WinGCLC — a Workbench for Formally Describing Figures

Predrag Janičić
Faculty of Mathematics, University of Belgrade,
Studentski trg 16, 11000 Belgrade, Yugoslavia
url: www.matf.bg.ac.yu/~janicic
email: janicic@matf.bg.ac.yu
Ivan Trajković
Faculty of Mathematics, University of Belgrade,
Studentski trg 16, 11000 Belgrade, Yugoslavia
email: ivan_t@verat.net

Abstract

In this paper we discuss the problem of describing geometrical/mathematical figures — the problem important for virtually all mathematicians. We also present our system WinGCLC, based on the idea of describing, rather than drawing figures. The system is suitable for use in teaching mathematics (and, especially, teaching geometry), for producing digital mathematical illustrations in different formats (including LaTeX format and bitmap format), and sometimes even as a guidance in problem solving or in some pieces of research. The system is very handy and gives an easy-to-use support for a number of geometrical (and not only geometrical) devices, easy manipulation with objects, animation etc.

1 Introduction

In teaching mathematics, and especially in teaching geometry, one is in a need for demonstrating and illustrating concepts and properties of different objects. In this, it cannot be overemphasized, there is a need of distinguishing abstract (i.e., formal, axiomatic) nature of geometrical objects and their usual models (e.g., Cartesian models). For example, a geometrical construction is a mere procedure of abstract steps and not a picture. However, for each construction, there is its counterpart in the standard Cartesian model.

When a mathematician produces a picture, he/she usually wants the picture to represent some concepts and/or some mathematical objects. Very often it is much more plausible to have a picture description rather than the picture itself. A picture description should be abstract, precise description of a mathematical meaning of the picture and from it one can generate the picture itself (while, of course, the other way round it is not possible). Clearly, mathematicians more often want to deal with formal descriptions of their pictures than with pictures themselves. It is also highly plausible to have a support for easy producing figures usable by systems such as LaTeX.

WinGCLC package (and its command-line version GCLC) is a tool for easy making of geometrical (but not only geometrical) figures.[1] The package is small and very easy to use. It enables producing figures in formats such as LaTeX format or the bitmap format. Figures in LaTeX format are natively supported by LaTeX and, hence, use LaTeX fonts. Making figures in WinGCLC is based on the idea of "describing figures" rather than of "drawing figures" (which we have discussed in two previous paragraphs). So, in a sense, this system is in spirit close to the TeX/LaTeX system [6, 7], or it is parallel to it. This approach for describing figures stresses, for instance, the fact that geometrical constructions are abstract, formal procedures and not figures. A figure can be generated on the basis of abstract description, in Cartesian model of Euclidean plane or Euclidean space. In the rest of this paper we will discuss only plane geometry and plane constructions.

[1] WinGCLC is freely available on-line from www.matf.bg.ac.yu/~janicic/gclc/. The first version of GCLC was made by the first author in 1996. and has been publicly available since then. GCLC comes from *Geometry Constructions ⇒ LaTeX Converter*. So far, the figures for a number of books and journal issues have been prepared by the GCLC package. This system has been guided by eight years of teaching geometry (of the first author) at the university level and hundreds of made figures and digital illustrations.

Overview of the paper: The structure of the rest of the paper is as follows: in Sect. 2 we discuss formal constructions and basic ideas of WinGCLC; in Sect. 3 we briefly present GCLC language and in Sect. 4 we give several examples; in Sect. 5 we discuss advanced object manipulations provided by WinGCLC; in Sect. 6 we present some of GCLC additional modules and in Sect. 7 we discuss export formats; in Sect. 8 we give a brief overview of some of related work; in Sect. 9 we present some ideas for future work and further improvements of WinGCLC and in Sect. 10 we draw final conclusions.

2 Formal Constructions and WinGCLC — Basic Ideas

As said, WinGCLC package is a tool which enables producing mathematical/geometrical figures (i.e., digital illustrations) on the basis of their formal descriptions. This approach is guided by the idea of formal geometrical constructions. The concept of formal geometrical constructions has been studied literary thousands of years, and since the ancient Greeks (and especially since Euclid's *Elements* [2]), it became a standard part of virtually any sort of education. One of the reasons for this was a general opinion that the rigor of geometrical constructions substantially helps in developing logical thinking. The rigor in geometrical proofs was one of the motivating reasons for the modern reform of mathematics, with Hilbert's *Grundlagen der Geometrie* as one of the milestones [3]. Modern approach to classical, synthetical geometry is still very much based on Hilbert's visions. During all this time, geometrical constructions remained one of the most rigor and yet most attractive parts of geometry.

A geometrical construction is a sequence of specific, primitive construction steps. These primitive construction steps we also call *elementary constructions* and they are:

- construction (by *ruler*) of a line such that two given points belong to it;
- construction of a point which is an intersection of two lines (if such a point exists);
- construction (by *compass*) of a circle such that its center is one given point and such that the second given point belongs to it;
- construction of a segment connecting two points;
- construction of intersections between a given line and a given circle.

The above primitive constructions use *abstract* instruments *ruler* and *compass*. For Euclidean geometry, usual real-world instruments ruler and compass, can help in making approximative representation of formal constructions in Cartesian plane. However, one should not mix-up these instruments with the abstract instruments which instances in different geometries can have different properties (e.g., abstract ruler will differ in Euclidean and hyperbolical geometries).

By using the set of primitive constructions, one can define more involve constructions (e.g., construction of the right angle, construction of the midpoint of a line segment, construction of the bisector of an angle etc.). In describing geometrical constructions, it is usual to use higher level constructions as well as the primitive ones.

WinGCLC package follows the idea of formal constructions. It provides an easy-to-use support for all primitive constructions, but also for a range of higher-level constructions. In addition, WinGCLC provides support for isometric transformations, general conics, etc. Although motivated by the formal geometrical constructions, WinGCLC provides a support for some non-constructible objects too (for instance, in WinGCLC it is possible to determine/use a point obtained by rotation for 1°, although it is not possible to construct that point by ruler and compass).

WinGCLC uses a specific language for describing figures. These descriptions are compiled by the processor and can be exported to different output formats. There is an interface which enables simple and interactive use of a range of functionalities, including making animations.

While a construction is an abstract procedure, in order to make its representation in Cartesian plane, we still have to make a some link between these two. For instance, given three vertices of a triangle we can construct a center of its inscribed circle (by using primitive constructions), but in order to represent this construction in Cartesian plane, we have to take three particular Cartesian points as vertices of the triangle. Thus, figure descriptions in WinGCLC are usually made by a list of definitions of several (usually very few) fixed points (defined in terms of Cartesian plane, e.g., by pairs of coordinates) and a list of construction steps based on that points.

3 GCLC Language

GCLC language is not a script language. It is rather a higher-level language (with a support for a number of advanced geometrical devices) designed for mathematicians. On one hand, it is very simple (it does not require programming skills) and, on the other hand, it enables describing very complex figures in only very few lines. There are no sorts, and figure descriptions can be written in a very flexible way.

GCLC language consists of the following groups of commands:

basic definitions: these commands include commands for defining fixed points, but also commands for defining a line on the basis of two selected point, defining a circle, a numeric constant etc.

basic constructions: these constructions include constructions of intersection points for two lines, or for a line and a circle, construction of the midpoint of a given segment, the bisector of an angle, perpendicular lines, parallel lines, etc.

transformations: these commands include commands for translation, rotation, line-symmetry, half-turn, but also some non-isometric transformations like scaling, circle inversion etc.

drawing commands: there are commands for drawing lines, segments, circles, arcs, and ellipses in several modes.

marking and printing commands: points can be marked in a number of ways. In addition, a text can be attached to a particular point (or a position).

low level commands: there is a support for changing line thickness, color, clipping area, figure dimensions etc.

Cartesian commands: this group of commands provides support for direct access to a defined Cartesian system. A user can define a system, its unit, and, within it, can define points, lines, conics, tangents etc.

commands for describing animations: this group of commands provides support for making animations in WinGCLC. Some points can be defined to move from one position to another, while points can also be *traced*.

4 Examples

In this section we present two examples illustrating some GCLC commands. Example given in Fig. 1 illustrates some of geometrical constructions. Groups of commands are explained in comments within the description itself. The output (in LaTeX format) is presented in Fig. 2. In this example we construct three bisectors of the angles of a triangle ABC. It is very well known that these three lines intersect in one point (in the center of the inscribed circle). This simple property can be, in a sense, explored by determining two intersections of two out of three bisectors. Let these intersections be S and S' and let the distance between them be d. For any three points A, B, and C it will be $d = 0$. This simple example shows how one can use WinGCLC also for investigating some geometrical property or hypothesis.

```
% fixed points
point A 5 5
point B 50 5
point C 20 50

% determining bisectors
bis a B A C
bis b A B C
bis c A B A

% determining intersections of bisectors
intersec S a b
intersec S' a c
distance d S S'

% marking points
cmark_b A
cmark_b B
cmark_t C
cmark_t S

% drawing sides of the triangle ABC
drawsegment A B
drawsegment B C
drawsegment C A

% drawing the circle inscribed in ABC
line a1 B C
perp a2 S a1
intersec P a1 a2
drawcircle S P
```

Figure 1: Example 1

Example given in Fig. 3 illustrates the support for a direct access to Cartesian system. The output (in LaTeX format) is presented in Fig. 4. In this example we describe one conic (parabola) and one its tangent. This example illustrate how a rather complex figure can be described in only a few lines.

Figure 5 shows one hyperbolical construction represented in Poincare's disc model, while Fig. 6 shows an object which representation was imported from *JavaView* [8].

5 Objects Manipulations and Animations

WinGCLC provides a range of interactive functionalities. In addition to a syntax coloring editor, tools for processing picture descriptions and locating errors, tools (*watch window*) for exploring values of selected objects in a construction (so WinGCLC can work as a *geometrical calculator*) there are also tools for easy and interactive moving of fixed points, updating pictures and making animations. (Animations and traced points can be defined both interactively and via GCLC commands.)

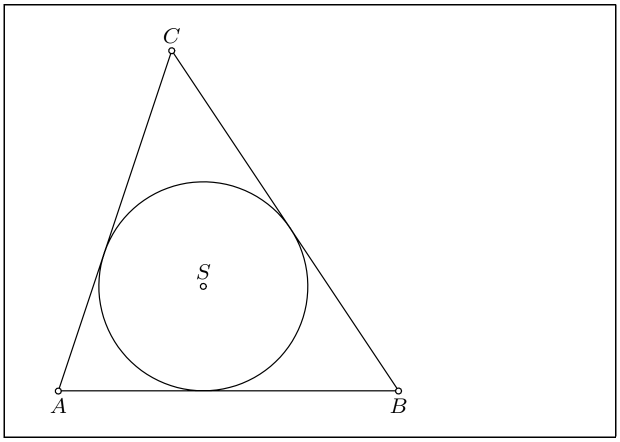

Figure 2: Output for Example 1

```
% define and draw Cartesian axis
ang_picture 0 0 50 50
ang_origin 20 20
ang_unit 8
ang_drawsystem

% define a conic
ang_conic h 0 0 1 -1 0 -3

% construct a point P on the conic
ang_point A1 2 2
ang_point A2 3 2
line l A1 A2
ang_intersec2 P P2 h l
cmark_t P

ang_tangent p P h
ang_drawline p
ang_drawconic h
```

Figure 3: Example 2

An animation is defined as a formal construction with a set of fixed points which linearly move from an initial to a destination position. The user moves points interactively and can define the total number of frames and the number of frames per second. Once an animation is generated, it can be played or explored frame by frame. Within the animation, the user can select some points (given or constructed ones) and explore all their positions. All positions of one selected point make so called *trace* (drawn in a selected color). Obviously, a *trace* can be studied as a corresponding *locus*. These features can be essential in teaching geometry, but can also help studying geometry or even guide some research in geometry (with WinGCLC serving as a machine assistant).

Figures 7 and 8 illustrate some of the above tools and devices (traces, animations, *watch windows*, etc.)

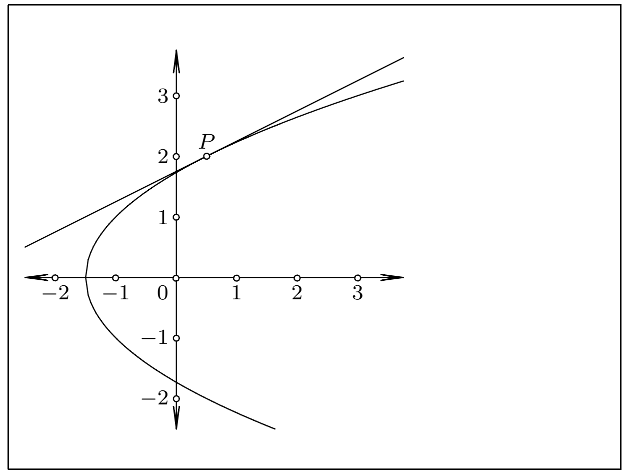

Figure 4: Output for Example 2

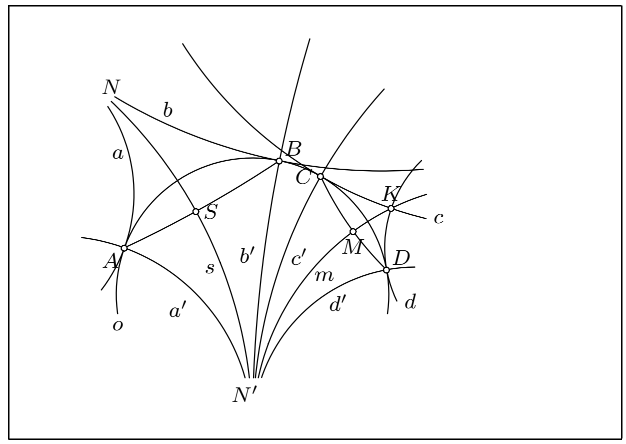

Figure 5: Example 3

6 Additional Modules

In addition to WinGCLC (and its command-line version GCLC with a screen previewer), there are also some additional GCLC modules. Some of them are:

- HYP-EUC (made by the second author): this module transforms an abstract hyperbolical construction into an abstract Euclidean construction in Poincare's disc model of a hyperbolical plane. The corresponding Euclidean construction (in GCLC) can then be processed in a usual way. By using this module, GCLC works as a platform for both Euclidean and hyperbolical geometry. It is interesting to investigate the same abstract construction in two geometries: Euclidean and hyperbolical one.

- JV2GCL: a converter from *JavaView* [8] `.jvx` format to GCLC format[2] This module links GCLC

[2]The JV2GCL module was made by the first author during his visit to Konrad Polthier's group at Mathematical Institute of TU Berlin. This visit was funded by DAAD.

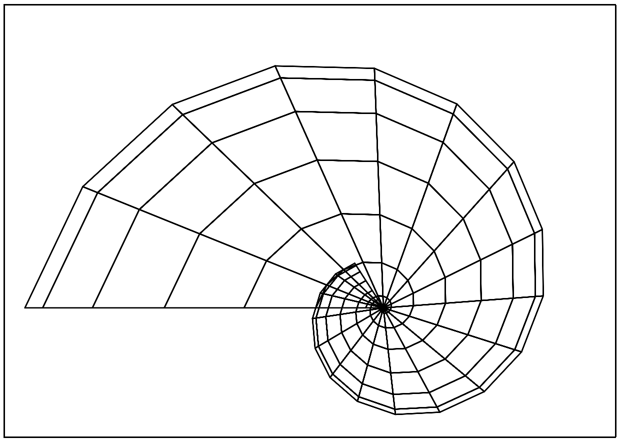

Figure 6: Example 4

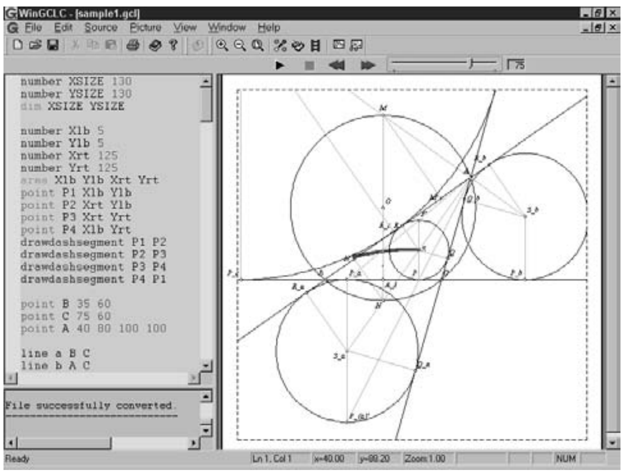

Figure 7: One screenshot of WinGCLC

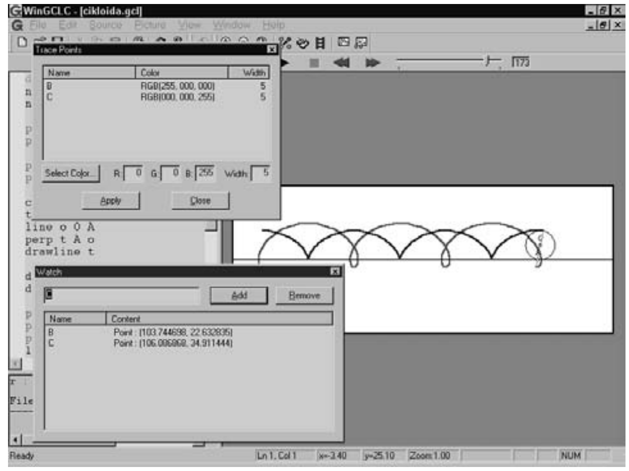

Figure 8: *Trace* and *watch* windows with cycloid described in WinGCLC

with the powerful *JavaView* package. *JavaView* pictures can be imported and further processed.

- a converter from a natural-language description of a construction into a GCLC form (made by Ivan Elčić, under the supervision of the first author). This module can process one restricted fragment of natural language used in describing constructions. This fragment flexibly covers a range of typical discourse structures that can be naturally transformed into a formal description (in GCLC language).

We are planning to develop new additional modules (for descriptive geometry, projective geometry etc.), so WinGCLC could work as a native platform for a range of geometries.

7 Export Formats

The current version of WinGCLC can produce pictures in two formats: LaTeX format and bitmap format.

Pictures in LaTeX can be included directly in LaTeX documents and they use LaTeX fonts which is essential for good looking figures in LaTeX documents (and this is a problem for many similar tools).

Pictures in bitmap format are suitable for different conversions and processing. GCLC pictures in bitmap format can also use colors.

Although the above two picture formats have their advantages, GCLC figures are normally stored in their original, source form. This form is not only precise (and sufficient for producing pictures), but also very concise[3].

8 Related Work

WinGCLC is related to a family of similar tools. However, most often, they focus on either producing digital illustrations (e.g., TeXCAD and other tools for TeX/LaTeX) or on teaching geometry (e.g., Cinderella [9]). We believe that some of the advantages of WinGCLC (comparing to other tools) are its simplicity, small size of the program[4] and its output files, output files natively supported by LaTeX, effective and illustrative devices such as animations and traces, it is freely available etc. One of the main novelty and characteristics of the system is its focusing on explicitly and formally describing figures (instead of drawing figures) and thus, focusing on *meaning* rather than on *layout* of the figure.

[3] All figures from a university book with 120 illustrated geometrical problems [4] have together (in the uncompressed, GCLC form) less than 130Kb.

[4] GCLC programs are very small in size: the command line version of GCLC is less than 100Kb, WinGCLC is around 200Kb.

9 Further Work

We are planning to develop new additional modules (for descriptive geometry, projective geometry etc.) We are also planning to extend the GCLC language by a support for symbolically given functions and for user-defined functions. We are also planning to add an interactive design mode which would enable constructing some objects more easily.

We are planning to release a version of WinGCLC for Linux.[5].

We are considering building-in some geometry theorem prover into WinGCLC. It is known that elementary geometry is decidable [10] and some of existing theorem provers for fragments of geometry [5, 1] would make WinGCLC a compact workbench for teaching, studying geometry, doing research in geometry, and also producing digital illustrations.

10 Conclusions

In this paper we presented the system WinGCLC. This is an easy-to-use system which can be used in teaching geometry, producing mathematical illustrations or even as a guidance in problem solving or in some pieces of research. The system is based on the idea of formally describing figures, so it is close to mathematical way of thinking. WinGCLC can produce pictures in LaTeX and bitmap formats. WinGCLC provides a support for interactive work, making animations, traces, etc.

The system is publicly available and is already being used by a number of mathematicians. We are planning to maintain and further improve the system.

Acknowledgments This work was supported by the Serbian Ministry of Science research grant 1646. We are grateful to prof. Neda Bokan and other members of the Group for geometry, education and visualization with applications (mostly based at the Faculty of Mathematics, University of Belgrade) for their invaluable support in developing the WinGCLC package.

References

[1] Shang-Ching Chou. *Mechanical Geometry Theorem Proving.* D.Reidel Publishing Company, Dordrecht, 1988.

[2] Euclid. *Elements.* (translation into Serbian: *Elementi*, Naučna knjiga, 1957, Belgrade).

[3] David Hilbert. *Grundlagen der Geometrie.* 1899. (translation into Serbian: *Osnovi geometrije*, Naučno delo, 1957, Belgrade).

[4] Predrag Janičić. *Zbirka zadataka iz geometrije.* Skripta Internacional, Beograd, 3rd edition 1999. Collection of problems in geometry (in Serbian).

[5] Predrag Janičić and Stevan Kordić. EUCLID — the geometry theorem prover. *FILOMAT*, 9(3):723–732, 1995.

[6] Donald Knuth. *TeXBook.* Addison Wesley Professional, 1986.

[7] Leslie Lamport. *LaTeX: A Document Preparation System.* Addison Wesley Professional, 1994.

[8] Konrad Polthier. JavaView. on-line at: http://www-sfb288.math.tu-berlin.de/vgp/javaview/index.html.

[9] Jürgen Richter-Gebert and Ulrich Kortenkamp. Cinderella. on-line at: http://www.cinderella.de.

[10] A. Tarski, A. Mostowski, and Robinson R. M. *Undecidable Theories.* North Holland, 1953.

[5]Currently there are command-line versions of GCLC for DOS/Windows and for Linux and WinGCLC for MS Windows