A new algorithm for computing \( \mu \)-bases of the univariate polynomial vector

Dingkang Wang\(^a,b\), Hesong Wang\(^a,b\), Fanghui Xiao\(^c\,\ast\)

\(^a\)KLMM, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China
\(^b\)School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China
\(^c\)College of Mathematics and Computer Science, Zhejiang Normal University, Jinhua, 321004, China

Abstract

In this paper, we characterized the relationship between Gröbner bases and \( \mu \)-bases: any minimal Gröbner basis of the syzygy module for \( n \) univariate polynomials with respect to the term-over-position monomial order is its \( \mu \)-basis. Moreover, based on the gcd computation, we construct a free basis of the syzygy module by the recursive way. According to this relationship and the constructed free basis, a new algorithm for computing \( \mu \)-bases of the syzygy module is presented. The theoretical complexity of the algorithm is \( O(n^3d^2) \) under a reasonable assumption, where \( d \) is the maximum degree of the input \( n \) polynomials. We have implemented this algorithm (MinGb) in Maple. Experimental data and performance comparison with the existing algorithms developed by Song and Goldman (2009) (SG algorithm) and Hong et al. (2017) (HHK algorithm) show that MinGb algorithm is more efficient than SG algorithm when \( n \) and \( d \) are sufficiently large, while MinGb algorithm and HHK algorithm both have their own advantages.

Keywords: \( \mu \)-basis, Gröbner basis, syzygy module, univariate polynomial

1. Introduction

Parametric equations and implicit equations are two basic representations for curves and surfaces. The process of converting a parametric equation into an implicit equation is called implicitization, and the inverse process is known as parameterization. There is plenty of research on the implicitization of rational curves and surfaces. Traditional implicitizing approaches include Gröbner bases, resultants and Wu’s characteristic set method, which have their own pros and cons, as well as applications (Becker et al., 1993; Buchberger, 1985, 1988; Canny, 1988; Chionh and Goldman, 1992, 1995a,b; Chionh et al., 2002; Cox et al., 2004, 2006; Manocha and Canny, 1992; Sederberg et al., 1984). In 1990s, a new technique called moving curves and surfaces was proposed to solve the implicitization problem (Cox et al., 2000; Sederberg and Chen, 1995; Sederberg and Saito, 1995; Sederberg et al., 1994, 1997; Zhang et al., 1999).

With this new technique, the concept of \( \mu \)-basis, which is defined as a special basis of the moving line ideal of planar rational curves, was firstly proposed by Cox et al. (1998). \( \mu \)-Bases are

\*Corresponding author

Email addresses: dwang@mmrc.iss.ac.cn (Dingkang Wang), wanghesong@amss.ac.cn (Hesong Wang), xiaofanghui@amss.ac.cn (Fanghui Xiao)
considered as the perfect combination of interpretability and effectiveness from the perspective of CAGD. From the perspective of interpretability, \( \mu \)-bases have an elegant geometric property that any planar rational curve of degree \( d \) is the intersection of its two \( \mu \)-basis elements whose degree sum up to \( d \). From the perspective of effectiveness, \( \mu \)-bases provide a compact representation for the implicit equation of a rational curve (Chen and Sederberg, 2002). To some extent, \( \mu \)-basis is an effective tool of connecting the parametric form and implicit form of a rational curve (Jia et al., 2018). In fact, \( \mu \)-bases’ algebraic counterparts are special syzygies of the parametric equations of rational curves or surfaces. In other words, a \( \mu \)-basis of a curve is a special basis of the syzygy module of the polynomials in the parametric equations.

Since the concept of \( \mu \)-basis was proposed, many researchers began to apply \( \mu \)-bases to various problems such as Jia and Goldman (2009); Song and Goldman (2009); Wang and Tesemma (2014). The development of algorithms for computing \( \mu \)-basis has been motivated by these emerging applications of \( \mu \)-bases. Cox et al. (1998) suggested an algorithm of computing a \( \mu \)-basis for a rational planar curve (its parametric equations correspond to 3 univariate polynomials) by computing two moving lines (corresponding to two syzygies) which satisfy the required properties. Zheng and Sederberg (2001) presented a different algorithm that is an automatic algorithm based on Buchberger-type reduction. Chen and Wang (2002) provided an improved algorithm for computing a \( \mu \)-basis. Deng et al. (2005) gave an efficient algorithm to compute the \( \mu \)-basis of a rational curve by using polynomial matrix factorization. These algorithms are for planar rational curves (3 univariate polynomials). Subsequently, the first algorithm (SG) for computing a \( \mu \)-basis of arbitrary \( n \) univariate polynomials was proposed by Song and Goldman (2009) as a generalization of the algorithm presented in (Chen and Wang, 2002). Recently, Hong et al. (2017) gave a new algorithm (HHK) for computing a \( \mu \)-basis of the syzygy module of \( n \) univariate polynomials. To be fair, SG algorithm and HHK algorithm achieved their own advances in performance.

Instead of computing \( \mu \)-bases directly by the above mentioned algorithms, in this paper we characterize a relationship between Gröbner bases and \( \mu \)-bases of the syzygy module for univariate polynomials: let \( \mathbf{a} = (a_1, \ldots, a_n) \) be a univariate polynomial vector, then any minimal Gröbner basis \( G \) of the syzygy module \( \text{syz}(\mathbf{a}) \) with respect to the term-over-position monomial order is a \( \mu \)-basis for \( \text{syz}(\mathbf{a}) \), but not vice versa. With this relationship, it is obvious that one can get a \( \mu \)-basis of the syzygy module \( \text{syz}(\mathbf{a}) \) by the Gröbner basis computation of the syzygy module. Generally, the Gröbner basis algorithm for syzygy modules starts from a generating set consisting of \( \binom{n}{2} \) trivial syzygies \( \{u_{ij} \mid 1 \leq i < j \leq n\} \) where \( u_{ij} = (0, 0, \ldots, -a_j, \ldots, a_i, \ldots, 0) \in \text{syz}(\mathbf{a}) \), provided \( \text{gcd}(\mathbf{a}) = 1 \). As is well known, the Gröbner basis approach is a powerful tool in algebraic system and algorithm efficiency is affected by the number of input elements. So to further optimize the algorithm for computing the Gröbner basis or \( \mu \)-basis, we focus on constructing a better generating set as the input of Gröbner basis algorithm. This construction is based on the gcd computation and the main idea is that, for any \( 2 \leq i \leq n - 1 \), if we have univariate polynomials \( a_{i-1}, \ldots, u_i \) such that \( u_i a_1 + \cdots + u_0 a_i = d_i \), where \( d_i = \text{gcd}(a_1, \ldots, a_i) \), then the polynomial vector \( \mathbf{v}_i = (u_1 a_{i+1}, \ldots, u_i a_{i+1}, -d_i, 0, \ldots, 0) \in \text{syz}(\mathbf{a}) \). Repeating this process at most \( n - 2 \) times, we get a set \( V = \{\mathbf{v}_1, \ldots, \mathbf{v}_{n-1}\} \), and prove that elements of the set divided by \( d_{i+1} \) form a free basis of the syzygy module. As a consequence, we present a new algorithm for computing a \( \mu \)-basis of the syzygy module. We analyze the theoretical computational complexity of the algorithm and show that the complexity is \( O(n^3d^2) \) under the assumption that \( a_1 \) and \( a_2 \) are relatively prime. Moreover, we have implemented this algorithm in the computer algebra system Maple and compared it with SG algorithm proposed by Song and Goldman (2009) and HHK algorithm.
proposed by Hong et al. (2017).

This paper is organized as follows. In Section 2, we introduce some correlative definition and lemmas on syzygy module, \( \mu \)-basis and Gröbner basis. In Section 3, we characterize the relationship between Gröbner basis and \( \mu \)-basis of the syzygy module for univariate polynomials. In Section 4, we firstly construct a generating set of the syzygy module and prove that it a free basis, then present the main algorithm for computing a \( \mu \)-basis. In Section 5, we analyze the complexity of the proposed \( \mu \)-basis algorithm and compare it with the existing \( \mu \)-basis algorithms proposed in (Song and Goldman, 2009) and (Hong et al., 2017) on computational performance. Section 6 is the conclusion of this paper.

2. Preliminaries

In this section, we shall present the definition of \( \mu \)-bases and Gröbner bases. The reader is referred to (Cox et al., 2006; Hong et al., 2017) for more details. Throughout this paper, \( K \) denotes a field, \( K[x] \) denotes a ring of univariate polynomials with the indeterminate \( x \) and \( K[x]^n \) denotes the set of \( n \)-dimensional row vectors with entries in the polynomial ring \( K[x] \), where \( n \) is a positive integer.

A subset of \( K[x]^n \) is called a module over \( K[x] \), if this subset is closed under addition and scalar multiplication by elements of \( K[x] \).

For a finite set of vectors \( f_1, \ldots, f_s \in K[x]^n \), we consider the set of all polynomial vectors in \( K[x]^n \) which can be written as a \( K[x] \)-linear combination of these vectors:

\[
M = \{ h_1 f_1 + \cdots + h_s f_s \in K[x]^n : h_i \in K[x] \text{ for } i = 1, \ldots, s \}.
\]

Then \( M \) is a submodule of \( K[x]^n \) generated by \( \{ f_1, \ldots, f_s \} \) and denoted by \( \langle f_1, \ldots, f_s \rangle \). The set \( \{ f_1, \ldots, f_s \} \) is called a generating set of \( M \). However, the generating set of \( M \) is not necessarily linearly independent. If \( \{ f_1, \ldots, f_s \} \) is \( k[x] \)-linearly independent, \( \{ f_1, \ldots, f_s \} \) is a basis of \( M \). Moreover, if a module \( M \) has a basis, then \( M \) is called a free module.

Given \( n \) univariate polynomials \( a_1, \ldots, a_n \in K[x] \), let \( a = (a_1, \ldots, a_n) \in K[x]^n \). Without loss of generality, we assume that \( a \) is a non-zero polynomial vector.

**Definition 1.** The set

\[
syz(a) = \{ u = (u_1, \ldots, u_n) \in K[x]^n : u \cdot a = u_1 a_1 + \cdots + u_n a_n = 0 \}
\]

is a module over \( k[x] \), and is called the syzygy module of \( a \).

**Remark 2.** Provided \( \gcd(a) = \gcd(a_1, \ldots, a_n) = h \). Let \( \overline{a} = a/h \), we have \( syz(a) = syz(\overline{a}) \).

There are two lemmas for the syzygy module. The details can refer to (Cox et al., 1998, 2006; Song and Goldman, 2009).

**Lemma 3.** Let \( a = (a_1, \ldots, a_n) \in K[x]^n \) with \( \gcd(a) = 1 \). Then \( syz(a) \) is generated by \( F = \{ u_{i,j} \} \) for \( 1 \leq i < j \leq n \), where \( u_{i,j} = (0, \ldots, 0, a_j, \ldots, a_i, \ldots, 0) \).

**Lemma 4.** Suppose \( a = (a_1, \cdots, a_n) \in K[x]^n \). Then the syzygy module \( syz(a) \) is a free module with \( n-1 \) generators.

Next we give the definition of \( \mu \)-basis. Before this, we introduce the following terminology.
Definition 5. The degree and the leading coefficient vector of $a = (a_1, \ldots, a_n) \in K[x]^n$ is defined as follows:

1. $\deg(a) = \max_{i=1, \ldots, n} \deg(a_i)$;
2. $\text{LV}(a) = (\text{coeff}(a_1, d), \ldots, \text{coeff}(a_n, d)) \in K^n$, where $d = \deg(a)$ and $\text{coeff}(a_i, d)$ denotes the coefficient of $x^d$ in $a_i$.

For example, $a = (-x^3 - 2x^2 + x + 1, x^3 + x^2 + 2x - 1, -3x)$, then $\deg(a) = 3$ and $\text{LV}(a) = (-1, 1, 0)$.

Definition 6. For a non-zero row vector $a \in K[x]^n$, a subset $U = \{u_1, \ldots, u_{n-1}\} \subset K[x]^n$ of polynomial vectors is called a $\mu$-basis of $a$, or equivalently, $\mu$-basis of the syzygy module $\text{syz}(a)$, if the following properties hold:

1. $\text{LV}(u_1), \ldots, \text{LV}(u_{n-1})$ are independent over $K$; and
2. $U$ is a free basis of $\text{syz}(a)$.

In this paper, we will also introduce a special generating set of modules: Gröbner basis.

As is well known, $K[x]^n$ is a free module with a standard basis $\{e_1, \ldots, e_n\}$, where $e_i$ is the unit vector. The monomial in $K[x]^n$ is of form $x^d$, where $d$ is a non-negative integer. Given a monomial order on $K[x]$, monomial orders on $K[x]^n$ can be obtained by extending it. In the following, we introduce two commonly used monomial order on $K[x]^n$. By convention, $e_1 < \cdots < e_n$.

Definition 7. Let $>$ be any monomial order on $k[x]$.

1. (POT) We say $z^\alpha e_i >_{POT} z^\delta e_j$ if $i > j$, or $i = j$ and $z^\alpha > z^\delta$.
2. (TOP) We say $z^\alpha e_i >_{TOP} z^\delta e_j$ if $z^\alpha > z^\delta$, or if $z^\alpha = z^\delta$ and $i > j$.

Fixing a monomial order $>$ on $k[x]^n$, for $g \in k[x]^n$, the leading term, leading coefficient, and leading monomial of $g$ are denoted by $\text{LT}(g)$, $\text{LC}(g)$, and $\text{LM}(g)$, respectively. Now we give the definition of Gröbner bases for modules.

Definition 8. Given a monomial order $>$ on $k[x]^n$, and let $M$ be a submodule of $k[x]^n$.

1. $\langle \text{LT}(M) \rangle$ denotes the monomial submodule generated by the leading terms of all $g \in M$ with respect to $>$. 
2. A finite set $G = \{g_1, \ldots, g_t\} \subset M$ is called a Gröbner basis for $M$ if $\langle \text{LT}(M) \rangle = \langle \text{LT}(g_1), \ldots, \text{LT}(g_t) \rangle$.

Further, the minimal Gröbner basis is defined as follows.

Definition 9. A Gröbner basis $G$ for $M \subset k[x]^n$ is said to be minimal, if $\text{LM}(g) \notin \text{LM}(G \setminus g)$ for all $g \in G$. 
3. The relationship between $\mu$-bases and Gröbner bases

As is mentioned before, there is a $\mu$-basis and a Gröbner basis for the syzygy module. Is there any relationship between them? What is the relationship between them? This is exactly what our research focuses on.

Based on such a thinking, we get the following theorem.

**Theorem 10.** Given a non-zero polynomial vector $\mathbf{a} \in K[x]^n$. A minimal Gröbner basis $G$ of $\text{syz}(\mathbf{a})$ with respect to the monomial order $\succ_{TOP}$ on $K[x]^n$ is a $\mu$-basis of $\text{syz}(\mathbf{a})$.

To prove the above theorem, we need the following lemma which can be found in (Hong et al., 2017).

**Lemma 11.** Given non-zero polynomial vectors $\mathbf{g}_1, \cdots, \mathbf{g}_s \in K[x]^n$, if $\text{LV}(\mathbf{g}_1), \cdots, \text{LV}(\mathbf{g}_s)$ are $K$-linearly independent, then $\mathbf{g}_1, \cdots, \mathbf{g}_s$ are $K[x]$-linearly independent.

With the above lemma, we give the proof of Theorem 10.

**Proof.** Suppose that $G = \{\mathbf{g}_1, \ldots, \mathbf{g}_s\}$ is a minimal Gröbner basis of $\text{syz}(\mathbf{a})$ with respect to the monomial order $\succ_{TOP}$. Since $G$ is minimal and $K[x]$ is a univariate polynomial ring, we have $\text{LM}(\mathbf{g}_k) = x^e \mathbf{e}_k$ and $\mathbf{e}_k \neq \mathbf{e}_l$ whenever $i \neq j$, then $s \leq n$. If we can prove that $\text{LV}(\mathbf{g}_1), \cdots, \text{LV}(\mathbf{g}_s)$ are $K$-linearly independent, then according to the above lemma, $\mathbf{g}_1, \cdots, \mathbf{g}_s$ are $K[x]$-linearly independent. In other words, $G$ is a free basis of $\text{syz}(\mathbf{a})$. Then $s = n - 1$ by Lemma 4. So with the definition of $\mu$-basis, $G$ is a $\mu$-basis of $\text{syz}(\mathbf{a})$. Thus, the remaining task is that $\text{LV}(\mathbf{g}_1), \cdots, \text{LV}(\mathbf{g}_s)$ are $K$-linearly independent.

Assume $\text{LV}(\mathbf{g}_1), \cdots, \text{LV}(\mathbf{g}_s)$ are $K$-linearly dependent, then there exist $c_1, \cdots, c_s \in K$ which are not all zero such that $c_1 \text{LV}(\mathbf{g}_1) + \cdots + c_s \text{LV}(\mathbf{g}_s) = \mathbf{0}$. Let $d = \max_{i=1, \ldots, s} \{ \deg(\mathbf{g}_i) \mid c_i \neq 0 \}$, $T = \max \{ \text{LM}(\mathbf{g}_i) \mid c_i \neq 0 \}$. With loss of generality, we assume $T = \text{LM}(\mathbf{g}_j)$ with $c_j \neq 0$. Then

$$\text{LV}(\mathbf{g}_j) = -\frac{1}{c_j} [c_1 \text{LV}(\mathbf{g}_1) + \cdots + c_{j-1} \text{LV}(\mathbf{g}_{j-1}) + c_{j+1} \text{LV}(\mathbf{g}_{j+1}) + \cdots + c_s \text{LV}(\mathbf{g}_s)].$$

By $\text{LM}(\mathbf{g}_i) = \text{LM}(\text{LV}(\mathbf{g}_i)x^{d_i})$, where $d_i = \deg(\mathbf{g}_i)$ for all $1 \leq i \leq s$, we get

$$\text{LM}(\mathbf{g}_i) = \text{LM}(\text{LV}(\mathbf{g}_i)x^{d_i}) = -\frac{1}{c_j} \text{LM}(c_1 \text{LV}(\mathbf{g}_1)x^{d_i}x^{d-d_1} + \cdots + c_{j-1} \text{LV}(\mathbf{g}_{j-1})x^{d-d_{j-1}}x^{d-d_{j-1}} + c_{j+1} \text{LV}(\mathbf{g}_{j+1})x^{d-d_{j+1}} + \cdots + c_s \text{LV}(\mathbf{g}_s)x^{d-d_s})$$

$$= -\frac{1}{c_j} (c_1 \text{LM}(\mathbf{g}_1)x^{d-d_1} + \cdots + c_{j-1} \text{LM}(\mathbf{g}_{j-1})x^{d-d_{j-1}} + c_{j+1} \text{LM}(\mathbf{g}_{j+1})x^{d-d_{j+1}} + \cdots + c_s \text{LM}(\mathbf{g}_s))x^{d-d_i},$$

which contradicts that each $\text{LM}(\mathbf{g}_i)$ contains the different standard basis vector $\mathbf{e}_s$. As a consequence, $\text{LV}(\mathbf{g}_1), \cdots, \text{LV}(\mathbf{g}_s)$ are $K$-linearly independent. \hfill \Box

**Remark 12.** $\mu$-bases of the syzygy module $\text{syz}(\mathbf{a})$ are not unique, but under the given module order, a reduced Gröbner basis of $\text{syz}(\mathbf{a})$ is unique, which is a minimal Gröbner basis.
It is well known that Grobner bases for modules are related to matrix canonical forms. For example, the reduced Grobner basis for the module of rows of a polynomial matrix using a term over position ordering is the same as the Popov normal form of the matrix. Beckermann et al. (1999, 2006) developed some efficient methods for computing a Popov normal form of a polynomial matrix. Neiger and Schost (2019) also proposed an algorithm of computing the reduced Gröbner basis of syzygy module.

Remark 13. Theorem 10 shows that a minimal Gröbner basis of the syzygy module \( \text{syz}(a) \) under the order \( \succ_{\text{TOP}} \) is a \( \mu \)-basis of \( \text{syz}(a) \). However, the converse of Theorem 10 is not true. That is, a \( \mu \)-basis of \( \text{syz}(a) \) isn’t necessarily a minimal Gröbner basis of \( \text{syz}(a) \). This can be confirmed by the following example.

Example 14. Given \( a = (x, x^2 + 1, x^3 - 1, x^4 + 6, x^2 + 1, -\frac{2}{7} x^3 - \frac{4}{7} x^2 - \frac{1}{7} x + \frac{2}{7}, x^4 + x^3 + 1, x^4 + 1) \in \mathbb{Q}[x]^7 \). By computation, we get a \( \mu \)-basis \( U = \{u_1, \cdots, u_6\} \) of \( \text{syz}(a) \), where

\[
\begin{align*}
    u_1 &= (1, 4, 6, 0, 3, -4, 4), \\
    u_2 &= (0, -1, -1, \frac{1}{6}, 0, 1, -7), \\
    u_3 &= (\frac{21}{2} x, -1, -2 x, -1, 3 - \frac{3}{2} x, 2 x, -2 x), \\
    u_4 &= (-2 x, \frac{2}{3} - \frac{2}{3} x, -\frac{1}{3} x, 0, -1, \frac{1}{3} x, -\frac{1}{3} x), \\
    u_5 &= (-x, x + 1, x, 0, 0, -1, 0), \\
    u_6 &= (-x, 1, 0, 0, 0, x, -x - 1).
\end{align*}
\]

However, under the order \( \succ_{\text{TOP}} \), \( \text{LM}(u_1) = \text{LM}(u_2) = e_7 \). It is obvious that \( U \) is not a minimal Gröbner basis of \( \text{syz}(a) \).

As above, the relationship between the \( \mu \)-basis and Gröbner basis of \( \text{syz}(a) \) is elaborated. Based on Theorem 10, we can get a \( \mu \)-basis by computing a minimal Gröbner basis of \( \text{syz}(a) \). As we all know, when given a generating set of \( \text{syz}(a) \), a minimal Gröbner basis of \( \text{syz}(a) \) can be obtained through existing algorithms for calculating Gröbner basis for the module.

4. The algorithm for computing the \( \mu \)-basis of \( \text{syz}(a) \)

From the above section, we know that in order to get \( \mu \)-basis by computing the minimal Gröbner basis, a generating set of the syzygy module \( \text{syz}(a) \) is needed. Generally, a generating set of the syzygy module \( \text{syz}(a) \) is given by the canonical method mentioned in Lemma 3. Such a generating set has \( \binom{n}{2} \) elements. It is easy to see that when \( n \) is large, this process of computing the Gröbner basis is very time-consuming. In this section, we focus on constructing a better generating set as the input of Gröbner basis (or \( \mu \)-basis) algorithm to improve the efficiency.

4.1. Constructing a free basis of \( \text{syz}(a) \)

Given \( n \) polynomials \( a_1, \cdots, a_n \in \mathbb{K}[x] \). When \( n = 2 \), \( (a_2, -a_1) \) is a syzygy of \((a_1, a_2)\). If \( \text{gcd}(a_1, a_2) = d_2 \), then \( \{(\frac{a_1}{d_2}, -\frac{a_2}{d_2})\} \) is a basis of \( \text{syz}(a_1, a_2) \). Obviously, when \( n = 3 \), \((a_2, -a_1, 0)\) is a syzygy of \((a_1, a_2, a_3)\). Can we use a recursive way to get the basis of \( \text{syz}(a_1, a_2, a_3) \) by means of the gcd computation?
Below we give the main ideas of the construction method. Assume that \( \gcd(a_1, a_2) = u_1a_1 + u_2a_2 = d_2 \), where \( u_1, u_2 \in K[x] \), then \( [(\frac{a_1}{d_2}, -\frac{a_2}{d_2})] \) is a basis of \( \text{syz}(a_1, a_2) \). For \( (d_2, a_3) \in K[x]^2 \), \( (a_3, -d_2) \) is a syzygy of \( \text{syz}(d_2, a_3) \). That is,

\[
a_3d_2 - d_2a_3 = 0 \implies a_3(u_1a_1 + u_2a_2) - d_2a_3 = 0 \implies a_3u_1a_1 + a_3u_2a_2 - d_2a_3 = 0.
\]

We get a new syzygy \( (a_3u_1, a_3u_2, -d_2) \) in \( \text{syz}(a_1, a_2, a_3) \). \( (a_2, -a_1, 0) \) and \( (a_3u_1, a_3u_2, -d_2) \) are linearly independent over \( K[x] \). Let \( \gcd(a_1, a_2, a_3) = \gcd(d_1, a_3) = d_3 \), then \( (\frac{a_1}{d_3}, -\frac{a_2}{d_3}, 0), \quad (\frac{a_3u_1}{d_3}, \frac{a_3u_2}{d_3}, -\frac{d_2}{d_3}) \) is a set of syzygies of \( (a_1, a_2, a_3) \). Later we will prove that it is a basis of \( \text{syz}(a_1, a_2, a_3) \). Go on and consider \( (d_3, a_4), \cdots, (d_{n-1}, a_n) \). According to this idea, we design an algorithm to obtain a free basis of \( \text{syz}(a) \).

Algorithm 1: The free basis algorithm of the syzygy module

**Input:** \( a = (a_1, \ldots, a_n) \in K[x]^n \), a non-zero polynomial vector.  
**Output:** \( U = \{u_1, \cdots, u_{n-1}\} \subset K[x]^n \), a free basis of \( \text{syz}(a) \).

1. \( d_2 := \gcd(a_1, a_2) \);
2. \( u_1 := (\frac{a_1}{d_2}, -\frac{a_2}{d_2}, 0, \ldots, 0) \);
3. for \( i \) from 2 to \( n - 1 \) do
4. \( d_{i+1} := \gcd(d_i, a_{i+1}) \);
5. \( \text{compute } u_1', \ldots, u_i' \in K[x] \text{ s.t. } d_i = \sum_{j=1}^{i} a_j u_j'; \)
6. \( u_i := \left( \frac{u_1'}{d_{i+1}}, \frac{u_2'}{d_{i+1}}, \frac{u_3'}{d_{i+1}}, \ldots, \frac{u_{i-1}'}{d_{i+1}}, -\frac{d_i}{d_{i+1}}, 0, \ldots, 0 \right) ; \)
7. return \( U = \{u_1, \cdots, u_{n-1}\} \).

For the step 5 in Algorithm 1, we can make use of the results from the previous time. If \( d_{i+1} = \gcd(d_i, a_{i+1}) = v_1d_i + v_2a_{i+1} \), and \( d_i = \sum_{j=1}^{i} a_j u_j' \), then

\[
d_{i+1} = v_1 \sum_{j=1}^{i} a_j u'_j + v_2 a_{i+1} = v_1 u'_1 a_1 + \cdots + v_1 u'_i a_i + v_2 a_{i+1}.
\]

So

\[
u_{i+1}^j = v_1 u'_j, \quad 1 \leq j \leq i;
\]

\[
u_{i+1}^i = v_2.
\]

Thus, we only compute \( d_{i+1} = \gcd(d_i, a_{i+1}) \) and it’s combination coefficients \( v_1, v_2 \) for \( 1 \leq i \leq n-1 \).

In addition, the matrix consisting of the transposition of \( u_1, \cdots, u_{n-1} \) is of the form:
where $\tau_1, \ldots, \tau_{n-1} \in K[x]$ are all non-zero polynomials. Obviously, it is a column-echelon matrix.

Now, we prove that Algorithm 1 is correct.

**Theorem 15.** Given a polynomial vector $a = (a_1, \ldots, a_n) \in K[x]^n$ as the input of Algorithm 1. If $U = [u_1, \ldots, u_{n-1}]$ is the output of the Algorithm 1, then $U$ is a minimal Gröbner basis of $\text{syz}(a)$ with respect to the monomial order $\succ_{\text{POT}}$. Further, $U$ is a free basis of $\text{syz}(a)$.

**Proof.** To prove that $U$ is a minimal Gröbner basis of $\text{syz}(a)$ under the monomial order $\succ_{\text{POT}}$, by the definition of Gröbner basis, it is only need to prove that for any $g = (g_1, \ldots, g_i, 0, \ldots, 0) \in \text{syz}(a)$ with $g_i \neq 0$, $u_{i-1}$ satisfies $\text{LM}(u_{i-1}) \nmid \text{LM}(g)$, where without loss of generality we assume that $a_i \neq 0$, and $2 \leq i \leq n$.

Suppose $\text{LM}(u_{i-1}) \nmid \text{LM}(g)$. Let

\[
    u_{i-1} = \frac{1}{d_i} (u^{i-1}_1 a_1, \ldots, u^{i-1}_n a_n, -d_{i-1}, 0, \ldots, 0),
\]

\[
    g = (g_1, \ldots, g_i, 0, \ldots, 0),
\]

where $d_{i-1} = \gcd(a_1, \ldots, a_{i-1}), d_i = \gcd(d_{i-1}, a_i)$ and $u^{i-1}_1, \ldots, u^{i-1}_n \in K[x]$ satisfying $\sum_{j=1}^{i-1} u^{i-1}_j a_j = d_{i-1}$. Then under the order $\succ_{\text{POT}}$, $\text{LM}(u_{i-1}) = \text{LM}(\frac{d_{i-1}}{d_i} e_i)$, and $\text{LM}(g) = \text{LM}(g_i) e_i$. By the assumption: $\text{LM}(u_{i-1}) \nmid \text{LM}(g)$, we have $\deg(\frac{d_{i-1}}{d_i} e_i) > \deg(g_i)$.

On the other hand, since $g = (g_1, \ldots, g_i, 0, \ldots, 0) \in \text{syz}(a)$, we have $g_1 a_1 + g_2 a_2 + \cdots + g_i a_i = 0$, which implies that $\frac{d_{i-1}}{d_i} (g_1 \frac{d_i}{a_1} a_1 + g_2 \frac{d_i}{a_1} a_1 + \cdots + g_i \frac{d_i}{a_1} a_1) = -g_i \frac{d_i}{a_i}$. Note that $\gcd(\frac{d_{i-1}}{d_i}, \frac{d_i}{a_i}) = 1$, so $\frac{d_{i-1}}{d_i} \mid g_i$, which contradicts $\deg(\frac{d_{i-1}}{d_i} e_i) > \deg(g_i)$. Therefore, $U$ is a minimal Gröbner basis of $\text{syz}(a)$ with respect to the monomial order $\succ_{\text{POT}}$. Further, $U$ is a free basis of $\text{syz}(a)$ because $u_1, \ldots, u_{n-1}$ are $K[x]$-linearly independent and $\text{syz}(a)$ is a free module with $n-1$ generators.

4.2. The proposed algorithm

Based on the above subsection, we are ready to formally give the algorithm for computing the $\mu$-basis of $\text{syz}(a)$.

The correctness of Algorithm 2 is guaranteed by Theorem 10 and Theorem 15, and the termination follows from the termination of Gröbner basis algorithm.

Moreover, we give a simple example that appeared in [28].

**Example 16.** Given $a = (x^4 + x^2 + 1, x^4 + x^3 + 1, x^4 + 1) \in \mathbb{Q}[x]^3$, compute a $\mu$-basis of $\text{syz}(a)$ by Algorithm 2.
Algorithm 2: The $\mu$-basis algorithm for univariate polynomials

Input : $a = (a_1, \ldots, a_n) \in K[x]^n$, a univariate polynomial vector.
Output: $U$, a $\mu$-basis of syz($a$).

begin
1 Compute a free basis of syz($a$) by calling Algorithm 1: $U' = \{u_1', \ldots, u_{n-1}'\}$;
2 $M := \langle U' \rangle$;
3 Compute a minimal Gröbner basis of the module $M$ w.r.t. the order $\succ_{\text{TOP}}$:
   $U = \{u_1, \ldots, u_{n-1}\}$;
4 return $U = \{u_1, \ldots, u_{n-1}\}$.

Step 1. Let

\[
\begin{align*}
a_1 &= x^4 + x^2 + 1, \\
a_2 &= x^4 + x^3 + 1, \\
a_3 &= x^4 + 1.
\end{align*}
\]

Now we compute a free basis of syz($a$) by calling Algorithm 1. First, we get
\[
\begin{align*}
d_2 &= \gcd(a_1, a_2) = v_1a_1 + v_2a_2 = 1, \\
d_3 &= \gcd(d_2, a_3) = 1.
\end{align*}
\]

where $v_1 = -\frac{1}{4}(2x^3 + 4x^2 + x - 2), v_2 = \frac{1}{4}(2x^3 + 2x^2 + x + 1)$. Since
\[
\begin{align*}
a_2a_1 - a_1a_2 &= a_2a_1 - a_1a_2 + 0 - a_3 = 0, \\
a_3d_2 - d_2a_3 &= a_3v_1a_1 + a_3v_2a_2 - d_2a_3 = 0,
\end{align*}
\]

we get a basis of syz($a$):
\[
U' = \{(a_2, -a_1, 0), (a_3v_1, a_3v_2, -d_2)\}
\]

\[
= \{(x^4 + x^3 + 1, -x^4 - x^3 - 1, 0), \left(-\frac{1}{3}(2x^3 + 4x^2 + x - 2)(x^4 + 1), \right.\}
\]

\[
\left.\frac{1}{3}(2x^3 + 2x^2 + x + 1)(x^4 + 1), -1\right)\}.
\]

Step 2. Regarding $U'$ as a generating set of the syzygy module syz($a$), call the Gröbner basis algorithm (‘GroebnerBasis’ command on the computer algebra system Maple) to compute a minimal Gröbner basis $U$ of syz($a$) under the monomial order $\succ_{\text{TOP}}$. The result is as follows:

\[
U = \{(-x, 1, x - 1), (-x^3 - 2x^2 - 2x + 1, x^3 + x^2 + 2x + 2, -3)\}.
\]

By Theorem 10, $U$ is a $\mu$-basis of syz($a$).

5. Complexity analysis and performance comparison

5.1. Theoretical complexity analysis

For arbitrary two polynomials, since the probability of that they are relatively prime is 1, without loss of generality, we will analyze the theoretical complexity of the $\mu$-basis algorithm under the assumption that $a_1, a_2$ are relatively prime.
The proposed $\mu$-basis algorithm in the previous section consists of two main parts: the computation of a free basis of $\text{syz}(a)$ and the computation of a minimal Gröbner basis of $\text{syz}(a)$ when given a set of generators. The algorithm involves the classical polynomial operations: polynomial multiplication and extended gcd of polynomials, which have the following complexity: $O(N \log N)$ and $O(N \log^2 N)$, respectively, where $N$ is the maximal degree of two univariate polynomials. For more details, we can refer to Pan (1992) and Moenck (1973).

We first estimate the complexity of the first part in Algorithm 2. In other words, we trace the theoretical complexity for computing a free basis of $\text{syz}(a)$ by means of the extended gcd computation.

Remark 17. Given $a = (a_1, \cdots, a_n) \in K[x]^n$. Let $U = \{u_1, \cdots, u_n-1\}$, a free basis of $\text{syz}(a)$, is computed by Algorithm 1. According the computation process and the assumption that $a_1, a_2$ are relatively prime, $d_2 = \gcd(a_1, a_2) = u_1a_1 + u_2a_2 = 1$ and for $2 \leq i < n, d_{i+1} = \gcd(d_i, a_i) = v_1d_i + v_2a_{i+1}$, where $v_1 = 1, v_2 = 0$. Thus, $u_1 = (a_2, -a_1, 0, \cdots, 0), u_i = (u_1a_{i+1}, u_2a_{i+1}, 0, \cdots, 0, -1, 0, \cdots, 0)$ with $-1$ in $(i+1)$-th position. That is, every $u_i$ satisfies that $\deg(u_i) < 2d, 2 \leq i \leq n-1$, where $d = \deg(a)$.

Proposition 18. The complexity of computing a free basis of $\text{syz}(a)$ by Algorithm 1 is $O(d \log^2 d + nd \log d)$.

Proof. According to Remark 17, the construction of $u_1, \cdots, u_n-1$ performs the extended gcd computation of $a_1$ and $a_2$, and $2(n-2)$ polynomial multiplications. Thus, the complexity of computing a free basis of $\text{syz}(a)$ by Algorithm 1 is

$$O(d \log^2 d + 2(n-2)d \log d) = O(d \log^2 d + nd \log d).$$

Next, we analyze the complexity of the second part in Algorithm 2 which computes a minimal Gröbner basis of $\text{syz}(a)$ under $>_{\text{TOP}}$ when given $U$ being a free basis of $\text{syz}(a)$. In order to analyze the complexity clearly and in view of the single-variable vector situation, the calculation process of the minimal Gröbner bases with Buchberger’s algorithm style is given as follows (see Cox et al. (2006)) for more details of Gröbner bases for modules.

1. Let $U := \{u_1, \cdots, u_{n-1}\}$ be the free basis of $\text{syz}(a)$;

2. While there is a nonzero S-vector in $U$ do:

   (a) find $1 \leq i, j \leq n-1$ such that $\text{LM}(u_j) = x^d \text{LM}(u_i)$;

   (b) update $u_j := S(u_j, u_i) = u_j - \frac{\text{LT}(u_j)}{\text{LT}(u_i)} u_i$, where $x^d = \deg(u_j) - \deg(u_i)$ and $S(u_j, u_i)$ is the S-vector of $u_j$ and $u_i$;

3. Return $U$ which is a minimal Gröbner basis of $\text{syz}(a)$.

It follows from Theorem 15 that a minimal Gröbner basis of $\text{syz}(a)$ with respect to $>_{\text{TOP}}$ is a $\mu$-basis of $\text{syz}(a)$. According to the property of $\mu$-bases that $\sum_{i=1}^{n-1} \deg(u_i) = \deg(a) - \deg(\gcd(a))$ and the above steps, in the process of computing a minimal Gröbner basis the sum of the degrees of elements in $U$ gradually decrease until it is equal to $\deg(a) - \deg(\gcd(a))$, and then remains unchanged which can be proved in the following.
Lemma 19. Let \( U = \{ u_1, \ldots, u_{n-1} \} \) be the free basis of \( \text{syz}(a) \) with \( \deg(u_1) \leq \deg(u_2) \leq \cdots \leq \deg(u_{n-1}) \). Then \( \sum_{i=1}^{n-1} \deg(u_i) = \deg(a) - \deg(\gcd(a)) \) if and only if for any set of generators \( v_1, v_2, \cdots, v_{n-1} \) of \( \text{syz}(a) \) with \( \deg(v_1) \leq \deg(v_2) \leq \cdots \leq \deg(v_{n-1}) \), \( \deg(u_i) \leq \deg(v_i) \) for \( i = 1, \ldots, n-1 \).

For the above lemma, the readers can refer to Song and Goldman (2009).

Lemma 20. Let \( U = \{ u_1, \ldots, u_{n-1} \} \) be the free basis of \( \text{syz}(a) \) with \( \deg(u_1) \leq \deg(u_2) \leq \cdots \leq \deg(u_{n-1}) \), and \( \sum_{i=1}^{n-1} \deg(u_i) = \deg(a) - \deg(\gcd(a)) \). For any \( u, u_j \in U \) with the nonzero S-vector \( S(u, u_j) \) under the monomial order \( >_{\text{TOP}} \), without loss of generality, suppose that \( 1 \leq i < j \leq n-1 \) and \( \text{LM}(u) = x^a e, \text{LM}(u_j) = x^d e \). Then \( \deg(u_j) = \deg(S(u, u_j)) \). Furthermore, \( \text{LM}(S(u, u_j)) = x^d e \) with \( s < t \).

Proof. Since \( S(u, u_j) = u_j \frac{\text{LT}(u)}{\text{LM}(u)} u \), let \( u'_j = S(u, u_j) \), then \( U' = \{ u_1, \ldots, u_{j-1}, u'_j, u_{j+1}, \ldots, u_{n-1} \} \) is still a free basis of \( \text{syz}(a) \) and \( \deg(u'_j) \leq \deg(u_j) \). Suppose \( \deg(u'_j) \leq \deg(u_k) \) with \( 1 \leq k \leq j \). Sorting the elements in \( U' \) in ascending order according to the degrees of elements, we have \( \deg(u_1) \leq \cdots \leq \deg(u_{j-1}) \leq \deg(u'_j) \leq \deg(u_k) \leq \cdots \leq \deg(u_{j-1}) \leq \deg(u_{j+1}) \leq \cdots \leq \deg(u_{n-1}) \).

However, it follows from Lemma 19 that
\[
\deg(u'_j) \geq \deg(u_k) \geq \deg(u_{k+1}) \geq \cdots \geq \deg(u_{j-1}) \geq \deg(u_j).
\]

Then \( \deg(u'_j) = \deg(u_k) = \cdots = \deg(u_{j-1}) = \deg(u_j) \). Thus \( \deg(u_j) = \deg(S(u, u_j)) \) and \( \text{LM}(S(u, u_j)) = x^d e \) with \( s < t \).

Proposition 21. The complexity of computing a minimal Gröbner basis of the module \( M = \langle U \rangle \) w.r.t. \( >_{\text{TOP}} \) is \( O(n^2d^2) \), where \( U = \{ u_1, \ldots, u_{n-1} \} \) is a free basis of \( \text{syz}(a) \).

Proof. According to the above analysis and the assumption, we divide the computation into two parts: the first is the decreasing process of the degrees’ sum of elements in \( U \) up to \( \deg(a) \), the second is the reduction of \( U \) in which the elements degrees’ sum is \( \deg(a) \) and unchanged to a minimal Gröbner basis of \( M = \langle U \rangle \) under \( >_{\text{TOP}} \). In the following, we will estimate the complexity of two parts.

1. By Remark 17, \( \deg(u_1) + \cdots + \deg(u_{n-1}) \leq d + 2d + \cdots + 2d = 2nd - 3d \). Since the degree decreasing process is until \( \sum_{i=1}^{n-1} \deg(u_i) = \deg(a) = d \) and for each \( u_i \) to reduce its degree by 1, at most \( n \) updates are required, the Buchberger-type reductions (i.e., “update” of \( u_i \)) by S-vector \( S(u_i, u_j) \) are performed at most \( n(2nd - 4d) \) times. For each update, the complexity is \( O(nd) \). Thus, the total complexity of this part appears to be \( O(n^2d^2) \).

2. Now \( U \) is a free basis of \( \text{syz}(a) \) with \( \sum_{i=1}^{n-1} \deg(u_i) = \deg(a) \). First, we sort the elements in \( U: u_1 \succ_{\text{TOP}} \cdots \succ_{\text{TOP}} u_{n-1} \), and group the elements according to the position indexes of their leading monomials. Suppose there are \( m \) groups. Obviously, \( n - 1 - m \) S-vectors need to be computed for updating \( u_1, \ldots, u_{n-2} \), where \( 1 \leq m \leq n - 1 \). The worst case is \( m = 1 \). By Lemma 20, after updating \( u_1, \ldots, u_{n-2} \) the position indexes of leading monomials all decrease, but their degrees are invariant.

If this process of updating continues until a minimal Gröbner basis is obtained, the number of S-vector needed to be computed is at most \( (n-2) + (n-3) + \cdots + 1 = \frac{(n-1)(n-2)}{2} \). Since the complexity of computing a S-vector is \( O(n^2d) \), the computational complexity of the second part is \( O\left(\frac{(n-1)(n-2)}{2}nd\right) = O(n^3d) \).
By summing up, the complexity of computing a minimal Gröbner basis of the module \( \text{syz}(a) \) w.r.t. \( \succ_{\text{TOP}} \) is \( O(n^3 d^2 + n^3 d) = O(n^3 d^2) \).

By Proposition 18 and 21, we get the complexity of Algorithm 2.

**Theorem 22.** The complexity of the Algorithm 2 is \( O(n^3 d^2) \).

**Remark 23.** In fact, Lemma 19 gives other characterizations of \( \mu \)-bases on the sum of the degrees and minimality of the degrees (Song and Goldman, 2009; Hong et al., 2017). Among them, the former (the sum of the degrees) is very useful for actual calculation. From the above analysis of the complexity of computing the minimal Gröbner basis for \( \text{syz}(a) \) under \( \succ_{\text{TOP}} \), we can see that one can obtain a \( \mu \)-basis before getting a minimal Gröbner basis since \( \sum_{i=1}^{n-1} \deg(u_i) = \deg(a) - \deg(\gcd(a)) \) is met in advance. In other words, in the ‘While’ loop step replacing the judgment condition “there is a nonzero S-vector in \( U \)” with “\( \sum_{i=1}^{n-1} \deg(u_i) \neq \deg(a) - \deg(\gcd(a)) \)”, a \( \mu \)-basis of \( \text{syz}(a) \) is output. But since this article focuses on the relationship between \( \mu \)-bases and Gröbner bases, the algorithm based on the sum of the degrees (i.e., the idea of replacing judgment condition) will not included here. Note that according to the proof of Proposition 21, this does not affect the total complexity.

### 5.2. Implementation and performance comparison

In the above subsection, we give a estimate of complexity of our proposed \( \mu \)-basis algorithm, which is \( O(n^3 d^2) \). In this section, we will compare our proposed \( \mu \)-basis algorithm (MinGb algorithm) with the existing efficient \( \mu \)-basis algorithms: SG algorithm proposed by Song and Goldman (2009) with complexity \( O(dn^5 + d^2 n^4) \) and HHK algorithm proposed by Hong et al. (2017) with complexity \( O(d^2 n + d^3 + n^2) \). We have implemented our algorithm in the computer algebra system Maple. The codes and examples are available on the web: [http://www.mmrc.iss.ac.cn/~dwang/software.html](http://www.mmrc.iss.ac.cn/~dwang/software.html).

In addition, we explain the experimental timing which corresponds to a point \((d, n, t)\), where \( d \) is the degree, \( n \) is the length of the input polynomial vector and \( t \) is the timing in seconds obtained by running the programs on Apple iMac (Intel i7, 4GHz, 32GB) using Maple 2018 version.

Now we give the fitted figures of experimental data obtained by running three algorithms (MinGb, SG and HHK) in Maple. The fitting model is based on the theoretical complexities of three algorithm, which was computed using least squares. The chosen three models for the timing are:

\[
\begin{align*}
I_{\text{MinGb}} &\approx 10^{-9} \cdot (4.1 d^2 n^3) \\
I_{\text{SG}} &\approx 10^{-8} \cdot (32.1 dn^5 + 4.4 d^2 n^4) \\
I_{\text{HHK}} &\approx 10^{-6} \cdot (10.7 d^2 n + 1.0 d^3 - 96.8 n^2)
\end{align*}
\]

The following figures show actual performance comparison of these algorithms.

In our experiments, the inputs are randomly-generated. For various values of \( n \) and \( d \), the corresponding polynomial is obtained by the command ‘randpoly’ on Maple. Moreover, to make the experimental data more reliable, we ran each algorithm several times on the same input and computed the average of the running time. The execution of a program on an input was cut off if its computing time exceeded 120 seconds. For each figure, the axes represent the range of values \( d = 3, \ldots, 200 \), \( n = 3, \ldots, 200 \) and \( t = 0, \ldots, 120 \).
As can be seen from the figures that for enough large $n$ and $d$, MinGb algorithm vastly outperforms SG algorithm. In fact, for most value of $n$, SG algorithm usually can’t terminate under 120 seconds. Besides, for a fixed $n$ (especially for a relatively small $n$), as $d$ increases MinGb algorithm outperforms HHK algorithm. In other words, HHK algorithm is more sensitive to the enlargement of $d$ than the MinGb algorithm. In contrast, for a fixed $d$, HHK algorithm is faster than MinGb algorithm when $n$ is sufficiently large. In addition, it’s not hard to see that these algorithms’ actual performance is conform to the theoretical complexity, which implies that our assumption that $a_1$ and $a_2$ are relatively prime is reasonable.

6. Conclusion

In the paper, the relationship between Gröbner bases and $\mu$-bases are characterized. For univariate polynomial vector $a$, any minimal Gröbner basis $G$ of the syzygy module $\text{syz}(a)$ with respect to $\succ_{\text{TOP}}$ is a $\mu$-basis of $\text{syz}(a)$. Based on this relationship, the $\mu$-basis of $\text{syz}(a)$ is obtained by Gröbner basis algorithm for modules. To improve the efficiency of this method, we construct a free basis of $\text{syz}(a)$ based on the gcd computation, which has successfully reduced the number of a generating set as the input of from $\binom{n}{2}$ to $n - 1$. As a consequence, a new algorithm for computing $\mu$-bases is presented. By analysis, we show that this algorithm has theoretical complexity $O(n^3d^2)$. We have implemented the algorithm and compared it with the existing efficient $\mu$-basis algorithms in (Song and Goldman, 2009; Hong et al., 2017). Experiments indicate that for enough large $n$ and $d$, the proposed algorithm vastly outperforms SG algorithm, while it and HHK algorithm have their own advantages.

Following the main result, whether a reduced Gröbner basis of the syzygy module $\text{syz}(a)$ as a special $\mu$-basis has special geometric meanings is worth exploring for rational curves. Moreover, the research of the relationship between minimal Gröbner bases and $\mu$-basis on rational surfaces is also our future researching work.
Acknowledgments

This research was supported in part by the CAS Key Project QYZDJ-SSW-SYS022.

References

Becker, T., Kredel, H., Weispfenning, V., 1993. Gröbner bases: a computational approach to commutative algebra. Springer-Verlag, London.
Beckermann, B., Labahn, G., Villard, G., 1999. Shifted normal forms of polynomial matrices. In: Proceedings of the 1999 international symposium on Symbolic and algebraic computation. pp. 189–196.
Beckermann, B., Labahn, G., Villard, G., 2006. Normal forms for general polynomial matrices. Journal of Symbolic Computation 41 (6), 708–737.
Buchberger, B., 1985. Gröbner bases: an algorithmic method in polynomial ideal theory. Multidimensional systems theory.
Buchberger, B., 1988. Applications of Gröbner basis in non-linear computational geometry. Trends in Computer Algebra 296, 52–80.
Canny, J., 1988. The complexity of robot motion planning. Doctoral dissertation, MIT, Cambridge, Massachusetts.
Chen, F., Sederberg, T., 2002. A new implicit representation of a planar rational curve with high order singularity. Computer Aided Geometric Design 19 (2), 151–167.
Chen, F., Wang, W., 2002. The -basis of a planar rational curve-properties and computation. Graphical Models 64 (6), 368–381.
Chionh, E.-W., Goldman, R. N., 1992. Implicitizing rational surfaces with base points by applying perturbations and the factors of zero theorem. In: Mathematical methods in computer aided geometric design II. Elsevier, pp. 101–110.
Chionh, E.-W., Goldman, R. N., 1995a. Elimination and resultants, part I: elimination and bivariate resultants. IEEE Computer Graphics and Applications 15 (1), 69–77.
Chionh, E.-W., Goldman, R. N., 1995b. Elimination and resultants, part II: Multivariate resultants. IEEE computer graphics and applications 15 (2), 60–69.
Chionh, E.-W., Zhang, M., Goldman, R. N., 2002. Fast computation of the Bezout and Dixon resultant matrices. Journal of Symbolic Computation 33 (1), 13–29.
Cox, D., Goldman, R., Zhang, M., 2000. On the validity of implicitization by moving quadrics for rational surfaces with no base points. Journal of Symbolic Computation 29 (3), 419–440.
Cox, D. A., Little, J., O’Shea, D., 2004. Ideals, Varieties, and Algorithms. Springer.
Cox, D. A., Little, J., O’Shea, D., 2006. Using algebraic geometry. Vol. 185. Springer Science and Business Media.
Cox, D. A., Sederberg, T. W., Chen, F., 1998. The moving line ideal basis of planar rational curves. Computer Aided Geometric Design 15 (8), 803–827.
Deng, J., Chen, F., Shen, L., 2005. Computing -bases of rational curves and surfaces using polynomial matrix factorization. In: Proceedings of the 2005 international symposium on Symbolic and algebraic computation. pp. 132–139.
Hong, H., Hough, Z., Kogan, I. A., 2017. Algorithm for computing -bases of univariate polynomials. Journal of Symbolic Computation 80, 844–874.
Jia, X., Goldman, R., 2009. -bases and singularities of rational planar curves. Computer Aided Geometric Design 26 (9), 970–988.
Jia, X., Shi, X., Chen, F., 2018. Survey on the theory and applications of -bases for rational curves and surfaces. Journal of Computational and Applied Mathematics 329, 2–23.
Manocha, D., Canny, J. F., 1992. Algorithm for implicitizing rational parametric surfaces. Computer Aided Geometric Design 9 (1), 25–50.
Moenck, R. T., 1973. Fast computation of GCDs. In: Proceedings of the fifth annual ACM symposium on Theory of computing. pp. 142–151.
Neiger, V., Schost, E., 2019. Computing syzygies in finite dimension using fast linear algebra. arXiv preprint arXiv:1912.01848.
Pan, V., 1992. Complexity of computations with matrices and polynomials. SIAM review 34 (2), 225–262.
Sederberg, T., Goldman, R., Du, H., 1997. Implicitizing rational curves by the method of moving algebraic curves. Journal of Symbolic Computation 23 (2-3), 153–175.
Sederberg, T. W., Anderson, D. C., Goldman, R. N., 1984. Implicit representation of parametric curves and surfaces. Computer Graphics, and Image Processing 28 (1), 72–84.
Sederberg, T. W., Chen, F., 1995. Implicitization using moving curves and surfaces. In: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques. pp. 301–308.
Sederberg, T. W., Saito, T., 1995. Rational-ruled surfaces: implicitization and section curves. Graphical Models and Image Processing 57 (4), 334–342.
Sederberg, T. W., Saito, T., Qi, D., Klimaszewski, K. S., 1994. Curve implicitization using moving lines. Computer Aided Geometric Design 11 (6), 687–706.
Song, N., Goldman, R., 2009. µ-bases for polynomial systems in one variable. Computer Aided Geometric Design 26 (2), 217–230.
Wang, H., Tesemma, M., 2014. Intersections of rational parametrized plane curves. European Journal of Pure and Applied Mathematics 7 (2), 191–200.
Zhang, M., Chionh, E.-W., Goldman, R. N., 1999. On a relationship between the moving line and moving conic coefficient matrices. Computer aided geometric design 16 (6), 517–527.
Zheng, J., Sederberg, T. W., 2001. A direct approach to computing the µ-basis of planar rational curves. Journal of Symbolic Computation 31 (5), 619–629.