Available online at www.sciencedirect.com

SciVerse ScienceDirect

Mathematics and Computers in Simulation 104 (2014) 3–20

MATHEMATICS AND COMPUTERS IN SIMULATION

www.elsevier.com/locate/matcom

A parametric approach to 3D dynamic geometry

Francisco Botana *

Departamento de Matemática Aplicada I, Universidad de Vigo, Campus A Xunqueira, 36005 Pontevedra, Spain

Received 31 August 2011; received in revised form 26 September 2012; accepted 14 December 2012
Available online 30 May 2013

Abstract

Dynamic geometry systems are computer applications allowing the exact on-screen drawing of geometric diagrams and their interactive manipulation by mouse dragging. Whereas there exists an extensive list of 2D dynamic geometry environments, the number of 3D systems is reduced. Most of them, both in 2D and 3D, share a common approach, using numerical data to manage geometric knowledge and elementary methods to compute derived objects.

This paper deals with a parametric approach for automatic management of 3D Euclidean constructions. An open source library, implementing the core functions in a 3D dynamic geometry system, is described here. The library deals with constructions by using symbolic parameters, thus enabling a full algebraic knowledge about objects such as loci and envelopes. This parametric approach is also a prerequisite for performing automatic proof. Basic functions are defined for symbolically checking the truth of statements. Using recent results from the theory of parametric polynomial systems solving, the bottleneck in the automatic determination of geometric loci and envelopes is solved. As far as we know, there is no comparable library in the 3D case, except the paramGeo3D library (designed for computing equations of simple 3D geometric objects, which, however, lacks specific functions for finding loci and envelopes).

Keywords: 3D dynamic geometry; Automated deduction; Groebner bases; Parametric polynomial systems; Degenerated conditions

1. Introduction

A method for mechanical theorem proving in geometries was proposed by Wu [39,40] during the 1980s, and, almost simultaneously, a novel application for the algorithm of Buchberger [9], also related with automated deduction, was studied (see, for instance [20]). Towards the end of that decade, two learning environments for geometry appeared in the educational software market, The Geometer's Sketchpad, GSP [34] and Cabri [11]. These programs marked the birth of the dynamic geometry (DG) paradigm, and formed the core of mathematical software used in schools.

A first implementation of the algorithm of Wu was carried out by Chou [13], followed by the work of Wang [36], and a DG environment using Wu's method, Geometry Expert, GEX, was developed [16]. Regarding the DG paradigm, the novel approach in GEX relied on using formal proving methods rather than the visual convincement offered by Cabri or GSP. Some authors tried to remedy this shortcoming of popular DG software through intercommunication with Computer Algebra Systems (CAS). For example, Roanes-Lozano et al. [27] use GSP constructions to perform proving tasks by means of a Maple library with a simple implementation of Wu's method. Botana and Valcarce [6,7]

* Tel.: +34 986801931; fax: +34 986801907.
E-mail address: fbotana@uvigo.es

0378-4754/$36.00 © 2013 IMACS. Published by Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.matcom.2012.12.004

develop a DG environment that communicates with CoCoA [14] and Mathematica, and where Groebner bases are used for loci finding and automated proof and discovery. Botana has also implemented a Mathematica based web application [3] for remotely dealing with 3D geometric constructions, and Roanes-Lozano et al. [26] have replied this approach using Maple with local access only.

In this paper we deal with a parametric description of 3D Euclidean constructions and the involved algorithms to perform automatic discovery processes in 3D DG systems. In Section 2 we state what a parametrical description of a geometric construction is and we review the state of the art concerning these approaches in 3D. Section 3 describes an open source library implementing our parametric setting, extends an elimination based approach for the automatic finding of new objects (loci and envelopes) in geometric constructions, and illustrates the proposal through selected examples. Section 4 discusses some drawbacks of the elimination techniques and introduces a new application of the theory of parametric systems solving to overcome these limitations.

2. Parametric description of dynamic geometry constructions

2.1. Issues in 2D

Most DG systems incorporate an option that allows users to see the constructive steps of constructions. Although there is no common terminology for it, standard systems such as the above cited Cabri and GSP, or the emerging educational standard GeoGebra [17], exhibit an almost natural language description of the steps in a construction. For instance, consider a circle in the plane defined by its center $O(0, 0)$ and passing through $A(0, 2)$. Define a point P lying on the circle and a new circle centered at P with radius 1. Drawing a line through A and P, define a common point X on the line and the last circle. Finally, ask for the locus of X when P moves along the basic circle. Fig. 1 shows this construction and its step-by-step description in GeoGebra.

Although primarily introduced for didactic purposes, this kind of descriptions has been used to capture the ordered constructive steps for other goals. Since the first standard DG systems, GSP and Cabri, are proprietary software, their code was not released and their constructions were encrypted or, at least, specified through a not easily accessible grammar. These were the main reasons why this option of exporting the construction protocol was used for further processing. The appearance of the GeoGebra open source system has made unnecessary the use of the protocol of constructions. GeoGebra specifies its constructions with a public grammar and by using XML. In any case, we assume here that the constructive part of a DG diagram can be easily read. We do not give here the details of such an approach, referring to [4], where a prototype reading GeoGebra constructions is described. Indeed, the European project Intergeo (see i2geo.net) had among its goals the specification of a common format for file exchange between DG systems. Although this format is currently unfinished, some experiments about embedding it in an automated deduction environment are reported in [5].

Going back to the GeoGebra construction, note that the first three steps consist of defining points O and A and the circle c. Whereas the points do not have any text on the Definition field of the Construction protocol, the circle is recorded via its constructive statement. At the same time, the Algebra field shows the coordinates of the points and the equation of the circle. As is well-known, if the user drags, let us say, A, the present position of the point changes and so do the trace and equation of the circle accordingly. We can interpret this behavior as a parametric one: the value of the variable (the circle c) is refreshed whenever its parameters (the points O, A) are updated. Nevertheless, DG developers usually rely on numeric assignments, that is, once A is moved, the system recomputes the equation of c and obtains this numeric expression as the whole algebraic knowledge about the circle. Now, placing a point P on c is again a parametric issue: given the position of the pointer on the screen and the equation of c, compute P coordinates satisfying both requirements and with a predetermined numeric precision. Thus, refreshing the drawing when some element is dragged is just a problem of moving through the data structure that encompasses the construction. So, any question involving a generic point P placed on c cannot be generally answered. This numeric approach faces its limitations when computing the last object in the considered example. In order to compute the locus of X when P moves along circle c, the standard strategy in DG consists of sampling the path of P and obtaining for each sample element a position for X. Joining the X positions, a line on the screen is returned as the sought locus. This method has two inherent limitations: no algebraic knowledge about the locus is obtained, and, since the sampling strategy must be a general one, returned loci sometimes exhibit anomalous behavior (see [2] for an in-depth study of this issue).