Applications of the Classical Theory of Invariants in Computer Vision[1]

Hakan Çivi and Aytül Erçil
Boğaziçi University
Department of Industrial Engineering,Bebek, Istanbul
Colin Christopher
Boğaziçi University
Department of Mathematics, Bebek, Istanbul

Abstract

Combining implicit polynomials and algebraic invariants for representing and recognizing complicated objects proves to be a powerful technique. In this paper, we explore the findings of the classical theory of invariants for the calculation of algebraic invariants of implicit curves and surfaces, a theory largely disregarded in the vision community by a shadow of skepticism. Here, the symbolic method of the classical theory is described, implemented and the results are discussed in their relations with computer vision.

1 Introduction

The idea of invariance is an inherent element in almost all object recognition works. One can trace at least two different usages of the concept: *photometric invariance* and *geometric invariance*. Of photometric properties whose invariance is sought are certain intensities such as shading, reflectance, color, etc. These properties arise from a complex interaction of surface characteristics, illumination conditions and view changes which is difficult to control. The paradigm associated with the use of geometric invariance views geometry as the basis for computer vision. With this approach the recognition problem is formulated as extracting ultimate invariant descriptors, stemming from the geometry of the object e.g., edges, which are independent of certain kind of viewpoint changes but not others. The viewpoint changes are expressible in terms of the action of a transformation group such as Euclidean, affine or projective motions.

Geometric invariance has found a wide application in several works on object recognition and pose estimation [12, 7]. Among approaches having this flavor are those exploiting local differential features -like curvature- and those combining local and global information through semi-differential invariants. In model-based vision using implicit polynomials, the focus has been mostly on algebraic invariants of polynomial curves or surfaces under various groups of coordinate changes [11, 5]. An algebraic invariant of a polynomial curve or surface in this case is just a function of polynomial coefficients. It is the generation of these functions which concerns us in this paper.

In mathematics, the search for *invariants*- that is quantities which remain the same under a given group of mappings- is fundamental, and its roots go far back into its history. Perhaps the most far reaching of these was the Erlangen Program of F.Klein in the 19th Century, where different geometries are classified by their invariants. According to this view, Euclidean geometry, for example, can be considered as the study of motions of the plane for which the distance between any two points is preserved. While in computer vision, the main question is to choose the right class of invariants for indexing shapes to be recognized. This choice tightly depends on the camera model and calibration assumptions, as well as the types of the shapes of interest [9].

Another product of the 19th century search for invariants was the classical theory of invariants due to Cayley, Sylvester, Young and many other leading mathematicians of the day including, of course, Hilbert. As aluded to above, the idea was to seek polynomial functions of

[1]Supported by TUBITAK NATO-A2, TUBITAK Misag-66, and DPT95K120320 Grants

the coefficients of a curve or surface which are invariant under various transformations. The two main problems that confronted them were how to calculate such invariant polynomials and how to know when you have found them all. A beatiful and complete answer to the first question was found by means of the symbolic method- of which more later. The second question was resolved by Hilbert in two outstanding papers in 1890 and 1893 [4]. The first of these shows that there is a finite set of invariants which generate all the others. The second demonstrates how one can constructively determine when this set has been obtained. Ironically, these papers which heralded a new era for mathematics contributed to the downfall of invariant theory. Having solved the major questions, only tedious and highly complex task of classifying the invariants remained. However, the mood of the times changed and constructive mathematics received very little attention. To this day very few real systems of equations have been exhaustively studied.

The subsequent history of invariant theory is quite chequered. Already in 1971 Dieudonné could write: "Invariant theory has already been pronounced dead several times, and like the phoenix it has been again and again rising from its ashes". In the past decade or so, a new interest has again been taken in the classical theory of invariants by combinatorics. In particular, the work of Kung and Rota sought to put the classical symbolic method onto more acceptable mathematical foundations. We shall describe this method in the next section.

Until now, the method of computing algebraic invariants for computer vision has been ad hoc. Attention has been restricted to arbitrary classes of invariants, or they have been computed by exhaustive brute force searches. In particular, the findings of the classical theory, especially the symbolic method, have often been ignored. The reason behind this has been a partially justifiable scepticism about the method's applicability [12, 5].

The aim of this paper is to demonstrate a more direct approach to the calculation of algebraic invariants via the symbolic method. Although there are several drawbacks to this method at the moment, we claim that we have a direct way of computing complete system of invariants up to a given degree for projective, affine and Euclidean groups of motion. Furthermore the algorithm is such that only to be coded once and can thereafter be applied to curves or surfaces of any degree. If required the method can be readily combined with a standard Grobner Basis package to remove unnecessary invariants.

The paper is organized as follows: In section 2 we provide more details of the symbolic method. In section 3 the implementation of the algorithm is described. The final section compares our results with alternative methods and draws some conclusions.

2 The Symbolic Method

2.1 Interpretations of the Symbolic Method

The symbolic method provides an elegant procedure for expressing the invariants of a given homogeneous polynomial, or form. There is no restriction in considering only homogeneous polynomials as there is a 1-1 correspondence between homogeneous polynomials in m variables and non-homogeneous polynomials in m–1 variables, via the transformation $F(x_1, x_2, \ldots, x_{m-1}) = x_m^n \phi(x_1, x_2, \ldots, x_m)$. Furthermore, we shall see later that invariants of F can be obtained from joint invariants of ϕ and x_m.

The main idea behind the symbolic method can be summarized as follows: given a binary form

$$f(x, y) = \sum_{k=0}^{n} \binom{n}{k} a_k \, x^k \, y^{n-k} \tag{1}$$

we rewrite it in the form $f(x, y) = (\alpha_1 x + \alpha_2 y)^n$ where α_1 and α_2 are symbols which we agree to evaluate as $A_0 = \alpha_1^n, A_1 = \alpha_1^{n-1}\alpha_2^1, \ldots, A_n = \alpha_2^n$. Although this would seem to lead to great confusion, a moment of reflection will convince the reader that as long as the coefficients A_i appear linearly in a function no ambiguity can occur. However, if we consider quadratic terms in the A_i then one can show $A_0 A_2 = A_1^2$! To overcome this difficulty, a second set of symbols β_1

and β_2 are introduced. Hence in the example above $A_0A_2 = \alpha_1^n\beta_1^{n-2}\beta_2^2$, and $A_1^2 = \alpha_1^{n-1}\alpha_2\beta_1^{n-1}\beta_2$. A similar process is observed for higher degree terms in the A_i.

For any pair of symbols α, β a bracket is defined as $[\alpha\beta] = \alpha_1\beta_2 - \alpha_2\beta_1$. The fundamental theorem of invariant theory is that every invariant can be written as a product of brackets, and every product of brackets with correct number of symbols is an invariant.

There are at least three different interpretations or formal justifications of the symbolic method. the first approach, used by Grace and Young [2], suggest treating each of the symbols α_1, α_2 as operators $\partial/\partial x_\alpha$ and $\partial/\partial y_\alpha$ acting on their own copy of the polynomial $f(x_\alpha, y_\alpha)$. The second approach, given by Gurevich [3], relates the polynomial f to a symmetric tensor $c_{ij\ldots k}x^ix^j\ldots x^k$. The symbols are then considered as place markers for various operations on the indices of c. This approach can also be seen, in a more abstract guise, in Dieudonné's book [1]. The third approach is that of Kung and Rota [6] who seek to justify the symbolic method directly via *umbral calculus*.

2.2 Umbral Notation

We now give a brief account of the work of Kung and Rota as it pertains to our own work. For simplicity of representation, we explain the notation for binary forms (that is, homogeneous polynomials in two variables), the case for more than two variables follows through with trivial modifications. We refer the reader to the above article and the Sturmfels's book [10] for complete proofs of the theorems stated below.

2.2.1 Binary Forms and Invariants

Consider the binary form (1). After a linear transformation

$$\begin{aligned} x &= \xi_1x' + \delta_1y' \\ y &= \xi_2x' + \delta_2y' \end{aligned} \tag{2}$$

we obtain the expression

$$f'(x', y') = \sum_{k=0}^{n}\binom{n}{k} a_k(\xi_1x' + \delta_1y')^k(\xi_2x' + \delta_2y')^{n-k} = \sum_{k=0}^{n}\binom{n}{k} a_kx'_ky'_{n-k} \tag{3}$$

An invariant of weight w of a binary form of degree n is defined as a polynomial $I(x_0, \ldots, x_n)$ for which following identity holds:

$$I(a'_0, \ldots, a'_n) = (\xi_1\delta_2 - \xi_2\delta_1)^wI(a_0, \ldots, a_n) \tag{4}$$

An *absolute invariant* is an invariant of weight zero. In the same way, given a set of r binary forms $f_i(x, y)$ of degree $r(i)$

$$f_i(x, y) = \sum_{k=0}^{r(i)}\binom{n(i)}{k}\ a_{ik}\,x^k\,y^{n(i)-k}\quad i = 1, 2, \ldots, r \tag{5}$$

we define a *joint invariant of weight* w to be a polynomial $I(x_{10}, \ldots, x_{1n(1)}, \ldots, x_{rn(r)})$ for which

$$I(x'_{10}, \ldots, x'_{1n(1)}, \ldots, x'_{rn(r)}) = (\xi_1\delta_2 - \xi_2\delta_1)^wI(x_{10}, \ldots, x_{1n(1)}, \ldots, x_{rn(r)}) \tag{6}$$

where the primed variables are obtained in a manner similar to above. It can be shown that any invariant can be written as a sum of invariants homogeneous in each of the sets of coefficients. We will only consider invariants of that sort here.

2.2.2 The Umbral Calculus

Let Φ be an alphabet of an infinite number (!) of Greek letters. Each Greek letter α is associated with two variables α_1, α_2. The *umbral space* U is the set of polynomials in these letters (viewed as variables).

The umbral operator $\Psi : U \to P$ is a linear operator from umbral space U to P, the ring of polynomials in the coefficients of f; it obeys the following rules:

1. $\langle\,\Psi\,|\,\alpha_1^k\alpha_2^{n-k}\,\rangle = \alpha_k$
2. $\langle\,\Psi\,|\,\alpha_1^i\alpha_2^j\,\rangle = 0\ \ for\ \ i + j \neq 0$
3. $\langle\,\Psi\,|\,\alpha_1^i\alpha_2^j\,\beta_1^k\beta_2^l\ldots\gamma_1^m\gamma_2^n\,\rangle = \langle\,\Psi\,|\,\alpha_1^i\alpha_2^j\,\rangle\langle\,\Psi\,|\,\beta_1^k\beta_2^l\,\rangle\ldots\langle\,\Psi\,|\,\gamma_1^m\gamma_2^n\,\rangle$

Every polynomial $I(a_0, \ldots, a_n)$ has an *umbral representation* $Q \in U$ for which I is equal to $\langle \Psi \mid Q(a_1, \ldots, a_n) \rangle$, conversely I is called the *umbral evaluation* of Q.

Umbral notation for several forms is defined in a similar way. For each form $f_i(x, y)$, we take a distinct alphabet. The umbral operator is extended in the obvious fashion.

2.2.3 Bracket Polynomials and the Fundamental Theorem

Given symbols as above, we define their bracket $[\alpha\,\beta\,]$ to be $\alpha_1\beta_2 - \alpha_2\beta_1$. An element of U which can be expressed as a polynomial in these brackets will be termed a *bracket polynomial.* A *bracket monomial* M is a product of brackets $M = [\alpha\,\beta\,][\alpha\,\delta\,]\ldots[\omega\,\nu\,]$ where the total number of brackets is called the *weight* of the monomial. If a bracket polynomial consists entirely of monomials of a fixed weight w, then we say the polynomial is of weight w. Let B be the subspace of bracket polynomials in U, and B_w the space of bracket polynomials of weight w.

Theorem 1(The fundamental theorem) Every invariant of weight w can be written as the umbral evaluation $\langle \Psi \mid P \rangle$ of a bracket polynomial P in B_w, and every such evaluation is an invariant of weight w.

2.2.4 Generating Invariants through the Standard Tableau

The fundamental theorem characterizes invariants in terms of bracket poynomials. However, if the theorem is to serve as a basis for the calculation of invariants, it becomes necessary to eliminate redundant expressions as much as possible. Such redundancy comes from two sources. First, an umbral expression may have several representations as bracket polynomials. The second problem is that two umbral expression may evaluate to the same expression.

The first problem is resolved by introducing a lexicographic ordering to the letters of the alphabet(s). A bracket monomial $M = [\alpha\,\beta\,][\alpha\,\delta\,]\ldots[\omega\,\nu\,]$ can be viewed as a tableau

$$\begin{pmatrix} \alpha & \beta \\ \alpha & \delta \\ & \vdots & \\ \delta & \gamma \end{pmatrix} = [\,\alpha\,\beta\,][\,\alpha\,\delta\,]\ldots[\,\delta\,\gamma\,]$$

Such a tableau is called *standard* if letters are ordered increasingly from left to right, and nondecreasingly from top to bottom.

Theorem 2 The standard tableaux of weight w form a basis of B_w.

Thus, when generating invariants from tableaux, it is only necessary to consider those tableaux which are standard.

It is worth mentioning at this point that given an element in B_w, it is possible to rewrite the brackets in an algorithmic way into a linear combination of standard tableaux- the so-called *straightening algorithm* [10]. However, we will not need this what follows.

As for the second problem, it can be shown [6] that an umbral expression evaluates to zero if and only if its symmetrisation with respect to equivalent letters is zero. Unfortunately, the process of symmetrisation and that of monomial ordering are mutually exclusive, thus we are led to seek different ways to ensure that duplicate invariants are discarded.

2.3 Joint, Affine and Euclidean Invariants

Joint invariants (also called mutual invariants) of several forms can be computed using the standard tableau as described in the previous subsection. Here, lexicographic ordering can be arbitrarily extended between different alphabets. Although there may be many latent applications of joint invariants in computer vision such as associated objects or objects represented through more than a single polynomial, our interest lies in how joint invariants allow one to obtain invariants for different groups of motion.

It can be shown (see Gurevich [3] for an excellent introduction) that the affine group of motions are exactly those which leave the line at infinity invariant (fixed). It follows that every affine invariant of a form can be written as a joint invariant of the form together with the line at infinity. In two dimensions, this amounts to finding the joint invariants of a ternary form with the linear form $f(x, y, z) = (\alpha_1 x + \alpha_2 y + \alpha_3 z)$ (in symbolic notation) where α_1 and α_2 evaluates to zero and α_3 evaluates to 1.

It is also possible to treat Euclidean invariants by the symbolic method. Here, we calculate joint invariants of the form, the line at infinity and the second order tensor

$$\eta^{ij} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

In order to do this we need a slight generalization of the symbolic method described so far. The details of this can be worked out directly from the framework given by Gurevich [3], and will be presented elsewhere. It suffices for our purposes here to say that we need only to add a new type of bracket symbol $(\alpha\,\beta)$ which represents the product $\alpha_1\beta_1 + \alpha_2\beta_2$ All invariants can be expressed as sums of products of the two types of brackets [] and () and vice versa.

It is possible to extend the tableaux mentioned above to contain these two types of brackets. Furthermore, if ζ, ξ are symbols representing the line at infinity, then it is easy to show that $[\alpha\,\beta\,\zeta][\gamma\,\delta\,\xi] = (\alpha\,\gamma)(\beta\,\delta) - (\alpha\,\delta)(\beta\,\gamma)$. Thus we need to consider at most one bracket of the form $[\alpha\,\beta\,\zeta]$.

2.4 Gröbner Bases and Hilbert's Basis Theorem

The main drawback to the symbolic method is that the same invariant may be generated many times. However, even if we exclude multiple copies of a given invariant we are still left with the problem that new invariants are just polynomials in the old invariants. But this problem is inherent in all other approaches used so far, as there is no analysis to remove such dependencies. Fortunately, theoretically at least, there is a method to deal with this problem which takes us back to the original solution Hilbert gave to the existence of a finite number of invariants.

Recall that Hilbert's basis theorem (developed exactly for this purpose) says that given an infinite number of polynomials $f_1, f_2, \ldots$ in a fixed number of variables, there exists a number N such that for any $i > N$, f_i can be written in the form

$$f_i = \sum_{k=1}^{N} g_k f_k \tag{7}$$

with the g_i polynomials in the same variables. Of course, this is not the same as saying that f_i is a function of $f_1, \ldots, f_N$. However in the case where the f_i are all invariants, it is possible to 'average' the coefficients g_i in such a way that they are also invariants.

Suppose that we have calculated all homogeneous invariants order less than m say (as the process which we outlined does for example). If a new invariant lies in the ideal generated by the ones already calculated, then we have a relation of the form (7) with the g_i invariants. It is easy to see that we can also choose that the degrees of the g_i are less than m and hence (7) expresses the fact that our new invariant is merely a polynomial in the invariants calculated so far.

To use this fact, we need to have some information about the ideal generated by our invariants. Fortunately, this is provided by the calculation of a Gröbner basis of the invariants. If a new invariant lies in the ideal of these invariants then its reduction modula this basis will be zero. If not, then the invariant provides new information and should be added to the list of invariants. A new Gröbner basis should then be calculated. Gröbner basis calculations and modulo a Gröbner basis are readily available for most computer algebra packages.

3 Implementation

The implementation was done by the REDUCE symbolic computation software [8] running on a SUN Sparc2. The two inputs to the program are the degree of the form (or degrees of the forms in the joint invariant case). The program performs three tasks: the generation of standard tableaux, the conversion of these into polynomials in the coefficients of the form, and the prunning of redundancy and dependencies.

The tableau generation phase exhaustively produces all possible tableaux up to a given height w iterating through the number of symbols. Since each bracket consists of k symbols for k-variable forms (2 for binary, 3 for ternary), the total degree of symbols in a tableau of height w is kw. If i is the degree of the invariant (number of symbols) then $w = n.i/k$. Similarly for the joint invariant of r different forms $w = 1/k(n_1 i_1 + \ldots + n_r i_r)$ where $n_1, \ldots, n_r$ are degrees of forms and $i_1 \ldots, i_r$ are the degrees of the invariant in the coefficients of each form. In the second phase, the standard bracket monomials are multiplied out to obtain polynomials in the umbral symbols. These in turn are converted to polynomials in the coefficients of the form, using the umbral evaluation rules implemented through the substitution/pattern matching mechanism of the REDUCE. The final step takes each new invariant and checks whether it can be expressed as a polynomial of previous invariants. This, as explained above is done by reducing the invariant modulo the current Gröbner basis. If the reduction is non-zero then the invariant is added to the list, and a new Gröbner basis is calculated.

4 Conclusion

In this work, we discussed the symbolic method of the classical theory of invariants to calculate projective, affine, Euclidean and joint invariants for polynomial curves and surfaces. The main advantages of the method are its generality and completeness, i.e., given a polynomial (2-D or 3-D) of any degree or pairs of such the method can compute the full system of invariants up to a required degree.

Our experiments showed that for low degree polynomials (2 to 4) the algorithm is very efficient. While for higher degrees the performance dramatically decreases due to the height of the standard tableau and its conversion into the corresponding algebraic invariant. Sometimes the expression length may exceed the limits handlable by the software bringing about further complications. This however is no surprise! Algebraic invariants of polynomials are normally pages long.

From the implementation point of view, the two research problems to be further addressed are the optimization of the tableaux generation -which is of combinatorial nature- through shortcuts to eliminate the repetitions at the generation phase and a closer investigation of the conversion into the algebraic invariants. From computer vision standpoint, in compliance with the spirit of the original research problem, the experiments with the computed invariants needs to be carried out to verify their practical usefulness.

References

[1] J. A. Dieudonné and J. B. Carrell, *Invariant theory, old and new*, Academic Press, 1971.

[2] J. H. Grace and A. Young, *The Algebra of Invariants*, Cambridge Univ. Press, 1903.

[3] G. B. Gurevich, *Foundations of the theory of algebraic invariants*, P. Noordhoff, 1964.

[4] D. Hilbert, *Theory of algebraic invariants*, Cambridge University Press, 1993.

[5] D. Keren, "Using Symbolic Computation to Find Algebraic Invariants," *IEEE PAMI.*, vol 16, no.11, pp. 1143-1149, 1994.

[6] J. P. S. Kung, G-C. Rota, "Invariant Theory of Binary Forms," *Bull. of the Amer. Math. Soc.*, Vol. 10, no. 1, pp. 26-85, 1984.

[7] J. L. Mundy, A. Zisserman, *Geometric invariance in machine vision*, MIT Press, 1992.

[8] G. Rayna, *REDUCE:software for algebraic computation*, Springer-Verlag, 1987.

[9] L.S. Shapiro, A. Zisserman, M. Brady, "3D Motion Recovery via Affine Epipolar Geometry," *IJCV*, 16, 147-182, 1995.

[10] B. Sturmfels, *Algorithms in invariant theory*, Springer-Verlag, 1993.

[11] G.Taubin, D.B.Cooper, "2D and 3D Object Recognition and Positioning with Algebraic Invariants and Covariants," in *Symbolic and Numerical Comput. for Artif. Int*, (ed.)B.R.Donald.

[12] I. Weiss, "Geometric Invariants and Object Recognition," *IJCV*, 10:3, pp. 207-231, 1993.