontrivial multiple of any other leading product in it, a branch is generated by asserting its leading coefficient h_i to be nonzero. The corresponding polynomial is made monic using Rabinovitch's trick of introducing a new variable to handle the disequality $h_i \neq 0$, and the Gröbner basis computation is performed again, simplifying polynomials whose leading power products are multiples, including their parametric coefficients.

4. THE PROPOSED ALGORITHM

We present below a new algorithm for computing a comprehensive Gröbner system which avoids unnecessary branches in the SS algorithm. This is done using the radical ideal membership check for parametric constraints asserted to be nonzero. Heuristics are employed to do this check; when these heuristics fail, as exhibited by Table 2 in Section 7 on experimental results, only then the general check is performed by introducing a new variable, since this check is very inefficient because of the extra variable. Further, all parametric constraints leading to the specialized Gröbner basis being 1 are output as a single branch, leading to a compactified output.

Another major improvement of the proposed algorithm is that along any other branch for which the specialized Gröbner basis is different from 1, exactly one polynomial from $G \setminus R$ per minimal leading power product is selected. This is based on a generalization of Kalkbrener's Theorem 3.1. All these results are integrated into the proposed algorithm, resulting in considerable efficiency over the SS algorithm and Nabeshima's improved algorithm by avoiding expensive Gröbner basis computations along most branches.

The proposed algorithm is based on the following theorem. The definitions below are used in the theorem.

DEFINITION 4.1. *Given a set G of polynomials which are a subset of $k[U, X]$ and an admissible block order with $U \ll X$, let* Noncomparable(G) *be a subset, called F, of G such that (i) for every polynomial $g \in G$, there is some polynomial $f \in F$ such that $\mathrm{lpp}_X(g)$ is a multiple of $\mathrm{lpp}_X(f)$ and (ii) for any two distinct $f_1, f_2 \in F$, neither $\mathrm{lpp}_X(f_1)$ is a multiple of $\mathrm{lpp}_X(f_2)$ nor $\mathrm{lpp}_X(f_2)$ is a multiple of $\mathrm{lpp}_X(f_1)$.*

It is easy to see that $\langle \mathrm{lpp}_X(\text{Noncomparable}(G)) \rangle = \langle \mathrm{lpp}_X(G) \rangle$. The following simple example shows that Noncomparable(G) may not be unique.

Let $G = \{ax^2 - y, ay^2 - 1, ax - 1, (a+1)x - y, (a+1)y - a\} \subset \mathbb{Q}[a, x, y]$, with the lexicographic order on terms with $a < y < x$. Then $F = \{ax - 1, (a+1)y - a\}$ and $F' = \{(a+1)x - y, (a+1)y - a\}$ are both Noncomparable(G). It is easy to verify $\langle \mathrm{lpp}_X(F) \rangle = \langle \mathrm{lpp}_X(F') \rangle = \langle \mathrm{lpp}_X(G) \rangle = \langle x, y \rangle$.

DEFINITION 4.2. *Given $F \subset k[U,X]$ and $p \in k[U,X]$, p is said to be* divisible *by F if there exists an $f \in F$ such that some power product in X of p is divisible by $\mathrm{lpp}_X(f)$.*

THEOREM 4.3. *Let G be a Gröbner basis of the ideal $\langle F\rangle \subset k[U,X]$ w.r.t. an admissible block order with $U \ll X$. Let $G_r = G \cap k[U]$ and $G_m = \mathrm{Noncomparable}(G \setminus G_r)$. Denote $h = \prod_{g\in G_m} \mathrm{lc}_X(g) \in k[U]$. If σ is a specialization from $k[U]$ to L such that $\sigma(g) = 0$ for $g \in G_r$ and $\sigma(h) \neq 0$, then $\sigma(G_m)$ is a Gröbner basis of $\langle\sigma(F)\rangle$ in $L[X]$ w.r.t. $\prec_X$.*

PROOF. Consider any $p \in G \setminus (G_r \cup G_m)$; p is divisible by G_m. p can be transformed by multiplying it with the leading coefficients of polynomials in G_m and then reduced using G_m, and then this process can be repeated on the result. Let r be the *remainder* of p w.r.t. G_m in X obtained by multiplying p by the leading coefficient of $g \in G_m$ such that r does not have any power product that is a multiple of any of the leading power products of polynomials in G_m (r could be different depending upon the order in which different polynomials in G_m are used to transform p). Thus,

$$(\mathrm{lc}_X(g_1))^{\alpha_1}\cdots(\mathrm{lc}_X(g_s))^{\alpha_s}p = q_1g_1 + \cdots + q_sg_s + r, \quad (1)$$

where $g_i \in G_m$, $q_i \in k[U,X]$ for $i = 1,\cdots,s$, $r \in k[U,X]$ such that no power product of r in X is a multiple of any of the leading power products of G_m. Since $p \in \langle F\rangle$, $r \in \langle F\rangle$. Since G is a Gröbner basis of $\langle F\rangle$ in $k[U,X]$, r reduces to 0 by G. However, r is reduced (in normal form) w.r.t G_m in X (and hence reduced w.r.t $G\backslash G_r$ in X also, by the definition of G_m); so r reduces to 0 by G_r only and further no new power products in X can be introduced during the simplification of r by G_r. So $r \in \langle G_r\rangle \subset k[U,X]$. Additionally, $\mathrm{lpp}_X(p) \succeq \mathrm{lpp}_X(q_ig_i)$ since $\mathrm{lc}_X(g_i) \in k[U]$.

Let $c = (\mathrm{lc}_X(g_1))^{\alpha_1}\cdots(\mathrm{lc}_X(g_s))^{\alpha_s}$. Apply σ to the both sides of (1), then we have:

$$\sigma(c)\sigma(p) = \sigma(q_1)\sigma(g_1) + \cdots + \sigma(q_s)\sigma(g_s) + \sigma(r).$$

Since $\sigma(h) \neq 0$ by assumption, $\sigma(\mathrm{lc}_X(g)) \neq 0$ for $g \in G_m$; $\sigma(g) = 0$ for $g \in G_r$ which implies that $\sigma(r) = 0$. Notice $0 \neq \sigma(c) \in L$ and $\mathrm{lpp}_X(p) \succeq \mathrm{lpp}_X(q_ig_i)$, using the following lemma, $\sigma(G_m)$ is a Gröbner basis of $\langle\sigma(G)\rangle = \langle\sigma(F)\rangle$. □

In the above theorem, if $G_r = \emptyset$, then G_m is actually a Gröbner basis of the ideal $\langle F\rangle \subset k(U)[X]$.

We assume that the reader is familiar with the concept of t-representations which is often used to determine if a set of polynomials is a Gröbner basis; for details, consult [1].

LEMMA 4.4. *Let G be a Gröbner basis of $\langle G\rangle \subset k[U,X]$ w.r.t. an admissible block order with $U \ll X$. Let $G_1 = \{g_1,\cdots,g_s\} \subset G$ and σ be a specialization from $k[U]$ to L such that $\sigma(\mathrm{lc}_X(g_i)) \neq 0$ for $i = 1,\cdots,s$. If for each $p \in G \setminus G_1$, there exist $p_1,\cdots,p_s \in L[X]$ such that: $\sigma(p) = p_1\sigma(g_1) + \cdots + p_s\sigma(g_s)$, where $\mathrm{lpp}_X(p) \succeq \mathrm{lpp}_X(p_i\sigma(g_i))$ for $i = 1,\cdots,s$, then $\sigma(G_1)$ is a Gröbner basis of $\langle\sigma(G)\rangle$ in $L[X]$ w.r.t. $\prec_X$.*

PROOF. By the hypothesis, it is easy to check $\sigma(G) \subset \langle\sigma(G_1)\rangle$ and hence $\sigma(G_1)$ is a basis of $\langle\sigma(G)\rangle$. So it remains to show $\sigma(G_1)$ is a Gröbner basis.

For each $g_j, g_k \in G_1$, we compute the s-polynomial of $\sigma(g_j)$ and $\sigma(g_k)$ in $L[X]$. Since $\sigma(\mathrm{lc}_X(g_j)) \neq 0$ and $\sigma(\mathrm{lc}_X(g_k)) \neq 0$, we have

$$\mathrm{spoly}(\sigma(g_j),\sigma(g_k)) = c\sigma(\mathrm{spoly}_X(g_j,g_k)), \quad (2)$$

where c is a nonzero constant in L and $\mathrm{spoly}_X(g_j,g_k) \in k[U][X]$ is the s-polynomial of g_j and g_k w.r.t. X.

Assume $G \setminus G_1 = \{g_{s+1},\cdots,g_l\}$. Since G is a Gröbner basis of $\langle G\rangle \subset k[U,X]$ and $\mathrm{spoly}_X(g_j,g_k) \in \langle G\rangle \subset k[U,X]$, there exist $h_1,\cdots,h_l \in k[U,X]$ such that

$$\mathrm{spoly}_X(g_j,g_k) = h_1g_1 + \cdots + h_lg_l,$$

where $\mathrm{lcm}(\mathrm{lpp}_X(g_j),\mathrm{lpp}_X(g_k)) \succ \mathrm{lpp}_X(h_ig_i)$ for $i = 1,\cdots,l$. Substitute back to (2), then obtain:

$$\mathrm{spoly}(\sigma(g_j),\sigma(g_k)) = c(\sigma(h_1)\sigma(g_1) + \cdots + \sigma(h_l)\sigma(g_l)), \quad (3)$$

where $\mathrm{lcm}(\mathrm{lpp}_X(\sigma(g_j)),\mathrm{lpp}_X(\sigma(g_k))) = \mathrm{lcm}(\mathrm{lpp}_X(g_j),\mathrm{lpp}_X(g_k)) \succ \mathrm{lpp}_X(h_ig_i) \succeq \mathrm{lpp}_X(\sigma(h_i))\mathrm{lpp}_X(g_i)$ for $i = 1,\cdots,l$. The next step is to use the hypothesis that for each $p \in G \setminus G_1$, there exist $p_1,\cdots,p_s \in L[X]$ such that: $\sigma(p) = p_1\sigma(g_1) + \cdots + p_s\sigma(g_s)$, where $\mathrm{lpp}_X(p) \succeq \mathrm{lpp}_X(p_i\sigma(g_i))$ for $i = 1,\cdots,s$. Substitute these representations back to (3), we get

$$\mathrm{spoly}(\sigma(g_j),\sigma(g_k)) = p'_1\sigma(g_1) + \cdots + p'_s\sigma(g_s), \quad (4)$$

where $p'_1,\cdots,p'_s \in L[X]$ and $\mathrm{lcm}(\mathrm{lpp}_X(\sigma(g_j)),\mathrm{lpp}_X(\sigma(g_k))) \succ \mathrm{lpp}_X(p'_i\sigma(g_i))$ for $i = 1,\cdots,s$. In fact, (4) is a t-representation of $\mathrm{spoly}(\sigma(g_j),\sigma(g_k))$ with $t \prec \mathrm{lcm}(\mathrm{lpp}_X(\sigma(g_j)),\mathrm{lpp}_X(\sigma(g_k)))$. Therefore, by the theory of t-representations, $\sigma(G_1)$ is a Gröbner basis. The lemma is proved. □

4.1 Algorithm

We are now ready to give the algorithm for computing a minimal comprehensive Gröbner system. Its proof of correctness uses Theorem 4.3. Its termination can be proved in a way similar to the SS algorithm presented in [16].

In order to keep the presentation simple so that the correctness and termination of the algorithm are evident, we have deliberately avoided tricks and optimizations such as factoring h below. All the tricks suggested in the SS algorithm can be used here as well. In fact, our implementation incorporates fully these optimizations.

Algorithm PGBMain
Input: (E,N,F): E, N, finite subsets of $k[U]$; F, a finite subset of $k[U,X]$.
Output: a finite set of 3-tuples (E_i,N_i,G_i) such that $\{(V(E_i)\setminus V(N_i),G_i)\}$ constitute a minimal comprehensive Gröbner system of F on $V(E)\setminus V(N)$.
begin
if $V(E)\setminus V(N) = \emptyset$ **then return** $\emptyset$ **end if**
$G \leftarrow$ ReducedGröbnerBasis$(F \cup E, \prec_{X,U})$
if $1 \in G$ **then return** $\{(E,N,\{1\})\}$ **end if**
$G_r \leftarrow G \cap k[U]$ # $V(G_r) \subset V(E)$
if $(V(E)\setminus V(G_r))\setminus V(N) = \emptyset$
  then $\mathcal{PGB} \leftarrow \emptyset$
  else $\mathcal{PGB} \leftarrow \{(E, G_r \wedge N, \{1\})\}$
end if
if $V(G_r)\setminus V(N) = \emptyset$
  then return $\mathcal{PGB}$;
  else $G_m \leftarrow$ Noncomparable$(G\setminus G_r)$
    $\{h_1,\cdots,h_s\} \leftarrow \{\mathrm{lc}_X(g) : g \in G_m\}$
    $h \leftarrow \mathrm{lcm}\{h_1,\cdots,h_s\}$;
    if $(V(G_r)\setminus V(N))\setminus V(h) \neq \emptyset$ **then**
      $\mathcal{PGB} \leftarrow \mathcal{PGB} \cup \{(G_r, N\wedge\{h\}, G_m)\}$
    end if
    $\mathcal{PGB} \leftarrow \mathcal{PGB} \cup$ PGBMain$(G_r\cup\{h_1\}, N, G\setminus G_r)\cup$
      PGBMain$(G_r\cup\{h_2\}, N\wedge\{h_1\}, G\setminus G_r)\cup$
      PGBMain$(G_r\cup\{h_3\}, N\wedge\{h_1h_2\}, G\setminus G_r)\cup$
      ·····

PGBMain($G_r \cup \{h_s\}, N \wedge \{h_1 \cdots h_{s-1}\}, G \setminus G_r$)
return $\mathcal{PGB}$
end if
end

In the above algorithm, $A \wedge B = \{fg | f \in A, g \in B\}$. Checking whether $V(A) \setminus V(B)$ is empty, is equivalent to the inconsistency of the parametric constraint (A, B). Similarly checking whether $(V(A) \setminus V(B)) \setminus V(C) = V(A) \setminus (V(B) \cup V(C))$ is empty, is equivalent to checking whether $(A, B \wedge C)$ is inconsistent. The next section focuses on how the consistency check of a parametric constraint is performed.

As should be evident, a branch is never generated for the case when (E_i, N_i) is inconsistent. Further, the constructible sets are disjoint by construction. More importantly, branching is done only based on the leading coefficients of $G_m = \text{Noncomparable}(G \setminus G_r)$, instead of the whole $G \setminus G_r$. As a result, the number of branches generated by the above algorithm is strictly smaller than that of the branches in the SS algorithm. In addition, efficient heuristics are employed to perform the consistency check; as a last resort only when other heuristics do not work, we introduce a new variable to do the consistency check. In fact, this general check is rarely performed as confirmed by experimental data discussed in Section 7. Because of these optimizations, the proposed algorithm has a much better performance than the SS algorithm as well as Nabeshima's speed-up algorithm, as experimentally shown in Section 7.

As shown in [16], a comprehensive Gröbner basis can be computed by adapting the above algorithm for computing a comprehensive Gröbner system by using a new variable. The same technique can be applied to the above algorithm as well for computing a comprehensive Gröbner basis.

5. CONSISTENCY OF PARAMETRIC CONSTRAINTS

As should be evident from the above description of the algorithm, there are two main computational steps which are being repeatedly performed: (i) Gröbner basis computations, and (ii) checking consistency of parametric constraints. As stated above, a parametric constraint (E, N), $E, N \subset k[U]$ is inconsistent if and only if for each $f \in N$, f is in the radical ideal of $\langle E \rangle$. This section discusses heuristics we have integrated into the implementation of the algorithm for the check whether $(E, \{f\})$ is inconsistent. In this section, we always assume that E itself is a Gröbner basis.

A general method to check whether $f \in \sqrt{\langle E \rangle}$ is to introduce a new variable y and compute the Gröbner basis G_y of $\langle E \cup \{fy - 1\} \rangle \subset k[U, y]$ for any admissible monomial order. If $G_y = \{1\}$, then $f \in \sqrt{\langle E \rangle}$ and $(E, \{f\})$ is inconsistent. Otherwise, $(E, \{f\})$ is consistent. However, this method can be, in general, very expensive partly because of introduction of a new variable. Consequently, this method is used only as a last resort when other heuristics fail.

The first heuristic is to check whether f is in the ideal generated by E; since in the algorithm, a Gröbner basis of E is already available, the normal form of f is computed; if it is 0, then f is in the ideal of E implying that $(E, \{f\})$ is inconsistent. This heuristic turns out to be quite effective as shown from experimental results in Section 7.

Otherwise, different heuristics are used depending upon whether E is 0-dimensional or not. In case E is 0-dimensional, the method discussed in the next subsection for the radical membership check is complete, i.e., it decides whether f is in the radical ideal of E or not. In case E is of positive dimension, then roughly, independent variables are assigned randomly, hopefully, resulting in a 0-dimensional ideal, for which the radical membership check can be done. However, this heuristic is not complete. If this heuristic cannot determine whether $(E, \{f\})$ is inconsistent, then another heuristic is employed that checks whether f^{2^k} is in the ideal of E for a suitably small value of k.

5.1 Ideal(E) is 0-dimensional

For the case when E is 0-dimensional, linear algebra techniques can be used to check the radical membership in E. The main idea is to compute the characteristic polynomial of the linear map associated with f, which can be efficiently done using a Gröbner basis of E.

Let $A = k[U]/\langle E \rangle$. Consider the map induced by $f \in k[U]$: $m_f : A \longrightarrow A$, $[g] \longmapsto [fg]$, where $g \in k[U]$ and $[g]$ is its equivalence class in A.

See [3, 17] for the proofs of the following lemmas.

LEMMA 5.1. *Assume that the map m_f is defined as above. Then,*
(1) m_f is the zero map exactly when $f \in \langle E \rangle$.
(2) For a univariate polynomial q over k, $m_{q(f)} = q(m_f)$.
(3) $p_f(f) \in \langle E \rangle$, where p_f is the characteristic polynomial of m_f.

LEMMA 5.2. *Let $p_f \in k[\lambda]$ be the characteristic polynomial of m_f. Then for $\alpha \in L$, the following statements are equivalent.*
(1) α is a root of the equation $p_f(\lambda) = 0$.
(2) α is a value of the function f on $V(E)$.

Using these lemmas, we have:

PROPOSITION 5.3. *Let $p_f \in k[\lambda]$ be the characteristic polynomial of m_f and $d = \deg(p_f)$.*
(1) $p_f = \lambda^d$ if and only if $f \in \sqrt{\langle E \rangle}$.
(2) $p_f = q$ and $\lambda \nmid q$ if and only if there exists $g \in k[U]$ such that $gf \equiv 1 \mod \langle E \rangle$.
(3) $p_f = \lambda^{d'} q$, where $0 < d' < d$ and $\lambda \nmid q$ if and only if $f \notin \sqrt{\langle E \rangle}$ and there exists $g \notin \sqrt{\langle E \rangle}$ such that $fg \in \sqrt{\langle E \rangle}$.

PROOF. (1) $\Rightarrow$) If $p_f = \lambda^d$, then $p_f(f) = f^d \in \langle E \rangle$ by lemma 5.1, which shows $f \in \sqrt{\langle E \rangle}$. $\Leftarrow$) Since $f \in \sqrt{\langle E \rangle}$, 0 is the sole value of the function f on $V(E)$. By lemma 5.2, $p_f = \lambda^d$.

(2) $\Rightarrow$) If $p_f = q$ and $\lambda \nmid q$, then there exist $a, b \in k[\lambda]$ such that $a\lambda + bp_f = 1$. Substitute λ by f. Then obtain $a(f)f + b(f)p_f(f) = 1$. $p_f(f) \in \langle E \rangle$ shows $a(f)f \equiv 1 \mod \langle E \rangle$. $\Leftarrow$) If there exists $g \in k[U]$ such that $gf \equiv 1 \mod \langle E \rangle$, then all the values of the function f on $V(f)$ are not 0, which means the roots of $p_f(\lambda) = 0$ are not 0 as well by the above lemma. So $\lambda \nmid p_f$.

(3) $\Rightarrow$) If $p_f = \lambda^{d'} q$, where $0 < d' < d$ and $\lambda \nmid q$, then we have $f \notin \sqrt{\langle E \rangle}$ by (1). By lemma 5.1, $p_f(f) = f^{d'} q(f) \in \langle E \rangle$, and hence, $fq(f) \in \sqrt{\langle E \rangle}$. It remains to show $q(f) \notin \sqrt{\langle E \rangle}$. We prove this by contradict. If $q(f) \in \sqrt{\langle E \rangle}$, then there exists an integer $c > 0$ such that $q^c(f) \in \langle E \rangle$, which implies $m_{q^c(f)} = q^c(m_f) = 0$. Thus, q^c is a multiple of the minimal polynomial of m_f and hence all the irreducible factors of p_f should be factors of q^c. But this contradicts with $\lambda \nmid q$. $\Leftarrow$) Since $f, g \notin \sqrt{\langle E \rangle}$ and $fg \in \sqrt{\langle E \rangle}$, both

f and g are nonzero functions on $V(E)$, but fg is a zero function on $V(E)$. This implies that f vanishes on some but not all points of $V(E)$. By lemma 5.2, $p_f = \lambda^{d'} q$, where $0 < d' < d$ and $\lambda \nmid q$. $\square$

For the case (2) of proposition 5.3, clearly $V(E) \setminus V(f) = V(E)$ holds. For the case (3), it is easy to check $V(E) \setminus V(f) = V(E \cup \{q(f)\})$ by Lemma 5.2. So the parametric constraint $(E, \{f\})$ is equivalent to $(E \cup \{q(f)\}, \{1\})$, which converts the disequality constraint into equality constraint. Both (2) and (3) will speed up the implementation of the new algorithm.

If E is zero-dimensional, then $k[U]/\langle E\rangle$ is a finite vector space and the characteristic polynomial of m_f can be generated in [3]. Since in our algorithm, E itself is a Gröbner basis, the complexity of doing radical membership check is of polynomial time, which is much more efficient than the general method based on Rabinovitch's trick.

The following algorithm is based on the above theory:

Algorithm *Zero-DimCheck*
Input: $(E, \{f\})$: E is the Gröbner basis of the zero dimensional ideal $\langle E\rangle$; f, a polynomial in $k[U]$.
Output: **true** (consistent) or **false** (inconsistent).
begin
$p_f \leftarrow$ characteristic polynomial of m_f defined on $k[U]/\langle E\rangle$
$d \leftarrow \deg(p_f)$
if $p_f \neq \lambda^d$ **then return true else return false end if**
end

5.2 Ideal(E) is of positive dimension

We discuss two heuristics, *CCheck* and *ICheck*, for radical membership check; neither one is complete.

A subset V of U is *independent* modulo the ideal I if $k[V] \cap I = \{0\}$. An independent subset of U is maximal if there is no independent subset containing V properly.

The following proposition is well-known.

PROPOSITION 5.4. *Let $I \subset k[U]$ be an ideal and $\prec_U$ be a graded order on $k[U]$. If $k[V] \cap \mathrm{lpp}_U(I) = \emptyset$, then $k[V] \cap I = \emptyset$. Furthermore, the maximal independent subset modulo $\mathrm{lpp}_U(I)$ is also a maximal independent subset modulo I.*

A maximal independent subset modulo the monomial ideal of $\langle E\rangle$ can be easily computed; the above proposition thus provides a method to compute the maximal independent subset modulo an ideal.

The following theorem is obvious, so the proof is omitted.

THEOREM 5.5. *Let $\langle E\rangle \subset k[U]$ with positive dimension, V be a maximal independent subset modulo $\langle E\rangle$, and $\bar{\alpha}$ be an element in k^l where l is the cardinality of V. If $f|_{V=\bar{\alpha}} \notin \sqrt{\langle E|_{V=\bar{\alpha}}\rangle}$, then $f \notin \sqrt{\langle E\rangle}$ i.e. $(E, \{f\})$ is consistent.*

Since V is a maximal independent subset modulo $\langle E\rangle$, the ideal $\langle E\rangle$ becomes a zero dimensional ideal in $k[U \setminus V]$ with probability 1 by setting V to a value in k^l randomly when the characteristic of k is 0. In this case, we can use the technique provided in the last subsection to check if $f|_{V=\bar{\alpha}} \notin \sqrt{\langle E|_{V=\bar{\alpha}}\rangle}$. If $(E|_{V=\bar{\alpha}}, f|_{V=\bar{\alpha}})$ is consistent, then $(E, \{f\})$ is consistent. This gives an algorithm for checking the consistence of $(E, \{f\})$. When $f \notin \sqrt{\langle E\rangle}$, this algorithm can detect it efficiently.

Algorithm *CCheck*
Input: $(E, \{f\})$: E is the Gröbner basis of $\langle E\rangle$ w.r.t. a graded monomial order $\prec_U$; f, a polynomial in $k[U]$.
Output: **true** (consistent) or **false** .
begin
$V \leftarrow$ independent variables of $\langle \mathrm{lpp}_U(E)\rangle$
$\bar{\alpha} \leftarrow$ a random element in k^l
$spE \leftarrow$ GröbnerBasis$(E|_{V=\bar{\alpha}}, \prec_U)$
if $\langle spE\rangle$ is zero dimension in $k[U \setminus V]$ **then**
  if *Zero-DimCheck*$(spE, f|_{V=\bar{\alpha}})$ =**true then**
    return true
  end if
end if;
return false
end

In the above algorithm, we only need to compute the Gröbner basis of $\langle E_{V=\bar{\alpha}}\rangle$ which is usually zero dimensional and has fewer variables. So *CCheck* is more efficient than the general method which needs to compute the Gröbner basis of $\langle E \cup \{fy - 1\}\rangle$ whose dimension is positive.

If *CCheck*$(E, \{f\})$ returns true, then $(E, \{f\})$ is consistent. However, if *CCheck*$(E, \{f\})$ returns false, it need not be the case that $(E, \{f\})$ is inconsistent.

The following simple heuristic *ICheck* checks whether f^{2^k} is in the ideal generated by E by repeatedly squaring the normal form of f^{2^i} in an efficient way.

Algorithm *ICheck*
Input: $(E, \{f\})$: E is the Gröbner basis of $\langle E\rangle$ w.r.t. $\prec_U$; f, a polynomial in $k[U]$.
Output: **true** (inconsistent) or **false** .
begin
loops$\leftarrow$ an integer given in advance
$p \leftarrow f$
for i **from** 1 **to** *loops* **do**
  $\{m_1, \cdots, m_l\} \leftarrow$ monomials of p
  $s \leftarrow 0$
  for $m \in \{m_1, \cdots, m_l\}$ **do**
    $s \leftarrow s +$NormalForm$(p \cdot m, E)$
  end for
  if $s = 0$ **then return true end if**
  $p \leftarrow s$
end for
return false
end

Clearly, if *ICheck*$(E, \{f\})$ returns true, then $(E, \{f\})$ is inconsistent.

5.3 Putting All Together

The above discussed checks are done in the following order for checking the consistency of a parametric constraint $(E, \{f\})$. First check whether f is in the ideal of E; this check can be easily done by computing the normal form of f using a Gröbner basis of E which is readily available. If yes, then the constraint is inconsistent. If no, then depending upon the dimension of the ideal of E, either *Zero-DimCheck* or *CCheck* is performed. If E is 0-dimensional, then the check is complete in that it decides whether the constraint is consistent or not. If E is of positive dimension then if *CCheck* returns true, the constraint is consistent; otherwise, *ICheck* is performed. If *ICheck* succeeds, then the constraint is inconsistent. Finally, the general check is performed by computing a Gröbner basis of $E \cup \{fy - 1 = 0\}$, where y is a new variable different from U.

6. A SIMPLE EXAMPLE

The proposed algorithm is illustrated on an example.

EXAMPLE 6.1. *Let* $F = \{ax - b, by - a, cx^2 - y, cy^2 - x\} \subset \mathbb{Q}[a,b,c][x,y]$, *with the block order* $\prec_{X,U}$, $\{a,b,c\} \ll \{x,y\}$; *within each block,* $\prec_X$ *and* $\prec_U$ *are graded reverse lexicographic orders with* $y < x$ *and* $c < b < a$, *respectively.*

(1) We have $E = \emptyset$, $N = \{1\}$: the parametric constraint (E, N) is consistent. The reduced Gröbner basis of $\langle F\rangle$ w.r.t. $\prec_{X,U}$ is $G = \{x^3 - y^3, cx^2 - y, ay^2 - bc, cy^2 - x, ax - b, bx - acy, a^2y - b^2c, by - a, a^6 - b^6, a^3c - b^3, b^3c - a^3, ac^2 - a, bc^2 - b\}$; $G_r = G \cap \mathbb{Q}[a,b,c] = \{a^6 - b^6, a^3c - b^3, b^3c - a^3, ac^2 - a, bc^2 - b\}$. It is easy to see that (E, G_r) and $(E, G_r \wedge N)$ are consistent. This leads to the trivial branch of the comprehensive Gröbner system for F: $(\emptyset, G_r, \{1\})$.

(2) $G \setminus G_r = \{x^3 - y^3, cx^2 - y, ay^2 - bc, cy^2 - x, ax - b, bx - acy, a^2y - b^2c, by - a\}$; $G_m = \text{Noncomparable}(G \setminus G_r) = \{bx - acy, by - a\}$. Further, $h = \text{lcm}\{\text{lc}_X(bx - acy), \text{lc}_X(by - a)\} = b$. This results in another branch of the comprehensive Gröbner system for F corresponding to the case when all polynomials in G_r are 0 and $b \neq 0$: $(G_r, \{b\}, G_m)$. Notice that $(G_r, \{b\})$ is consistent, which is detected using the *ZeroDimCheck*.

(3) The next case to consider is when $b = 0$. The Gröbner basis of $G_r \cup \{b\}$ is $\{a^3, ac^2 - a, b\}$. This is the input E' in the recursive call of PGBMain, with the other input being $N' = \{1\}$ and $F' = G \setminus G_r$. It is easy to see that (E', N') is consistent. The reduced Gröbner basis for $F' \cup E'$ is: $G' = \{x^3 - y^3, cx^2 - y, cy^2 - x, a, b\}$ of which $G'_r = \{a, b\}$. It is easy to check the parametric constraint (E', G'_r) is inconsistent: the check for a being in the radical ideal of E' is confirmed by *Icheck*; b is in the ideal of E'. So no branch is generated from this case.

$G'_m = Noncomparable(G' \setminus G'_r) = \{cx^2 - y, cy^2 - x\}$ and $h' = \text{lcm}\{\text{lc}_X(cx^2 - y), \text{lc}_X(cy^2 - x)\} = c$. This results in another branch: $(G'_r, \{c\}, G'_m)$.

(4) For the case when $h' = c = 0$, $E'' = \{a, b, c\}$ is the Gröbner basis of $G'_r \cup \{c\}$ and $N'' = \{1\}$, $F'' = \{x^3 - y^3, cx^2 - y, cy^2 - x\}$. The Gröbner basis for $F'' \cup E''$ is $G'' = \{x, y, a, b, c\}$. Then $G''_r = \{a, b, c\}$ and $G''_m = \{x, y\}$. Since $h'' = \text{lcm}\{\text{lc}_X(x), \text{lc}_X(y)\} = 1$, this gives another branch: $(G''_r, \{1\}, G''_m)$. As $h'' = 1$, no other branches are created and the algorithm terminates.

The result is a comprehensive Gröbner system for F:

$$\begin{cases} \{1\}, & \text{if } a^6 - b^6 \neq 0 \text{ or } a^3c - b^3 \neq 0 \text{ or } b^3c - a^3 \neq 0 \text{ or } ac^2 - a \neq 0 \text{ or } bc^2 - b \neq 0, \\ \{bx - acy, by - a\}, & \text{if } a^6 - b^6 = a^3c - b^3 = b^3c - a^3 = ac^2 - a = bc^2 - b = 0 \text{ and } b \neq 0, \\ \{cx^2 - y, cy^2 - x\} & \text{if } a = b = 0 \text{ and } c \neq 0, \\ \{x, y\} & \text{if } a = b = c = 0. \end{cases}$$

7. IMPLEMENTATION AND COMPARATIVE PERFORMANCE

The proposed algorithm is implemented in the system *Magma* and has been experimented with a number of examples from different application domains including geometry theorem proving and computer vision. Since the algorithm is able to avoid most unnecessary branches and computations, it is efficient and can compute comprehensive Gröbner systems for most problems in a few seconds. In particular, we have been successful in completely solving the famous P3P problem for pose-estimation from computer vision, which is investigated by Gao et al [5] using the characteristic set method; see the polynomial system below.

We have compared our implementation with the implementations of Suzuki and Sato's algorithm as well as Nabeshima's speed-up version as available in the PGB (ver20090915) package implemented in Asir/Risa system. We have picked examples F3, F5, F6 and F8 from [12] and the examples E4 and E5 from [11]; many other examples can be solved in essentially no time. To get more complex examples, we modified problems from the F5, F6 and F8 in [12] slightly, and they are labeled as S1, S2 and S3.

The polynomials for S1, S2, S3 and P3P are given below:
$S1 = \{ax^2y + bx^2 + y^3, ax^2y + bxy + cy^2, ay^3 + bx^2y + cxy\}, X = \{x, y\}, U = \{a, b, c\}$;
$S2 = \{x^4 + abx^3 + bcx^2 + cdx + da, 4x^3 + 3abx^2 + 2bcx + cd\}, X = \{x\}, U = \{a, b, c, d\}$;
$S3 = \{ax^2 + byz + c, cw^2 + by + z, (x - z)^2 + (y - w)^2, 2dxw - 2byz\}, X = \{x, y, z, w\}, U = \{a, b, c, d\}$;
$P3P = \{(1 - a)y^2 - ax^2 - py + arxy + 1, (1 - b)x^2 - by^2 - qx + brxy + 1\}, X = \{x, y\}, U = \{p, q, r, a, b\}$.

Table 1: Timings

Example	Algorithm	Sys.	Br.	time(sec.)
F3	*pgbM*	Magma	6	0.016
	Suzuki-Sato	Risa/Asir	31	0.5148
	Nabeshima	Risa/Asir	22	0.8268
F5	*pgbM*	Magma	8	0.016
	Suzuki-Sato	Risa/Asir	11	0.0156
	Nabeshima	Risa/Asir	54	16.04
F6	*pgbM*	Magma	8	0.078
	Suzuki-Sato	Risa/Asir	875	35.97
	Nabeshima	Risa/Asir	17	0.078
F8	*pgbM*	Magma	18	0.140
	Suzuki-Sato	Risa/Asir	–	$> 1h$
	Nabeshima	Risa/Asir	–	$> 1h$
E4	*pgbM*	Magma	9	0.016
	Suzuki-Sato	Risa/Asir	15	0.0468
	Nabeshima	Risa/Asir	24	0.7644
E5	*pgbM*	Magma	38	0.546
	Suzuki-Sato	Risa/Asir	98	24.09
	Nabeshima	Risa/Asir	102	12.53
S1	*pgbM*	Magma	29	3.167
	Suzuki-Sato	Risa/Asir	–	$> 1h$
	Nabeshima	Risa/Asir	–	$> 1h$
S2	*pgbM*	Magma	15	1.420
	Suzuki-Sato	Risa/Asir	–	$> 1h$
	Nabeshima	Risa/Asir	49	5.413
S3	*pgbM*	Magma	30	3.182
	Suzuki-Sato	Risa/Asir	–	$> 1h$
	Nabeshima	Risa/Asir	–	$> 39m$ Error
P3P	*pgbM*	Magma	42	6.256
	Suzuki-Sato	Risa/Asir	–	$> 1h$
	Nabeshima	Risa/Asir	–	$> 28m$ Error

In the above table, the algorithm *pgbM* is the proposed algorithm; the algorithm *cgs*1 stands for the Suzuki-Sato's algorithm, and the algorithm *cgs_con*1 stands for the Nabeshima's algorithm from Nabeshima's PGB package [13] were used. All the timings in the table are obtained on Core2 Duo3.0 with 4GB Memory running WinVista64.

As is evident from Table 1, the proposed algorithm gen-

erates fewer branches. This is why our algorithm has better performance than the others.

An efficient check for the consistency of parametric constraints is critical for the performance of the proposed algorithm. The role of various checks discussed in Section 5 has been investigated in detail. This is reported in Table 2 below, where *Tri*, 0-*dim*, *C*, *I*, and *Gen* stand, respectively, for the *trivial check*, *Zero-DimCheck*, the *CCheck*, *ICheck*, and the *general method*.

Table 2: Info about various consistence checks

Exp		Tri.	0-dim	pos-dim		Gen.	Total
				C.	I.		
F3	Num	10	2	3	0	0	15
	≈ %	67%	13%	20%	0%	0%	
F5	Num	22	0	10	0	0	32
	≈ %	69%	0%	31%	0%	0%	
F6	Num	22	0	7	8	1	38
	≈ %	58%	0%	18%	21%	3%	
F8	Num	47	0	29	0	0	76
	≈ %	62%	0%	38%	0%	0%	
E4	Num	10	7	3	0	0	20
	≈ %	50%	35%	15%	0%	0%	
E5	Num	67	10	55	0	6	138
	≈ %	49%	7%	40%	0%	4%	
S1	Num	115	21	36	0	11	183
	≈ %	63%	11%	20%	0%	6%	
S2	Num	36	0	27	6	0	69
	≈ %	52%	0%	39%	9%	0%	
S3	Num	110	9	45	1	0	165
	≈ %	67%	5%	27%	1%	0%	
P3P	Num	144	4	63	3	13	227
	≈ %	63%	2%	28%	1%	6%	

About 61% of the consistency check is settled by the trivial check that a polynomial is in the ideal; about the remaining 36% of the consistency check is resolved by the *Zero-DimCheck*, *CCheck* and *ICheck*. The general method for checking consistency using Rabinovitch's trick of introducing a new variable is rarely used (almost 3%). We believe that this is one of the main reasons why our proposed algorithm has a vastly improved performance over Nabeshima's speed-up algorithm which relies on using the general check for the consistency of the parametric constraints.

8. CONCLUDING REMARKS

A new algorithm for computing a comprehensive Gröbner system has been proposed using ideas from Kalkbrener, Weispfenning, Suzuki and Sato. Preliminary experiments suggest that the algorithm is far superior in practice in comparison to Suzuki and Sato's algorithm as well as Nabeshima's speed-up version vis a vis the number of branches generated as well as execution speed. Particularly, we are able to do examples such as the famous P3P problem from computer vision, which have been found extremely difficult to solve using most symbolic computation algorithms.

We believe that the proposed algorithm can be further improved. We are exploring conditions under which the radical membership ideal check is unwarranted and additional ideas to make this check more efficient whenever it is needed. We also plan to compare our implementation with other implementations of comprehensive Gröbner system algorithms.

9. REFERENCES

[1] Becker, T. and Weispfenning, V. (1993). Gröbner Bases, A Computational Approach to Commutative Algebra. Springer-Verlag. ISBN 0-387-97971-9.

[2] Chen, X.F., Li, P., Lin, L., Wang, D.K.(2005) Proving geometric theorems by partitioned-parametric Gröbner bases. In: Hong, H., Wang, D. (eds.) ADG 2004. LNAI, vol. 3763, 34-44. Springer.

[3] Cox, D., Little, J., O'Shea, D. (2004). Using Algebraic Geometry. New York, Springer. 2nd edition. ISBN 0-387-20706-6.

[4] Donald, B., Kapur, D., and Mundy, J.L.(eds.) (1992). Symbolic and Numerical Computation for Artificial Intelligence. Academic Press.

[5] Gao, X.S., Hou, X., Tang, J. and Chen, H. (2003). Complete Solution Classification for the Perspective-Three-Point Problem, IEEE Tran. on PAMI, 930-943, 25(8).

[6] Kalkbrener, K. (1997). On the stability of Gröbner bases under specialization, J. Symb. Comp. 24, 1, 51-58.

[7] Kapur, D.(1995). An approach to solving systems of parametric polynomial equations. In: Saraswat, Van Hentenryck (eds.) Principles and Practice of Constraint Programming, MIT Press, Cambridge.

[8] Kapur, D.(2006). A Quantifier Elimination based Heuristic for Automatically Generating Inductive Assertions for Programs, J. of Systems Science and Complexity, Vol. 19, No. 3, 307-330.

[9] Manubens, M. and Montes, A. (2006). Improving DISPGB Algorithm Using the Discriminant Ideal, J. Symb. Comp., 41, 1245-1263.

[10] Montes, A. (2002). A new algorithm for discussing Gröbner basis with parameters, J. Symb. Comp. 33, 1-2, 183-208.

[11] Montes, A., Recio, T.(2007). Automatic discovery of geometry theorems using minimal canonical comprehensive Gröbner systems. ADG 2006, LNAI 4869, Springer, 113-138.

[12] Nabeshima, K.(2007) A Speed-Up of the Algorithm for Computing Comprehensive Gröbner Systems. In Brown, C., editor, ISSAC2007, 299-306.

[13] Nabeshima, K.(2007) PGB: A Package for Computing Parametric Gröbner Bases and Related Objects. Conference posters of ISSAC 2007, 104-105.

[14] Suzuki, A. and Sato, Y. (2002). An alternative approach to Comprehensive Gröbner bases. In Mora, T., editor, ISSAC2002, 255-261.

[15] Suzuki, A. and Sato, Y. (2004) Comprehensive Gröbner Bases via ACGB. In Tran, Q-N.,editor, ACA2004, 65-73.

[16] Suzuki, A. and Sato, Y. (2006) A Simple Algorithm to compute Comprehensive Gröbner Bases using Gröbner bases. In ISSAC2006, 326-331.

[17] Wang, D.K. and Sun, Y. (2009) An Efficient Algorithm for Factoring Polynomials over Algebraic Extension Field. arXiv:0907.2300v1.

[18] Weispfenning, V. (1992). Comprehensive Gröbner bases, J. Symb. Comp. 14, 1-29.

[19] Weisphenning, V. (2003). Canonical Comprehensive Gröbner bases, J. Symb. Comp. 36, 669-683.