Compact and efficient implicit representations

Clément Laroche[⋆]

National and Kapodistrian University of Athens
Department of Informatics and Telecommunication
claroche@di.uoa.gr

Abstract. In the perspective of manipulating geometric objects, there exists two main representations of curves and surfaces: parametric and implicit representations. Both are useful for different purposes and thus complement each other. Parametric representations are efficient in sampling points on an object; implicit representations are efficient in determining whether a point belongs to an object or not. Because of that, having both representations of the same objects at the same time maximizes the range of operations one can do with geometric objects. Switching from one representation to another is not an easy task. It usually requires the use of algebraic properties. Thus, there is a strong link between algebra and geometry, symbolised by the algebraic varieties: they are geometric objects described by an algebraic structure.
This thesis explores new kinds of implicit representations and algorithms for computing implicit representations. We show that different methods are adapted to different situations even when it comes to the choice of an implicit representation amongst several possibilities. Space curves can thus be described implicitly by conical surfaces, moving lines and/or moving quadrics. . . each description having different geometrical properties and practical usage. As there is not one implicit representation or implicitization algorithm that would be the best in any situation, we develop methods that fit to different kinds of informations known about the object we want to represent. As we show, objects constructed by sweeping a rigid body can be represented using the knowledge of that nature. Similarly, very particular curves may have a complicated algebraic structure. Depending on our tolerance to approximation, such curves can thus be perturbed to simplify greatly their algebraic structure or, on the contrary, be represented by a rich implicit representation format.

Keywords: Algebraic geometry · Implicitization · CAGD-CAE · Resultants · Syzygies

1 Dissertation Summary

We are interested in the representation of geometric objects, such as surfaces and curves. They can be stored in computers by different types of informations;

[⋆] Dissertation Advisor: Ioannis Z. Emiris, Professor the representation method then determines how these informations should be interpreted. For example, Bézier curves and patches form a basis of simple shapes that can be combined to have more complex shapes in Computer-Aided Geometric Design (CAGD). Even simpler: polygonal surfaces can be combined in order to for a polygonal mesh of a 3D object, which is useful in Computer-Aided Engineering (CAE). Also useful for CAE, Algebraic Geometry defines a shape as the zero set of polynomial or rational equation(s). In bitmap imaging, objects are described pixel by pixel. And so on...

Of course, an object representable by one representation method may not be representable by another method. An elliptic curve, for instance, can be described as the zero set of a degree 3 polynomial but it cannot be described using Bézier curves or rational parameterizations. On the other hand, a large set of objects can be represented using several different representation methods. An important problem is then to be able to pass from a representation to another. The difficulty to convert an object from one representation to another relates with the diversity of these representation methods.

Amongst representation methods, two kinds are of greater importance:

- Parametric representations, taking one or several parameters as input and a point of the object as output. Typically, a parametric representation of an algebraic object is a birational map between a projective space and that object.
- Implicit representations, taking a point of the ambient space as input and outputting whether that point belongs to the object or not. Typically, an implicit representation is a set of polynomials that simultaneously vanish on the object.

The work presented in this thesis is about (1) describing interesting ways to represent algebraic objects with a focus on implicit representations and (2) developing algorithms to switch from a representation to another with a focus on implicitization, i.e. algorithms that output an implicit representation.

1.1 Interest in applications and previous work

Although 2D and 3D objects are the core objects of the study, the algebraic tools developed can be applied to other situations, in particular when higher dimensional spaces are involved. The big picture is that parametric representations are useful when sampling and object while implicit representations are useful when looking for the relative position of a given point with respect to the object. They are complementary. Having both representations of a single object is usually the best way to go, thus the need of implicitization algorithms for building an implicit representation associated with a parametric representation.

Both representations allow rendering algorithms. Raytracing methods for rendering implicitly-represented objects can handle reflections and lightning accurately, which makes them suitable for very high-quality rendering. However, the speed when displaying parametrically-represented objects is outmatched and can make the difference between a real-time rendering and a non-real-time rendering. Because of the speed advantage, designers use parametric representations to produce free-form objects very easily in practice. Then, if needed, other representations must be computed. For instance, CAE engineers need another represen-

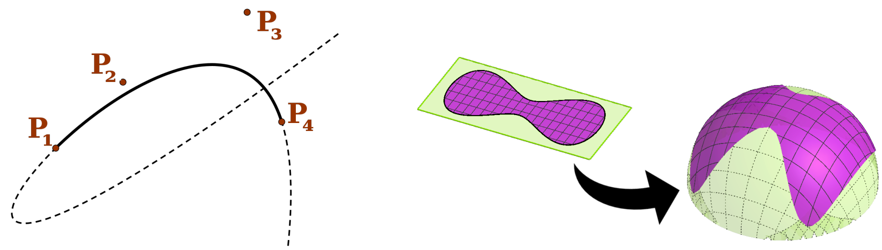


Fig. 1: Problems with global (implicit) representations: trimming a Bézier curve to its control polygon may keep unwanted parts of the global curve (*left*), trimming on the parametric space cannot be carried to a global representation, unless an inverse map from the ambient space to the parametric space is available (*right*)

tation (implicit or mesh) in order to compute the objects' hardness, flexibility or, more generally, its physical properties and behaviour (which consists in solving partial differential equations most of the time).

On top of that, a lot of 3D objects are represented using polygonal meshes (triangular meshes for most of it). These can be considered both as parametric representations or implicit representations since it only consists of linear surfaces, from which both representations are immediate to construct. Meshes are more than enough to represent very simple objects (i.e. with flat surfaces). They are not efficient any more when it comes to represent curved shapes, as many polygons are required for them for a result that is not as smooth as what we could ask for. We need to use more accurate representations when meshes are not a satisfying solution, and thus have algorithms for manipulating them and switch from a representation to another with the minimal loss of precision.

The algorithm of marching cubes (see [14]) can be used to compute a mesh out of an implicit representation. Using it, an approximation of implicitly-represented objects can be displayed relatively fast, though slower than when a parametric representation is directly available.

One of the biggest issues with parametric representations is the fact that the intersection of two rationally parameterized objects (typically two surfaces) are not necessarily rational varieties (the intersection curve(s) cannot be rationally parameterized). This is a huge problem to deal with in CAE and in particular when switching back and forth from parametric representations produced by CAGD to implicit representations or meshes required in CAE.

Implicit matrix representations give a solution to that problem and are a fitting way to represent a wide range of varieties. Instead of considering a variety as the set of common zeros of implicit polynomials, it is described as the set of points dropping the rank of a formal matrix. A square matrix containing formal coordinates thus represents an hypersurface; the drop of rank property is equivalent to the vanishing of the determinant (which is then the variety's unique defining polynomial). For varieties of lower dimensions, a rectangular

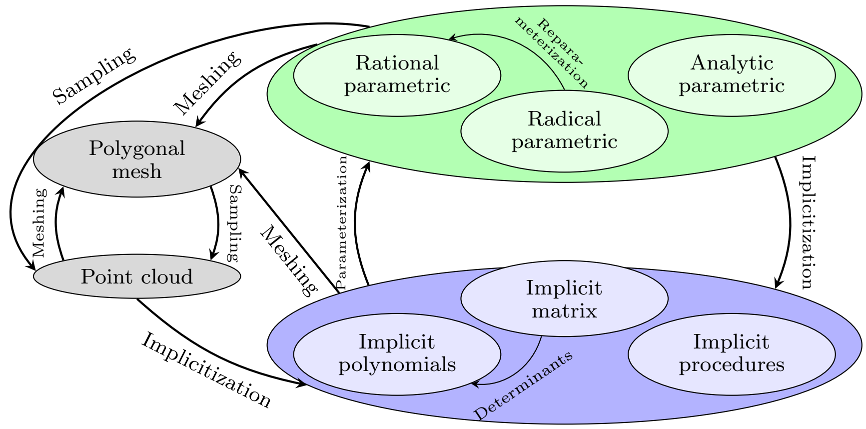


Fig. 2: Different representations of varieties

matrix can be used and the drop of rank property is equivalent to the vanishing of all of that matrix's largest minors.

There exists plenty of parameterization and implicitization algorithms. When it comes to exact implicitization, there are traditionally two frameworks: Gröbner bases and resultants. Gröbner bases were invented, as the name suggests not, independently by Hironaka in 1964[11] (while proving that any variety in characteristic 0 admits a resolution of singularities) and Buchberger (Gröbner's student) in 1965. They play only a little role in the work presented in this thesis because we rather use resultants or syzygies. Resultants have been introduced by Sylvester in 1840[18]. They can be used to determine whether $d+1$ polynomials in d variables share a common root. Resultants are both polynomials and often described as a matrix's determinant: they are thus highly algebraic objects. The use of syzygies in implicitization algorithms is a rather new field of research. Sederberg and Chen described the grounds for such algorithms in 1995[15]. Busé and Luu Ba have pushed the method further since 2009[2].

In this thesis approximate implicitizations are also a concern, although a minor one. When doing approximate implicitization, a lot of problems that arise at non-generic situations can be ignored or bypassed. Thus, the use of algebra is much less prominent.

1.2 Summary and contributions of the thesis

This thesis presents results and algorithms in the field of implicit representations and implicitization algorithms.

The author has followed Ph.D. studies at the National and Kapodistrian University of Athens in Greece in the framework of the project ARCADES funded by the Marie Skłodowska-Curie Actions. As part of this studies, the author has spent 3 months at the research centre SINTEF in Oslo, Norway, and 4 months at the research centre RISC Software GmbH in Hagenberg, Austria. In both occasions, work has been done collaboratively with the local research teams.

During his stay at SINTEF, the author implemented in C++ a sparse resultant algorithm based on an existing Maple implementation[7]. Although C++ is globally faster than Maple, it lacks of standard mathematical libraries (for instance, *newmat* or *Eigen* are two of the many existing matrix libraries, both having advantages and drawbacks). This implementation relies on the matrix library *newmat* and uses its own implementation of polynomials and polynomial operations.

During his stay at RISC, the author developed in C++ a new implicitization algorithm for a special kind of 3D objects: swept volumes. Those are volumes generated by an object following a time-dependent 3D rigid transformation. The swept volume itself is the union of all the points in space that come in contact with that base object at some point in time. Several works have been done regarding the implicitization of swept volumes; some of them require the base object to be polyhedral (such as [19]). The algorithm developed here is very flexible in the sense that it accepts many kinds of representations of the base object as input (and not only polyhedral ones) and take advantage of it in order to generate an implicit representation of the swept volume.

Another, more theoretical contribution, is the design and implementation in Maple of a new implicitization algorithm of varieties of codimension strictly greater than 1. The idea behind that algorithm comes from the theory of Chow forms. We prove that, using this algorithm, we can always represent space curves with 3 implicit equations. While it is known since a long time that there always exists 3 equations defining a space curve, many algorithms do not reach that lower bound and output more than 3 equations. This algorithm has been published as the article [8].

Finally, our last contribution is a new matrix-based implicitization algorithm. This algorithm relies on syzygies and chain complexes. It is a continuation of the work described in [2], done in collaboration with L. Busé and F. Yıldırım from INRIA Sophia-Antipolis, France. It provides a very strong link between parametric and implicit representations, allowing to reverse the map between the parametric and ambient spaces through the implicit matrix that it produces. The improvement of our new implicit matrix compared to [2] is that our matrices are more compact and faster to use. This amelioration comes with the drawbacks of being slower to compute (however, computing the matrix is done once while using it is done many times) and are slightly less easy to manipulate (because of its compactness, singular points of high degree may be uneasy to revert, that is to find all of its parameterization's preimages). This algorithm has been published as the article [3].

2 Results and Discussion

2.1 Chow Form Algorithm

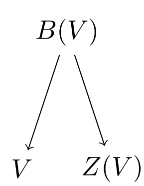
Chow forms have been studied in computer algebra, in particular for varieties of codimension $\geqslant 2$, since they provide a method to describe the variety by a single polynomial [5,10]. The Chow form of a variety V is basically a polynomial R_V which indicates when linear subspaces intersect V. For example, the Chow form of a space curve in projective 3-dimensional space is a polynomial in the indeterminates u_{ij} that vanishes whenever the planes

$$\begin{aligned} H_0 &= u_{00}x_0 + u_{01}x_1 + u_{02}x_2 + u_{03}x_3 = 0, \\ H_1 &= u_{10}x_0 + u_{11}x_1 + u_{12}x_2 + u_{13}x_3 = 0, \end{aligned} \tag{1}$$

intersect on the curve. If the space curve is given parametrically, the Chow form represents the variety in terms of R_V. It can be computed by a *symbolic resultant* of the system of linear equations (1) where the set of variables $X = (x_i)_i$ is substituted with the parametric equations; the resultant eliminates the parameters and yields a polynomial in the variables $U = (u_i)_i$. The implicit hypersurfaces in X containing the variety have to be extracted through *rewriting rules*. These make implicitization algorithms that rely on the computation of R_V impractical for varieties of high degree and/or dimension.

Due to their complexity, very few implementations exist for computing the Chow forms themselves. Amongst them, [17] is an implementation in Macaulay2, based on the formula of the Chow form in the Grassmannian space using the Plücker coordinates, and [12, Subroutine 7] is an algorithm using polynomial ring tools and based on a Poisson-like formula of the Chow form.

To formally define the Chow form let $Gr(k+1, n+1)$ denote the Grassmannian space of k-dimensional linear projective subspaces of $\mathbb{P}^n$. For a variety $V \subset \mathbb{P}^n$ of codimension c, let $B(V) \subset \mathbb{P}^n \times Gr(c, n+1)$ be the set of (P, L) such that P belongs both to V and to the projective linear subspace L of dimension $c-1$. Then we obtain V by forgetting the second component in $B(V)$ and we obtain an hypersurface $Z(V) := \{L \in Gr(c, n+1) \mid L \cap V \neq \emptyset\}$ of the Grassmannian space $Gr(c, n+1)$ by forgetting the first component in $B(V)$.



$Z(V)$ is called the *Chow variety* of V and has the advantage of being an hypersurface in the Grassmannian space, so it is determined by a unique implicit equation up to a constant factor: the Chow form R_V. Despite being determined by a unique equation, $Z(V)$ describes the variety V of unconstrained (co)dimension, see Proposition 1. Note that when V is a variety of codimension

1, we have $Z(V) \simeq V$; this explains why the theory of Chow form is effective only for codimension $c > 1$. On the other hand, the Chow form of a zero-dimensional variety $V = \{v_1, \ldots, v_k\}$ is also known as the u-resultant.

Definition 1. *Let $V \subset \mathbb{P}^n$ be a d-dimensional irreducible variety and $H_0, \ldots, H_d$ be linear forms where*

$$H_i = u_{i0}x_0 + \cdots + u_{in}x_n, \quad i = 0, \ldots, d \tag{2}$$

and u_{ij} are new variables, $0 \leqslant i \leqslant d$, $0 \leqslant j \leqslant n$. The Chow form R_V of V is a polynomial in the variables u_{ij} such that

$$R_V(u_{ij}) = 0 \Leftrightarrow V \cap \{H_0 = 0, \ldots, H_d = 0\} \neq \emptyset.$$

The intersection of the $d+1$ hyperplanes H_i defined in equation (2) is generically a $(n-d-1)$-dimensional linear subspace L of $\mathbb{P}^n$, i.e., an element of the Grassmannian $Gr(n-d, n+1) = Gr(c, n+1)$, where c is the codimension of V.

Proposition 1. [10, Prop.2.5,p.102] *A d-dimensional irreducible subvariety $V \subset \mathbb{P}^n$ is uniquely determined by its Chow form. More precisely, a point $\xi \in \mathbb{P}^n$ lies in V if and only if any $(n-d-1)$-dimensional plane containing ξ belongs to the Chow variety $Z(V)$ defined by R_V.*

Suppose that V is a space curve homogeneously parameterized as

$$x_j = f_j(t), \quad j = 0, \ldots, 3, \quad t = (t_0 : t_1).$$

Let the line L be defined by a symbolic point $\xi = (\xi_0 : \cdots : \xi_3)$ and a sufficiently generic point $G \notin V$. Define two planes $\mathrm{Aff}(G, \xi, P_0)$ and $\mathrm{Aff}(G, \xi, P_1)$ that intersect along L, by choosing two random points P_0 and P_1 and let $H_0(x_0 : \cdots : x_3)$ and $H_1(x_0 : \cdots : x_3)$ be their respective implicit equations, as in (1). The coefficients of H_0 and H_1 are now *linear polynomials* in ξ. The (homogeneous) Sylvester resultant of this system, where we set $x_j = f_j(t)$, eliminates t and returns a polynomial in ξ which vanishes on V (but not only on V), thus offering a necessary but not sufficient condition.

Lemma 1. *Let $\delta = \deg f_j(t)$, $j = 0, \ldots, 3$ and R_G be the Sylvester resultant of*

$$H_0(f_0(t) : \ldots : f_3(t)), \quad H_1(f_0(t) : \ldots : f_3(t)), \tag{3}$$

where H_0, H_1 are defined as above. Then R_G is of degree 2δ and factors into:

1. *a degree δ polynomial defining the conical surface $\mathcal{S}_V^G \supset V$ of vertex G and directrix V,*
2. *and a polynomial E_L^δ, where E_L is a linear polynomial defining the plane passing through points G, P_0, P_1.*

Theorem 1. *Let $f : \mathbb{P}^1 \to \mathbb{P}^3$, be a homogeneous parameterization of a space curve V and $\mathcal{S}_V^{G_k}$, $k = 1, 2, 3$ be three conical surfaces obtained by the method above with 3 different random points $G_k \notin V$. We distinguish two cases.*

1. *If V is not planar and the points G_k are not collinear, then V is the only 1-dimensional component of $\mathcal{S}_V^{G_1} \cap \mathcal{S}_V^{G_2} \cap \mathcal{S}_V^{G_3}$.*
2. *If V is contained in a plane $\mathcal{P}$ and if G_1 is not in $\mathcal{P}$, then $V = \mathcal{P} \cap \mathcal{S}_V^{G_1}$.*

This gives an implicitization algorithm of space curves that output either 2 or 3 implicit polynomials of degree δ. In the complete thesis, we generalise this algorithm to varieties of any codimension $c > 1$ embedded in spaces of any dimension $\mathbb{P}^n$.

2.2 Moving Quadrics Algorithm

In what follows, we suppose that an homogeneous parameterization of a rational curve $\mathcal{C} \subset \mathbb{P}^n$, $n \geq 2$, is given over a field $\mathbb{K}$ by

$$\begin{aligned} \phi : \quad \mathbb{P}^1 \quad &\to \mathbb{P}^n \\ (s:t) &\mapsto (f_0(s,t) : f_1(s,t) : \cdots : f_n(s,t)), \end{aligned} \tag{4}$$

where $f_0, \ldots, f_n$ are homogeneous polynomials in $\mathbb{K}[s,t]$ of the same degree $d \geq 1$. For the sake of simplicity we assume without loss of generality that these polynomials have no common factor, so that the map ϕ is well defined everywhere on $\mathbb{P}^1$.

Unlike the case of plane curves, if $n \geq 3$ a single polynomial equation in $\mathbb{K}[x_0, \ldots, x_n]$ is not sufficient to describe implicitly the curve $\mathcal{C}$. Such an equation describe an hypersurface in $\mathbb{P}^n$ and hence a collection of at least $n-1$ of them are necessary for characterizing a curve by a dimension argument, and in general more than $n-1$ equations are needed. To be more precise, consider the ring morphism

$$\begin{aligned} \mathbb{K}[x_0, \ldots, x_n] &\to \mathbb{K}[s,t] \\ x_i &\mapsto f_i(s,t), \; i = 0, \ldots, n. \end{aligned}$$

The set of polynomials that are in the kernel of this map, that is to say the polynomials $P(x_0, \ldots, x_n)$ such that $P(f_0, \ldots, f_n) = 0$, is an ideal of $\mathbb{K}[x_0, \ldots, x_n]$ that is called the defining ideal of the curve $\mathcal{C}$, denoted $\mathfrak{I}_\mathcal{C}$. Choosing a finite set of generators of this ideal with a good shape and in small number is known to be a difficult task (see for instance [9,16,13]). In what follows, an alternative implicit representation under the form of a matrix whose entries depend on the variables $x_0, \ldots, x_n$, is presented.

A *moving hyperplane* of degree $\nu \in \mathbb{N}$ is a polynomial of the form

$$H(s,t; x_0, \ldots, x_n) = g_0(s,t)x_0 + \cdots + g_n(s,t)x_n$$

where $g_0, \ldots, g_n$ are homogeneous polynomials in $\mathbb{K}[s,t]$ of degree ν. For any point $(s_0 : t_0) \in \mathbb{P}^1$, $H(s_0, t_0; x_0 \ldots, x_n)$ is a linear form in the variables $x_0, \ldots, x_n$ that can be interpreted as the defining equation of a hyperplane in $\mathbb{P}^n$. This hyperplane moves when the point $(s_0 : t_0)$ varies in $\mathbb{P}^1$, hence its name. In addition, the moving hyperplane H is said to follow the parameterization ϕ if

$$H(s,t; f_0(s,t), \ldots, f_n(s,t)) = g_0 f_0 + \cdots + g_n f_n = 0.$$

Geometrically, this implies that the hyperplane defined by the equation $H = 0$ goes through the point $\phi(s:t) \in \mathcal{C}$.

For any integer $\nu \geq 0$, it is straightforward to compute a basis $H_1, \ldots, H_{r_\nu}$ of the vector space of moving hyperplanes of degree ν following ϕ by solving a simple linear system. We define the matrix $\mathbb{M}_\nu(\phi)$, or simply $\mathbb{M}_\nu$, as the matrix whose columns are filled with the coefficients of the moving hyperplanes H_j with respect to the variables s, t. More precisely, $\mathbb{M}_\nu$ is defined by the matrix equality

$$(H_1 \ H_2 \ \cdots \ H_{r_\nu}) = (s^\nu \ s^{\nu-1}t \ \cdots \ t^\nu) \cdot \mathbb{M}_\nu. \tag{5}$$

It is of size $(\nu+1) \times r_\nu$ and its entries are linear forms in $\mathbb{K}[x_0, \ldots, x_n]$. Therefore, it has sense to evaluate the matrix $\mathbb{M}_\nu$ at a point $p \in \mathbb{P}^n$, which we denote by $\mathbb{M}_\nu(p)$.

Proposition 2. [15,1] *For all integer $\nu \geq \delta - 1$ we have $r_\nu \geq \nu + 1$ and*

$$\text{rank } \mathbb{M}_\nu(p) < \nu + 1 \iff p \in \mathcal{C}.$$

Thus, Proposition 2 shows that the matrices $\mathbb{M}_\nu$ are implicit representations of the curve $\mathcal{C}$ for all $\nu \geq \delta - 1$, in the sense that they allow to discriminate the points $p \in \mathbb{P}^n$ that belong to the curve $\mathcal{C}$.

As we call a moving hyperplane an equation of a hyperplane that moves as the parameter $(s:t) \in \mathbb{P}^1$ varies, we call a *moving quadric* an equation of a quadric hypersurface whose coefficients depend on the parameter $(s:t) \in \mathbb{P}^1$. More concretely, a *moving quadric* of degree $\nu \in \mathbb{N}$ is a polynomial of the form

$$Q(s,t;x_0,\ldots,x_n) \quad = \quad g_{0,0}(s,t)x_0^2 \ + \ g_{0,1}(s,t)x_0x_1 \ + \ \cdots \ + \ g_{n,n}(s,t)x_n^2$$

where the polynomials $g_{i,j}(s,t)$ are homogeneous polynomials of degree ν in $\mathbb{K}[s,t]$. In addition, this moving quadric is said to follow the parameterization ϕ if

$$Q(s,t;f_0,\ldots,f_n) = \sum_{0 \leq i \leq j \leq n} g_{i,j}(s,t) f_i(s,t) f_j(s,t) = 0.$$

Similarly to moving hyperplanes, this latter condition means geometrically that the quadric defined by the polynomial Q goes through the point $\phi(s:t) \in \mathcal{C}$.

We can consider the vector space of moving quadrics following the parameterization ϕ of degree ν and, similarly to what we did with moving hyperplanes, build a coefficient matrix from them. However, such a matrix is useless in general because its entries are exclusively quadratic forms in $\mathbb{K}[x_0, \ldots, x_n]$ and hence the determinants of its minors are always polynomials of even degree. A better option is to combine both moving hyperplanes and moving quadrics in a same coefficient matrix. We proceed as follows.

Choose an integer ν and let $\langle H_1, \ldots, H_{r_\nu} \rangle$ be a basis of the vector space of moving hyperplanes following ϕ. We can consider the vector space W_ν of moving quadrics following ϕ. Each moving hyperplane H_j of degree ν following

ϕ generates $n+1$ moving quadrics of the same degree ν, still following ϕ, that are given by $x_i H_j$, $0 \leq i \leq n$. Observe that geometrically, such a moving quadric consists of the union of the moving hyperplane of equation $H_j = 0$ and the static hyperplane of equation $x_i = 0$. We denote by V_ν the sub-vector space of moving quadrics generated by these moving quadrics obtained from moving hyperplanes. Now, let $\langle Q_1, \ldots, Q_{c_\nu} \rangle$ be basis of the quotient vector space W_ν / V_ν. Then, we define the matrix $\mathbb{MQ}_\nu(\phi)$ by

$$(H_1 \; H_2 \; \cdots \; H_{r_\nu} \; Q_1 \; \cdots \; Q_{c_\nu}) = (s^\nu \; s^{\nu-1}t \; \cdots \; t^\nu) \cdot \mathbb{MQ}_\nu.$$

It is a matrix of size $(\nu+1) \times (r_\nu + c_\nu)$. By definition, the first r_ν columns of $\mathbb{MQ}_\nu$ correspond to the matrix $\mathbb{M}_\nu$ previously introduced and its entries are linear forms in $\mathbb{K}[x_0, \ldots, x_n]$. On the other hand, its last c_ν columns are built from moving quadrics and hence its corresponding entries are quadratic forms in $\mathbb{K}[x_0, \ldots, x_n]$.

Let the sequence of increasing integers $\mu_1 \leq \mu_2 \leq \ldots \leq \mu_n$ denote the degrees of a μ-basis of ϕ, which is defined in the complete thesis or in [4]. The dimension r_ν of the space of moving hyperplanes following ϕ is given by $r_\nu = \sum_{i=1}^n \max(0, \nu - \mu_i + 1)$. Here is our main result.

Theorem 2. *Assume that $\nu \geq \mu_n - 1$. Then $r_\nu + c_\nu \geq \nu + 1$ and*

$$\operatorname{rank} \mathbb{MQ}_\nu(p) < \nu + 1 \iff p \in \mathcal{C}.$$

Moreover, we have that

$$c_\nu = \sum_{1 \leq i < j \leq n} \max(0, \mu_i + \mu_j - 1 - \nu).$$

In particular, if $\nu \geq \mu_n + \mu_{n-1} - 1$ then $c_\nu = 0$ and it follows that $\mathbb{MQ}_\nu = \mathbb{M}_\nu$.

We discuss the shape of this matrix for some specific values of the degrees of the μ-basis. We emphasize that unlike in the case of plane curves, the matrices $\mathbb{MQ}_\nu$ will never be square matrices for space curves because a space curve cannot be defined by a single equation over an algebraically closed field.

In the family of matrices $\mathbb{MQ}_\nu$, $\nu \geq \mu_n - 1$, the matrix $\mathbb{MQ}_{\mu_n - 1}$ is evidently the one with the smallest number of rows. Moreover, the smallest possible value for the integer μ_n is $\lceil d/n \rceil$ because of the equality $\sum_{i=1}^n \mu_i = d$. It corresponds to the situation where the μ_i's are evenly distributed. It turns out that this balanced situation is the generic one when $\mathbb{K}$ is an algebraic closed field: fixing a degree d and picking n random homogeneous polynomials in (s,t) of degree d, $f_0, \ldots, f_n$ using a dense distribution of the coefficients such as Gaussian distribution, the degrees of its μ-basis are evenly distributed with probability 1 (see [6, Theorem 1.2] for the case $n = 2$ and [4, Section 3, Theorem 1] for a proof that generalises to arbitrary dimension $n \geq 2$).

Here are some further specific settings:

- $\mu_1 = 0$: An element of degree 0 in the μ-basis corresponds to a (non-moving) hyperplane containing the curve. In this situation, we have $\mu_2 + \ldots + \mu_n = d$ and the problem is reduced to examining a curve in $\mathbb{P}^{n-1}$ a μ-basis of which is $(p_2, \ldots, p_n)$.
- $\mu_1 = \mu_2 = 1$: In this situation, the curve is contained in a (non-moving) quadric the equation of which is given by the resultant of p_1 and p_2.
- $\mu_i = d/n$ for all i: In this case, the degree d is a multiple of n and the matrix $\mathbb{MQ}_{d/n-1}$ is purely quadratic since there is no moving hyperplane of degree $d/n - 1$ following the parameterization.

3 Conclusions

We have seen that implicitization is not a trivial operation. Although most simple varieties, e.g. the hypersurfaces of degree 1 or 2, can be represented both implicitly and parametrically, and their representation can be switched effortlessly, it is not the case any more beyond that algebraic complexity threshold. For these difficult varieties, various implicitization algorithms exist without one being superior to all the others in all the situations. Simple varieties for their part, are often constructed as approximations of complicate shapes or point clouds and they also require different kinds of methods for which design purposes are at least as important as algorithmic efficiency.

In the industry, implicit representations are often implemented to be polynomials or functions; the implicit matrix representation that we developed may require updates of softwares for them to be usable, despite their advantages (in particular, the fact that they can solve the inversion problem, finding the parameters associated with a point lying on the variety).

A very trendy approach is the use of neural networks, and more precisely deep learning methods, to achieve various kinds of goal. Deep learning can indeed be trained on producing implicit representations starting from various kind of inputs. The results might be good for approximate implicitization but hardly usable for exact implicitization. Another advantage of deep learning methods is that they get extremely efficient with the class of shapes they are trained on. While algebraic methods and, to a lesser extent, usual approximation methods are able to handle correctly a wide variety of shapes, even bizarre ones, because their heuristical aspects are adjusted by the implementations (e.g. the error measurement), neural networks automatically adapt to the objects they are given and are more efficient on similar ones.

However, we think that there is and there will always be room for algebraic standpoints when it comes to representations of geometric objects. Indeed, the richness of algebraic structures offers possibilities for geometric operations that are not possible otherwise, because polynomials are well-known and easy to handle. Approximate implicitization may have little use of algebraic theory; however, when it comes to exactness and conversions without loss of precision, the protean algebra toolbox is there to avail.

References

1. L. Busé and M. Chardin. Implicitizing rational hypersurfaces using approximation complexes. *J. Symbolic Comput.*, 40(4-5):1150–1168, 2005.
2. L. Busé and T. Luu Ba. Matrix-based implicit representations of rational algebraic curves and applications. *J. CAGD*, 27(9):681–699, 2010.
3. Laurent Busé, Clément Laroche, and Fatmanur Yıldırım. Implicitizing rational curves by the method of moving quadrics. *Computer-Aided Design*, 114:101–111, sep 2019.
4. D.A. Cox, T.W. Sederberg, and F. Chen. The moving line ideal basis of planar rational curves. *J. CAGD*, 15(8):803–827, 1998.
5. J. Dalbec and B. Sturmfels. Introduction to chow forms. In Neil L. White, editor, *Invariant Methods in Discrete and Computational Geometry: Proc. Curaçao Conference, 13–17 June, 1994*, pages 37–58. Springer, 1995.
6. Carlos D'Andrea. On the structure of μ-classes. *Communications in Algebra*, 32:159–165, 03 2004.
7. Ioannis Z. Emiris and John F. Canny. Efficient incremental algorithms for the sparse resultant and the mixed volume. *Journal of Symbolic Computation*, 20(2):117–149, aug 1995.
8. Ioannis Z. Emiris, Christos Konaxis, and Clément Laroche. Implicit representations of high-codimension varieties. *Computer Aided Geometric Design*, 74:101764, oct 2019.
9. E. Fortuna, P. Gianni, and B. Trager. Generators of the ideal of an algebraic space curve. *Journal of Symbolic Computation*, 44(9):1234–1254, sep 2009.
10. I. M. Gelfand, M. M. Kapranov, and A. V. Zelevinsky. *Discriminants, Resultants and Multidimensional Determinants*. Birkhäuser, Boston, 1994.
11. Heisuke Hironaka. Resolution of singularities of an algebraic variety over a field of characteristic zero. *The Annals of Mathematics*, 79(1):109, jan 1964.
12. Gabriela Jeronimo, Teresa Krick, Juan Sabia, and Martín Sombra. The computational complexity of the chow form. *Foundations of Computational Mathematics*, 4(1):41–117, Feb 2004.
13. X. Jia, H. Wang, and R. Goldman. Set-theoretic generators of rational space curves. *J. Symbolic Comput.*, 45(4):414–433, 2010.
14. William E. Lorensen and Harvey E. Cline. Marching cubes: a high resolution 3D surface construction algorithm. *ACM SIGGRAPH Computer Graphics*, 21(4):163–169, aug 1987.
15. T.W. Sederberg and F. Chen. Implicitization using moving curves and surfaces. In R. Cook, editor, *Proc. SIGGRAPH*, volume 29, pages 301–308. Addison Wesley, 1995.
16. T.W. Sederberg, R. Goldman, and H. Du. Implicitizing rational curves by the method of moving algebraic curves. *J. Symbolic Comput.*, 23(2-3):153–175, 1997.
17. Giovanni Staglianò. Macaulay2 package "Resultants", May 2018. Available at `http://www2.macaulay2.com/Macaulay2/doc/Macaulay2-1.14/share/doc/Macaulay2/Resultants/html/`.
18. J.J. Sylvester. A method of determining by mere inspection the derivatives from two equations of any degree. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 16(101):132–135, feb 1840.
19. Xinyu Zhang, Young Kim, and Dinesh Manocha. Reliable sweeps. In *2009 SIAM/ACM Joint Conference on Geometric and Physical Modeling on - SPM '09*, pages 373–378. ACM Press, 01 2009.