

Design decisions in interactive geometry software

Kate Mackrell

Accepted: 13 March 2011 / Published online: 5 April 2011
© FIZ Karlsruhe 2011

Abstract Software design is increasingly being recognized as an important factor in student learning when interacting with interactive geometry software (IGS). A categorization of the operations possible within an IGS is used to identify and analyse design decisions made in a number of current IGS programs: Cabri II Plus, Cabri 3D, Cinderella, GeoGebra and Geometer's Sketchpad. The analysis, in the context of exploring the area of a circle, is focused on construction, dragging, and alternative spatial and semantic views. A wide diversity of both design issues and individual design decisions was identified, illustrating both the scarcity of research in this area and a number of inevitable tensions, such as between functionality and complexity, and between static and dynamic geometry, related to which future research questions might be posed.

1 Introduction

The first interactive geometry software (IGS) programs were Cabri-Géomètre in 1989 and Geometer's Sketchpad (GSP) in 1991 (see Laborde & Laborde, 2008 and Goldberg, Scher and Feurzeig, 2008, for the history of these programs). A large number of IGS programs have been developed subsequently, and interactive geometry has also been incorporated into programs with a broader scope, such as TI-Nspire and Autograph.

Laborde (2007), in a summary of the research on the use of IGS in the classroom, comments on the increasing awareness that learning outcomes depend not only upon tasks and pedagogical context, but also on software design.

This awareness is also evidenced by the recent ICMI study 17 on mathematics education and technology, in which one out of five sections was devoted to issues of design (Hoyles & Lagrange, 2010).

Jackiw identifies the importance of design as follows:

Design certainly acts as the first doorway and first doorkeeper to any deeper curricular or epistemological innovation an educational technology might offer. For it is not at the structural level, but rather on the surface—at the designed interface—that users interact with technologies, that meanings are negotiated; that cognitive, psychological, educational and social transformation may, or may not, occur. (Butler, Jackiw, Laborde, Lagrange & Yerushalmy, 2009, p. 432)

With the aim of clarifying design issues in IGS, this paper uses the possible interactions in a visual mathematics representation (VMR) identified by Sedig and Sumner (2006) to develop a categorization of the operations that can take place in an IGS program. An exploration of the design decisions involved in several fundamental operations is then undertaken using the current versions of the two original programs [Cabri II Plus v1.4.3 (Laborde et al., 2003, Laborde & Laborde, 2008), Cabri 3D v2.1.2 (Bainville & Laborde, 2004, 2007), GSP v5.01 (Jackiw, 2009)] and two more recent programs, Cinderella v2.1 (Richter-Gebert & Kortenkamp, 2007, 2010) and GeoGebra v3.2.46 (Hohenwarter et al., 2002, 2010).

2 A framework

Although there is little published information concerning decisions made on interface design in GeoGebra, the designers of Cabri, GSP and Cinderella have published at

K. Mackrell (✉)
Institute of Education, University of London, London, UK
e-mail: kmackrell@ioe.ac.uk

length about such decisions (e.g. Laborde & Laborde, 2008; Butler, Jackiw, Laborde, Lagrange & Yerushalmy, 2009; Richter-Geibert & Kortenkamp, 2010). The designers focus on particular decisions, however, and do not provide any framework within which such decisions may be interrelated. Such a framework was hence created.

Sedig & Sumner (2006) developed a categorization of interactions possible in a VMR from an analysis of a range of professional and educational mathematics software programs, including GSP. This categorization was used to identify possible interactions within an IGS. Two categories are given below: see Appendix A for a list of other categories.

Interaction	Description	In IGS
Composing	Putting together separate visual elements to create a VMR	Constructing Making custom tools
Repicturing	Displaying a VMR in an alternative manner	Spatially: dragging, changing axes
	Spatially: VMR or space in which it exists goes through geometric transformation	Semantically: showing figure descriptions
	Semantically: VMR displayed using alternative representational form	Aesthetically: changing colour, etc.
	Aesthetically: colour, etc.	

Fundamental operations in IGS were then identified. Such operations involve one or more types of interaction and are characterized by the purpose, aspect of the IGS involved, and the result.

Some basic terms: the screen in IGS fundamentally consists of an area containing tools (by which actions are performed), a page containing objects, which are the results of actions, and (optionally) an area containing further information. Objects are anything upon which operations may be performed, and can be geometric, numerical, textual, graphical, algebraic, etc.

The basic operations are as follows:

a. Construction operations

Tools take as input previously existing objects (or points created as the tool is used) and output new objects, normally displayed on the page. The interactions involved are composing, but also probing, when this results in a new object such as a measurement. Construction operations include creation, relative construction, transformation, measurement, tabulation, calculation, graphs, and algebra.

b. Object operations

These are distinguished from construction in that the result is a change in an existing object, rather than the

creation of a new object. Possible interactions are annotating, rearranging, repicturing, filtering, and cutting. Object operations include labelling, redefining (which does not correspond to any of Sedig and Sumner's interactions), and dragging.

c. Tool operations

These result in changes to the tools available to the user. Interactions are filtering (suppressing tools), chunking (rearranging tools), and composing (creating new tools).

d. View operations

These change the way in which the geometric page is viewed or display alternative views. Interactions include probing (panning and zooming), spatial repicturing (changing the viewing angle, the underlying geometry, or the coordinate axes), and semantic repicturing (viewing figure descriptions).

e. Information operations

An information operation is any probing interaction that does not produce an object for further interaction and that does not change the view. An example is using a contextual menu to access information about the object, or displaying information about the relationship between objects.

f. Replay operations

These correspond to scoping interactions and consist of replaying construction, undoing, and redoing.

g. Export operations

In these operations (save, print, save screenshot, export to HTML, etc.) the page is represented in a different media. These do not correspond to any of Sedig and Sumner's (2006) interactions.

h. Page operations

These, found only in GSP, enable multiple pages in a single document and do not correspond to any of Sedig and Sumner's (2006) interactions.

3 Methodology

To identify design issues most closely related to practice, a task, exploring the area of a circle, was performed by the researcher with each program. This task involved creating and measuring a circle, dragging, calculating, tabulating and then plotting data, creating a graph from the circle measurements, fitting an algebraically defined curve, and substituting to test a formula.

A surprising diversity of design differences and issues arose simply in creation, measurement, dragging and

interpreting the results, and hence the decision was made to limit this paper to issues arising in these stages of the task, and correspondingly to some aspects of construction, object, and view operations.

An early draft of the paper was sent to each of the software designers for comment.

4 Construction operations

In exploring the area of a circle, the first construction operations created a circle and a segment to represent its radius, and measured the area and radius of the circle.

There are four aspects to a construction operation: deciding on an appropriate tool, locating the tool, using the tool, and interpreting the result.

4.1 Deciding on an appropriate tool

The fundamental design decision here is the tools that a program will provide. Each program provides a variety of possibilities for creating a circle and hence the circle tools will be used for illustration. See Appendix B for a list of these tools.

A tool is considered to be a means of using given input to create output. Tools will be considered to be distinct if they involve different types of input, different selection processes, or different types of output, where input and output are defined by both static and dynamic properties. For example, two circles of the same radius and centre, one defined by centre and radius point and the other by centre and numerical radius, will have different behaviour when their centres are dragged and are hence not considered to be the same type of object. Distinct tools will be considered to have the same functionality if the same input creates the same output.

There are a number of possible design decisions about the relationship between tools and functionalities. One possibility is to allow distinct tools to have the same functionality. Multiple representation of functionality occurs in Cinderella, where almost all tools appear both as a menu command and as a toolbar icon. Tools with the same functionality may also need to be used differently. In GSP, a circle by centre and radius point may be constructed either through first selecting centre and radius points and then selecting a menu command, or by first selecting a toolbar icon and then selecting (or creating) centre and radius points. A third possibility is that tools may require input to be specified differently. GeoGebra and Cinderella each have tools of identical functionality, one of which requires inputting an object's name and another which requires clicking on the object with the mouse. When circle tools with the same functionality are discounted, the

original 1 (Cabri 3D), 2 (Cabri II plus), 3 (GSP), 5 (GeoGebra), and 11 (Cinderella) are reduced to 2 (GSP), 4 (GeoGebra), and 5 (Cinderella).

Tools may have multiple functionalities, whereby different types of input result in different types of output. For example, there are six basic circle-creation functionalities represented among the programs (see Appendix B) and the Cabri 3D circle tool has five of these. The functionalities available now become five (Cabri 3D), four (Cabri II plus), three (GSP), four (GeoGebra) and five (Cinderella). The effect of tools with multiple functionalities has not yet been researched. Single-functional tools are easier to document, with more clearly defined initial objects, but multifunctional tools enable a less cluttered interface and also alleviate the problem of needing to decide between a number of similar icons in choosing a tool (Kortenkamp & Dohrmann, 2010, p. 61). Single-functional tools may blur conceptual similarities: multifunctional tools may blur conceptual differences.

Further decisions follow from a distinction between two types of functionality. Goldenberg, Scher & Feurzeig (2008, p. 68) describe how, given an initial wish list of 400 functionalities for GSP, the decision was made to pare this down to "atoms". A tool is hence defined as having atomic functionality if it transforms input in a way which cannot be achieved by any other tool or sequence of tools which do not have the same functionality as the original tool. For example, the tool which creates a circle by centre and radius point has atomic functionality. A tool has molecular functionality if its output can be produced from the given input by an operation or series of operations using tools with atomic functionality. For example, the tool which creates a circle, given a centre point and a segment for radius length, has molecular functionality, as it may be replaced by first using the atomic tool that measures the length of a segment, and then by using the atomic tool that constructs a circle given a centre and numerical radius.

In geometric object creation and measurement, tools with atomic functionality roughly correspond to straight-edge, compass, ruler, and protractor. Tools with molecular functionality involve standard Euclidean constructions (such as a circle through three points, or a perpendicular line) and measurements that require calculation (such as the area of a circle) and may be supplied by the program or created by the user. Of the six circle functionalities, three are atomic. Cabri II Plus and Cinderella have all three atomic functionalities, while Cabri 3D, GSP, and GeoGebra each have two.

Making available tools with new atomic functionality increases the affordance of a program, sometimes dramatically as with the introduction of measurement, and sometimes only subtly, as can be seen by the list of circle

functionalities. Making available tools with new molecular functionality does not directly affect the affordance of a program.

There are hence two fundamental decisions concerning the provision of tools. The first of these is the affordances that a program will have and the atomic tools that will enable these affordances. This decision is beyond the scope of this paper, as each of the programs has the required affordances to complete the stages of the circle task here reported on. However, decisions regarding affordances become critical in comparing the more advanced functionality of the programs.

The second decision is which molecular tools to provide in addition to atomic tools. Molecular functionalities have a large impact on the way in which a program can be used and hence no IGS restricts itself to atomic tools. One reason is the complexity of performing an operation. For example, the area of a triangle may be found by constructing and measuring an altitude, and then geometrically multiplying this number by the base length, but this requires a lengthy construction and also knowledge of the formula for the area of a triangle. All of the programs hence include both calculation and area tools.

Molecular tools also encapsulate fundamental relationships and constructions. It is straightforward to construct the midpoint of a segment, or a line perpendicular to a given line using atomic tools, but the molecular tools available speed up construction, suppress irrelevant and distracting detail, and enable a focus on the “big ideas” (Goldenberg, Scher & Feurzeig, 2008, p. 67).

Mathematical language is also a consideration. By using a tool such as the angle bisector, learners may at the same time encounter the correct mathematical language for a concept and explore its meaning. GSP has five different tools for measuring length in order to introduce the correct mathematical language for each measurement (Steketee, personal communication, 13 September 2010).

Another reason is to enhance the perceived affordance of a program. Sedig and Sumner (2006, p. 6) distinguish between real affordance, which in IGS construction operations comes from the scope of the atomic tools, and perceived affordance. Many users, unaware of the distinction between atomic and molecular tools and unable to create or locate custom tools, may not be aware that a program has a particular affordance unless this is made explicit in the form of a tool.

There are also pedagogical reasons for limiting the molecular tools available. One of these is to focus the learner. For example, students were encouraged to use transformation tools to create a parallelogram by suppressing the Parallel and Compass tools in Cabri II Plus (Laborde, 2007, p. 80). Another reason is to encourage learners to build their own tools after thinking through the

geometry in a construction (Goldenberg, Scher & Feurzeig, 2008, p. 68). In general, there is not a direct relationship between easily accessible functionality and pedagogical value.

4.2 Finding the tool

One of the main issues in finding the tool is that the number of tools affects the complexity of the interface. Kortenkamp and Dohrmann (2010, p. 59) highlight the tension between a wide functional range and usability. Kittel (cited in Kortenkamp & Dohrmann) found that a large number of tools were often overwhelming for students and teachers, who preferred to use an interface which did not contain the distraction of unused tools. This is another reason for restricting the number of molecular tools.

Schimpf and Spannagel (2010) suggested that complexity can be reduced by either structuring or removing functionality, but found that reducing the number of icons from 48 to 23 did not significantly affect the time taken to locate icons. The icons were meaningless to subjects, however; meaningful icons might provide conceptual as well as visual distraction. Their result shows, however, that it is simplistic to assume that the tension between functionality and complexity can be easily alleviated by reorganizing or suppressing tools, as is possible with the Cabris and GeoGebra.

One decision in locating tools is whether to use toolbars with pictorial icons, or menus with names, or text input. In the Cabris and GeoGebra, tools are organized into toolboxes on toolbars, while in Cinderella and GSP tools are in a combination of menus and toolbars. Cinderella and GeoGebra also have text entry boxes. Kortenkamp and Dohrmann (2010, p. 61) note that a similar action is required to select a tool from a toolbox or from a drop-down menu. Menu titles are easier to interpret than icons, but a menu item, particularly if in a cascading menu such as that used by Cinderella, may take longer to access than a tool from a toolbox, where the most recently activated tool may be reactivated with a single click. The use of both toolbars and menus may also be more difficult for users than the use of one or the other.

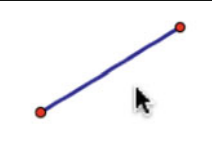
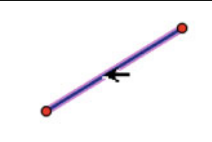
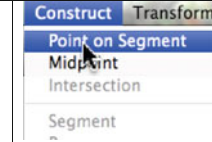
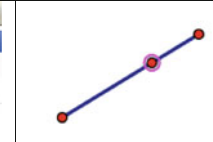
A further decision is how to group tools. Grouping may impact on both the ability to locate a tool (as suggested, but not explored, by Schimpf and Spannagel (2010)) and the conceptual understanding of the tool. All programs group transformation and measurement tools, but the classification of geometric tools is surprisingly problematic. Classifying by resultant object results in lines through two points and lines parallel to a given line being grouped together, leaving out a fundamental difference in the relationships used in constructing the two lines. Classifying by type of construction is equally problematic, as criteria to

distinguish free construction (such as a circle by centre and radius point) from relative construction (such as a line perpendicular to another line) are not straightforward. Cinderella classifies by resultant object, while the other programs use a mixture of resultant object and type of construction.

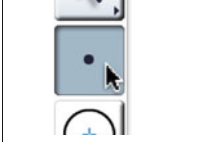
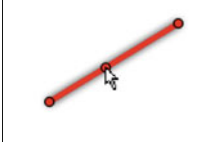
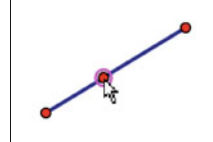
4.3 Using the tool

“When defining new elements, there are two fundamentally different approaches: (a) first choose the objects, then an action using these objects [O/A], or (b) first choose an action, then choose the objects used for this action [A/O]” (Kortenkamp & Dohrmann, 2010, p. 60). The difference between these is illustrated in the following GSP screenshots:

Object – Action:

			
Initial object	Choose the object	Choose the action	Result

Action – Object:

			
Initial object	Choose the action	Choose the object	Result

This design decision has been identified as a major difference between Cabri (A/O) and GSP (O/A). Jackiw states that the two programs “differ absolutely in the most basic mathematical language they offer for summoning objects into existence” (Butler, Jackiw, Laborde, Lagrange, & Yerushalmy, 2009, p. 432). However, each of the programs uses both of these options at different times. The Cabris use O/A when changing the appearance and some properties of an object. The GSP toolbar uses A/O tools (including custom tools) and some of the number and graph menu commands are also A/O. Several GSP tools also involve O/A/O: selecting an object, specifying the action, and then making a further selection of objects from a dialogue box or the page. GeoGebra follows the Cabris, but in Cinderella most tools will allow the objects and tool to be chosen in any order.

One of the fundamental considerations involved in the choice of A/O or O/A is the impact of the choice on an operation in itself. An argument commonly made for O/A is that it follows the conventions of most word-processing programs. O/A also fits with recommended user interface guidelines to avoid modal states, or states in which input will produce different results (Steketee, personal communication, 13 September 2010). However, when Blanc-Brude (cited in Laborde & Laborde, 2008) compared the use of GSP and Cabri in construction tasks, no speed differences were found in basic constructions, but planning time was longer for more complex constructions using GSP. Laborde and Laborde conjecture that this might be due to consistent A/O in Cabri versus the change from A/O to O/A for more complex constructions in GSP. In any IGS program, A/O can be consistently used for all construction,

but O/A cannot be used for operations such as free point creation and calculation. They also stress that functional dependencies (e.g. perpendicular) constitute the meaning of objects to be constructed, and that relations are invariant, while objects are variable. The user may hence see the goal of the action as expressed by the relation and only secondarily by the objects involved.

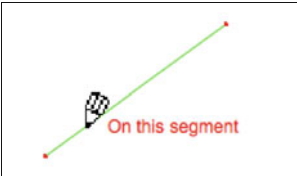
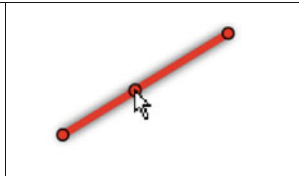
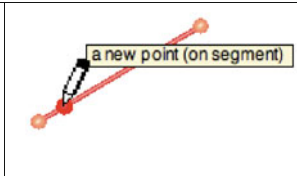
Both have advantages in construction: if objects are automatically selected as created, O/A can allow a quick sequence of actions. It also allows the same action to be applied to multiple objects. A/O is useful when the same action is to be applied to different groups of objects. The more flexible selection order of Cinderella has the advantages of both, but can be confusing initially.

The second fundamental consideration is the consequences for other potential program affordances. A/O

makes possible points on the fly, in which points (free, on objects, or at intersections) may be created in the course of using a tool. This capability may significantly cut down on both the time and complexity involved in using a tool.

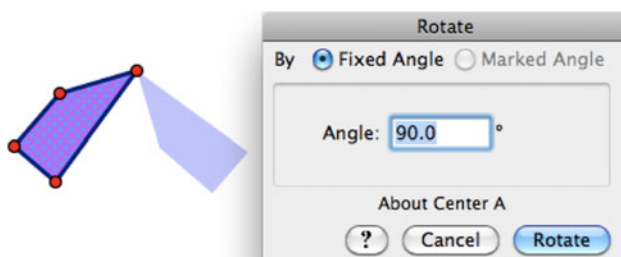
Another consequence is the provision of tool help. A/O enables a description to be given once a tool is chosen and as objects are selected. In O/A construction, the only feedback that can be given is an indication that the tool may or may not be used with the objects selected, and the way in which the tool can be used needs to be documented separately.

A/O also enables each of the programs to give hints as to which objects may be selected as the cursor moves over the page. These take two forms: text tooltips and highlighting, as illustrated below in creating a point on a segment.

		
Tooltip (Cabri II Plus)	Highlighting (GSP)	Tooltip and highlighting (Cabri 3D)

Each of the programs uses one or other of these with its A/O tools. Highlighting is used in all but Cabri II Plus, and is the most noticeable in GSP and least noticeable in GeoGebra. Tooltips are used in all but GSP and range from the identification of all objects as the cursor passes over in GeoGebra to the identification only of objects relevant to the construction, together with the role which the object could play in Cabri 3D. Laborde and Laborde (2008, p. 35) state that the aim here was to implement a user-centred interface, with feedback messages to scaffold a user's actions.

A/O and O/A/O selection also enable graphic previews, in which, as soon as enough objects are selected, a preview is given of the object that will result. In the above screenshots of a point being created on a segment, GSP and Cabri 3D give a graphic preview of the point, but Cabri II Plus does not. A more vivid example is shown below using the GSP Rotate tool. The rotated object is not constructed until the Rotate button is clicked: the angle may still be changed.



A problem with graphic previews (apart from the GSP transformation tools) is that they are visually identical to the completed construction, which may cause users to select the next tool rather than select the objects required to finish the construction.

One particular consequence of A/O tool help and tooltips is that when the order in which objects are selected affects the result (for example, creating the circle by centre and radius point), this order can be specified as the tool is used. In contrast, in O/A this order is not visible to the user (Kortenkamp & Dohrmann, 2010, p. 60), particularly as objects can be selected simultaneously.

One further design decision that needs to be made concerning tool use is the way in which one of a number of coincident or nearly coincident objects may be selected.

The different design decisions made illustrate the difference between user choice and default behaviour. In Cabri II Plus, the user is required to make a choice whenever objects are sufficiently close together. In the other programs, the most recently created object is selected by default. The decisions also illustrate the tension between functionality and complexity. User choice is the most flexible option, but the requirement to make a choice may confuse and distract. The default behaviour considerably simplifies tool use, but does not allow other objects to be selected. To change a default, the user must first be aware that it can be changed and then must be able to perform the operations necessary to change it. Otherwise, users may stay with the default even when other options are preferable. In this case, in none of the programs is the possibility of an alternative choice both clearly indicated and easy to perform.

4.4 Interpreting the result

Appendix C shows part of the page for each object after creating a circle by centre and radius point, creating a segment between the centre and radius point, and measuring the area of the circle and the distance between the centre and radius point. Note that in GSP and Cinderella, objects are selected as they are created (not shown in the screenshots). This focuses user attention on the result of using a tool and facilitates appearance changes or further construction using the resultant object.

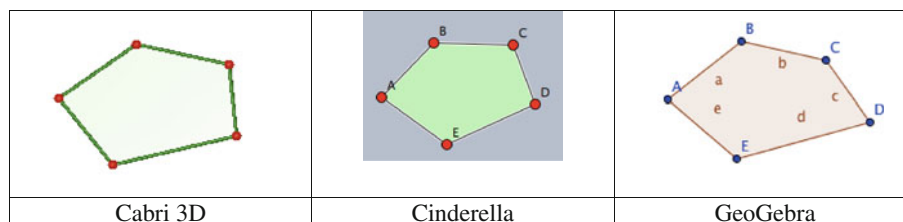
Two areas in which design decisions are necessary can be seen in the screenshots. One of these is the calculation and display of measurements, discussed in this section, while the other is the display of labels, to be discussed in the next section.

The measurement of area provides an example of a design decision that is not always visible to the user: a number of different algorithms may be used to calculate area. In the case of the circle, each of the programs will give the same result, but in some circumstances, such as a “crossed quadrilateral” that does not have a conventionally defined area, each program gives a different result.

Measurement units and accuracy also involve decisions regarding default behaviour and illustrate that defaults may be changed in three different ways: for individual objects, for all objects, and for all future objects. Units are a default which may only be changed for all objects. The Cabris and GSP initially give units in centimetres or degrees as appropriate, and GeoGebra and Cinderella give no units. Different units may be chosen for the Cabris, GSP, and Cinderella, but in GeoGebra, measurement units can only

5 Object operations

In any geometric diagram, it is useful to assign labels to geometric objects and measurements to be able to identify and refer to them. This is particularly important in IGS, where the position of objects is not fixed. Accordingly, the designer must decide which objects can be labelled and how the object operation of labelling takes place. Similar decisions have been made about which objects can be labelled: all measurements and all geometric objects (with the exception of Cinderella, where only points, lines, and segments may be labelled). A major difference, however, is that in the Cabris, all labelling must be performed by the user, while the others use some form of default labelling, ranging from labelling all objects and parts of an object (GeoGebra) to assigning labels to objects only when they are measured (GSP). Such defaults may be changed for individual objects or all objects, however, and all labels may be edited. The screenshots below illustrate the tension between information and clutter when a polygon is created and the default labelling options are used:



be displayed on axes rather than on individual measurements. In contrast, all the programs have a default setting of two-decimal place accuracy, which may be changed globally for all programs, for individual measurements in all but GeoGebra, and for future measurements with the Cabris.

One particular design decision illustrates the impact of differences in mathematical convention. When a number is rounded to a certain number of decimal places, the convention in German-speaking countries is to leave off all trailing zeroes. Hence, in Cinderella and GeoGebra, 3.004 rounded to two decimal places is represented as 3. In contrast, GSP and the Cabris represent the number as 3.00: 3 would represent the integer 3 or the number rounded to the nearest whole number. A statement such as “ $3 \times 4 = 12.02$ ”, where all numbers are represented to the same degree of accuracy, is hence somewhat problematic. All the programs have been translated into a variety of languages, and a decision must be made as to the extent of translation that is appropriate: should conventions as well as mathematical terms and notation be changed?

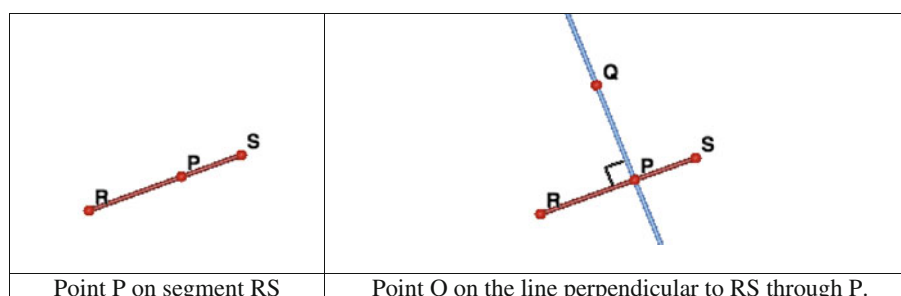
By requiring the active creation of labels, the Cabris avoid clutter; the amount of information displayed by GeoGebra may be confusing. However, the other programs automatically give information that the inexperienced user may not think of providing and which may later be important in interpreting and referring to the figure.

The other important object operation in the circle task is dragging. In an IGS environment, dragging in the generic sense of moving an object on a computer screen with a mouse includes Sedig and Sumner’s (2006) categories of rearranging (when moving non-geometric objects such as text, calculations, or measurements or moving geometric objects without changing their appearance) and refiguring, when movement of a geometric object causes a change in its appearance. In light of the extensive literature on dragging in IGS, however, dragging will be restricted here to refer to any movement of a geometric object by means of the mouse, whether or not this motion involves refiguring, and will not be used for the movement of non-geometric objects. The defining feature of an IGS is that when dragging takes place, the appearance of an object may

change, but the relationships involved in its creation are preserved.

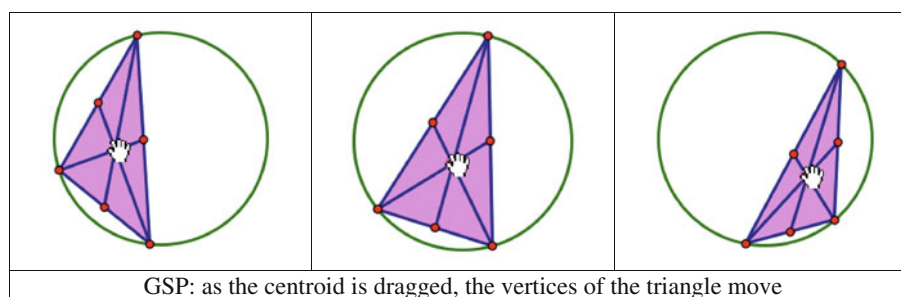
One of the more fascinating aspects of interactive geometry is that the behaviour of objects when dragged cannot be fully determined by Euclidean geometry (or, optionally with Cinderella, non-Euclidean geometry). This necessitates decisions on the part of the designer (Goldenberg, Scher, & Feurzeig, 2008). One example is the behaviour of a point on an object. In all programs, the decision was made that a point P on a segment RS would continue to divide the segment in the same proportion as the segment changes (Jones, Mackrell, & Stevenson, 2009, p. 51).

It is also necessary to decide which objects can be dragged. In all programs, free points, points on objects, and any object created entirely from free points may be dragged. However, in GSP, every object may be dragged. The aim was to never suggest a static state (Goldenberg, Scher, & Feurzeig, 2008, p. 59), and also to allow reversibility: if a dependent object has a simple enough relationship with an independent object, such as the reflection of a point in a line, then dragging it will move the original object to maintain the relationship. For example, dragging the centroid of a triangle with vertices on a circle causes the vertices of the triangle to move



In contrast, when a line is constructed through P perpendicular to the segment and a point Q is created on the line as shown above, different decisions were made as to the position of Q as the length of the segment RS changes.

(Goldenberg et al., 2008, p. 69), as shown below. If the relationship is more complex, then dragging will translate all aspects of the figure to which the dragged object is related.



In the Cabris, the distance PQ is independent of the length of the segment; in GeoGebra and GSP the distance PQ is proportional to the length of the segment; and in Cinderella, the distance PQ depends on the slope rather than the length of the segment. Another issue is that dragging may be reversible or have continuous results, but not both (Kortenkamp, 1999), with the result, for example, that the intersection points between two circles will either jump unexpectedly, or not return to their original positions when dragging is reversed, dependent on the decision of the designer. There is an important pedagogical implication here: design choices rather than geometry may affect the invariants constructed by students.

Reversibility may be problematic, however. In the example above, the position of the centroid does not entirely determine the position of the vertices, and hence rules from outside geometry must be used to select among the non-unique solutions, making the results less predictable for the user (Goldenberg et al., 2008, p. 75). Reversibility also does not maintain the pedagogically valuable distinction between free and dependent objects. Falcade et al. (2007), for example, used the distinction to introduce dependent and independent variables when using Cabri to explore the concept of a function.

Note that in all of the programs in which dragging is limited, it is possible to visually identify which points may

be dragged, either by appearance (Cinderella and GeoGebra) or by causing such points to flash (the Cabris).

In each program, the decision has been made that dragging may occur at almost any time, but very different methods are used to implement this. All programs have a specific selection tool, which when active allows left-click dragging of objects. In GSP, the prevalence of O/A means that the selection tool is almost always active. In the Cabris, almost all active tools limit which objects may be selected, but allow any draggable object to be dragged without being selected. In GeoGebra and Cinderella, right-click dragging is possible when the selection tool is not active. One implication is that it is not necessary to avoid modal states in order to allow dragging to take place at any time. A question is whether requiring a change in the

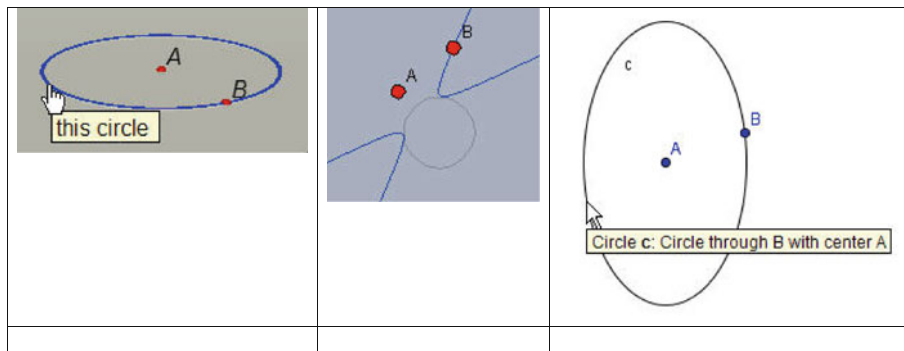
interpreting a static figure: might this have been different in an environment with different dragging behaviour?

6 View operations

Changing the view will reveal further significant design decisions and also enable a deeper examination of the results of the above construction and object operations.

6.1 A spatial change

Here are some screenshots of a variety of circles created from centre and radius points after particular changes to the view.



method of dragging might mitigate against its use when the selection tool is not active.

A further decision is whether objects attached to a geometric object will remain attached when it moves. This has particular significance for measurement: attached measurements may give more immediate feedback about the effect of dragging than measurements in an unrelated location on the screen. When created, GSP measurements are not attached (but may later be attached); in the other programs, measurements are attached (with the exception of area measurements in GeoGebra) but may later be detached.

Laborde (2007, p. 73) gives a summary of the research on student use of dragging and the ways in which such use may be categorized. However, there has not been much consideration of the effects of the differences in dragging in different IGS environments. For example, if all objects move when dragged, does this affect student understanding of the hierarchy of dependencies in a figure (Jones, 2000)? Talmon and Yerushalmy (2004) suggest that this may be the case and call for more research. Sinclair (2003) found with GSP that students were reluctant to drag and preferred

In Cabri 3D, the circle is seen at an oblique angle. The circle in Cinderella is represented in a model of hyperbolic geometry. The circle in GeoGebra was created simply by changing the scale on the y axis, however, and illustrates a major design decision about how geometric objects are to be represented on a screen. The following is based on Richter-Gebert (personal communication, 14 September 2010).

All interactive geometry depends on the internal built-in coordinate system of the screen; however, objects they are defined by the user are internally represented by coordinates. There are two choices that can be made about the way in which objects are defined.

One possibility is to use the underlying coordinate representation to create a visual simulation of a Euclidean plane. On this plane, distance is defined by a rigid ruler and location is not a fundamental property of objects. The user has no access to the internal coordinate representation of objects. A coordinate system may be introduced, but is not intrinsically linked to geometrically created objects: the system is simply another object on the plane, which may be used to give geometrically defined objects a relative

location and algebraic representation as appropriate, or to define objects algebraically. The behaviour of an object is dependent on the way in which it is defined: as axes change, a geometrically defined circle does not change in visual location or appearance, although its algebraic representation relative to the axes will change; however, an algebraically defined object will change in visual location or appearance, but not in algebraic representation. This is the position taken by the Cabris and GSP.

The second possibility is to reflect the internal geometry more directly: the plane is considered to be based on a coordinate system, to which all objects will automatically be referred. The circle is defined by using the coordinate locations of its centre and radius point to generate its equation. This equation is considered to be fundamental; if the axes change, then the equation is preserved but the appearance or location of the circle will change. This is the position taken by GeoGebra and Cinderella.

Both decisions give equivalent visual models of a Euclidean plane—provided that the axes used to define objects always have the same relative scale. In Cinderella, a deliberate decision was made to preserve the visual model: the axes can only be changed by zooming or translating and cannot be scaled independently. In contrast, in GeoGebra the axes can easily be independently scaled, with the consequence that the visual model breaks down. Instead, distances appear to depend on direction and angles appear to depend on orientation, which more advanced learners may find interesting to explore, but which may be highly problematic for learners still dependent on visual representation. What effect would it have on a 12- or 13-year-old learning about the area of a circle to see the circle change shape and apparent area as the axes change? The axes may be hidden to avoid such scaling, but this precludes the possibility of, for example, constructing a graph of the area of the circle against its radius.

6.2 A semantic change

The name “figure description” will be used for a list of objects and construction steps. In GSP, such a description needs to be created; in the others, it can simply be displayed. As GeoGebra’s algebra window acts as a figure description to some extent, this is shown together with other figure descriptions in Appendix C.

The main decisions for a figure description are the information that it shows and the way in which the information is organized and conveyed. In each description (apart from the GeoGebra algebra window, where it is optional), the name and geometric definition of each object are given.

Cinderella and GeoGebra, as appropriate to the decision described above to define objects relative to an accessible

coordinate system, also give information related to this coordinate system.¹ In Cinderella, this information, given in a column somewhat misleadingly headed “where?”, consists of the coordinates of the points, the equation of the circle, and values of measurements and text.

In GeoGebra, this information is called the value (in the construction protocol) and the algebraic representation (in the algebra window) of an object. “Value” and “algebraic representation” are not equivalent descriptions. Objects are equated with their value/algebraic representation, which is acceptable for measurements and text but is seriously misleading for segments. We hence have the statement “ $a = 1.44$ ”, where a is defined as the segment between points A and B. The first problem is that a geometric object is equated with one of the many measurements that can be made of this object. The second is that a name is used as a variable, a common error in the early stages of algebraic understanding (Küchemann, 1981), but also one that may persist into university and adversely affect performance in undergraduate mathematics courses (Gray, Loud & Sokolowski, 2005). This is a deliberate design decision, however: “the algebraic representation of a segment is its length” (Hohenwarter, personal communication, 9 September 2010).

The organization of objects depends upon the purpose of the figure description: if the description exists primarily to show how a figure was created, then objects are organized by order of creation. In contrast, the GeoGebra algebra window is designed specifically to give information about individual objects and to enable interaction with these objects; hence, objects are listed alphabetically. Both GSP and the GeoGebra algebra window also specifically divide objects into dependent and independent.

The quality of the description varies considerably: GSP and Cabri 3D give enough information to enable a student to reconstruct the figure, while the others are briefer and require further interpretation. Mackrell (2008) found that figure descriptions in Cabri 3D, together with the ability to replay construction, were effective in enabling students to learn new techniques: clarity may have been an important factor here. The construction protocol of GeoGebra is particularly complicated.

The figure descriptions also highlight a further design decision concerning the modes of possible interaction with objects. In the Cabris and GSP, objects are selected only by clicking on the screen and hence any labels given serve only to identify and describe objects. In contrast, in Cinderella and GeoGebra, an object may be selected by typing its name and hence an object has a name for selection and a label or caption, which may or may not be identical to the

¹ In the Cabris and GSP such information, not considered intrinsic to the definition of objects, can be obtained by measurement.

name, for identification and description. The conceptual difference between a name and a label may not be easy to grasp, and may complicate interpretation of the figure description, but the difference is essential for the text input on which GeoGebra is heavily reliant and the use of the inbuilt programming language in Cinderella.

Another decision is the extent of interaction that the figure description enables with the figure. Apart from GSP and the GeoGebra construction protocol, selecting an object on the page highlights it in the figure description and vice versa, making a clear link between objects and their description. In Cabri 3D and the GeoGebra algebra window, objects may be selected in the figure description for further use in construction and object operations, useful in a complex file with a large number of objects.

The figure description also gives a record of existing objects, which enables a problematic feature of GeoGebra to be identified. In all other programs, replicating the construction of an object, such as a segment, does not result in a new object being created: in GeoGebra any number of segments can be created between the same points, and each is listed as a separate object in the algebra window and when the cursor passes over the segment on the page. This is confusing and not mathematically acceptable: a fundamental axiom of Euclidean geometry is that there is only one straight line that passes through two points.

7 Evaluating design decisions

There is currently insufficient research evidence to evaluate most of the design decisions described here. It may be the case, for example, that O/A, with its consequent limitation of both functionality and information, is less useful pedagogically than A/O, or it may be that A/O, involving changes of modal state, is more problematic.

An exception, however, is where a design decision results in mathematics that is generally considered to be misleading or incorrect. Some issues arise from cultural differences in mathematical convention, such as the display of trailing zeros. Exposure to different conventions may or may not be useful pedagogically: this is a matter for further research.

GeoGebra alone among the programs contains mathematics that would universally be considered to be misleading or incorrect. One issue is the ease with which the visual model of the Euclidean plane breaks down when axes are scaled independently: it would be seriously misleading for a learner to conclude that length or angle depends on orientation, as would be entirely possible in such an environment. There are also two mathematical mistakes. It is unclear what the effect might be of being

able to create multiple identical objects, such as segments defined by two points. However, the confusion of values and algebraic representation, and in particular the use of a symbol to represent both the name of an object and a measure of its length, is likely to seriously affect the development of the concept of variable, crucial in mathematics. Given the increasing popularity of the program, such mistakes are a cause for concern.

8 Some provisos

I have used Cabri-Géomètre since 1993, GSP since 2001, and Cabri 3D since 2004 with secondary students, university students, and teachers, and have also developed a number of preconstructed files with each program. Prior to commencing this paper, I had attended several GeoGebra workshops and had used Cinderella to explore advanced transformations: most of what I know about these two programs was learnt in the course of researching this paper.

The design decisions identified and discussed are taken from the literature and my own general experience with IGS, together with what was noticed during the performance of a particular task. IGS programs are sophisticated pieces of software with many affordances that have not been discussed in the literature, and hence I cannot claim to have identified more than a selection of design decisions. In particular, limiting the paper to decisions arising in a relatively simple situation means that only three of the eight basic operations have been explored and important decisions regarding transformational, numerical, and algebraic affordances have been omitted entirely, as have unique advanced affordances such as the 3D capabilities of Cabri 3D and the programming capabilities of Cinderella.

A further proviso is that design decisions are affected by more than pedagogical considerations. A decision may well need to be maintained both because of the cost of changing it and in order not to upset the schemes of use of experienced users. The consequence is that the design decisions described here may well no longer be considered by the designers to be the best decisions possible. Each of the programs is also evolving: a particular feature may be omitted as a deliberate design decision or may simply not yet be implemented.

9 Conclusion

Despite the limitations described above, this study has shown a great diversity in both the design decisions that

need to be made and in the decisions made by individual designers. It is surprising, for example, that a large number of issues and possibilities emerge from simply exploring the circle tools available. This diversity is an indication that creating an IGS program is not simply a matter of representing the conventions of static Euclidean geometry on a screen, but is dependent on the epistemology of the designer and is influenced by both cultural conventions and pedagogical considerations.

A further consequence of this diversity of decisions, however, is to emphasize the very limited amount of research that has been conducted regarding their impact, even for well-known decisions such as selection order and the draggability of objects. A large number of questions for further research can hence be formulated. Some of these are straightforward, such as the effect on student understanding of the way in which measurements and units are displayed. More interesting questions arise, however, in considering some of the tensions identified as inevitable when design decisions are made.

A major tension is that between functionality and complexity; any gain in functionality may well be offset by the resulting increase in complexity. This can be seen in decisions regarding the number and type of atomic and molecular tools to provide. It also appears in user choice versus default behaviour and as a tension between information and clutter in labels and figure descriptions. Questions here concern the significance of the tension for student learning and the effects of possible strategies for alleviating the tension. For example, grouping tools and using multifunctional tools may alleviate complexity without affecting functionality, or may simply create different types of complexity.

Another tension identified is that between static and dynamic geometry. No model of static geometry is sufficient to completely determine dynamic geometry and hence decisions need to be made concerning the behaviour of objects when dragged, ranging from the movement of points on objects to the effect of dragging dependent objects. A major question is the effect that such decisions have on the geometry that students construct when interacting with an IGS.

A related tension is that between mathematics which is: universally acceptable, local to a particular culture, or unacceptable. The origin of the five programs in four

different countries with different education systems is likely one of the sources of the diversity in design decisions, some of which may seem problematic. A question is the extent to which conventions from a different culture impact on student learning. A further question concerns the criteria for distinguishing acceptable from unacceptable mathematics.

Two further tensions that have been identified are those between the primacy of geometric or algebraic representation and between easy affordance and pedagogical value. The implications of each of these types of decision could usefully be further considered.

Selection order illustrates an important concern for research: a design decision may have both an immediate impact and an impact due to the further functionality that is enabled or disabled by the decision. It is important to explore both the impact of selection order in itself and also the effect of the functionalities such as points on the fly and graphic previews that are enabled by A/O selection order.

Hopefully, such research, together with proposed generic criteria for assessing the quality of interactions such as those developed by Sedig and Liang (2006), will enable the creation of criteria by which current design decisions may be evaluated and careful future design decisions may be made.

In conclusion, I will again quote Jackiw:

To the degree our work in mathematics technology aspires to educational influence at significant scale, rather than just to the pleasure of small, pre-qualified technological elites, we have first to admit that design matters—that specific design matters, specifically—and, second, to develop a much richer discourse for design analysis. (Butler, Jackiw, Laborde, Lagrange & Yerushalmy, 2009, p. 432)

It is my hope that this paper is a contribution to the development of this discourse.

Acknowledgments I would like to acknowledge comments made by Markus Hohenwarter and to thank Jean-Marie Laborde, Colette Laborde, Nicholas Jackiw, Scott Steketee, Michael Borchers, and Ulrich Kortenkamp for participating in some very interesting and informative dialogues about Cabri, Geometer's Sketchpad, GeoGebra, and Cinderella. I am also grateful to the reviewers for many helpful criticisms and suggestions. I would particularly like to thank Jürgen Richter-Gebert for his extremely insightful and helpful reflections both on Cinderella and on the design of IGS as a whole.

Appendix A: Further interactions possible in a visual mathematical representation

Interaction	Description	In IGS
Animating	Generating movement within a VMR	Animating
Annotating	Augmenting a VMR by placing notes or marks on it	Labelling Creating text Tracing Changing appearance Scribbling
Chunking	Grouping a number of similar or related, but disjointed, visual elements	Organizing tools
Cutting	Removing unwanted or unnecessary portions of a VMR	Deleting
Filtering	Showing, hiding, or transforming a selected subset of the visual elements of a VMR according to certain characteristics or criteria	Hiding/showing Changing appearance Suppressing tools
Fragmenting	Breaking a VMR into its component or elemental parts	
Probing	Focusing on or drilling into some aspect, property, or component of a VMR for further analysis and information, includes panning and zooming	Object information Measuring Changing scale
Rearranging	Changing the spatial position and/or direction of elements within a VMR	Dragging Rearranging tools
Scoping	Changing the degree to which a VMR is visually constructed/deconstructed by adjusting its field of view	Replaying construction Do/undo
Searching	Seeking out the existence of or position of specific features, elements, or structures within a VMR	Searching for tools, etc.

Sedig & Sumner (2006)

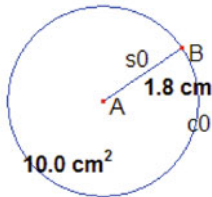
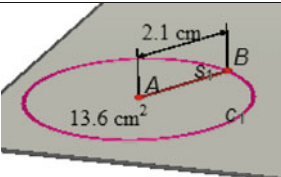
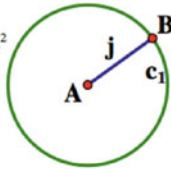
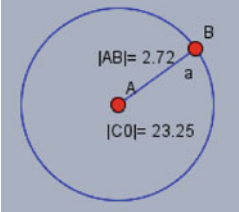
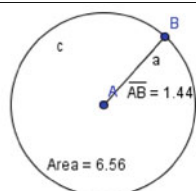
Appendix B: Different types of circles that can be constructed in an IGS:

1. circle defined by its centre, with radius defined by a mouse click at a particular distance from its centre (atomic);
2. circle defined by its centre and a point on its circumference (radius point) (atomic);
3. circle defined by its centre and a number for its radius (also known as a compass tool) (atomic);
4. circle defined by an axis and a radius point (molecular);
5. circle defined by its centre and a segment, the length of which gives its radius (also known as a compass tool) (molecular);
6. circle defined by three points (molecular).

Program	1	2	3	4	5	6
Cabri II Plus	Y	Y	Y		Y	
Cabri 3D		Y	Y	Y	Y	Y
Cinderella	Y	Y	Y		Y	Y
GeoGebra		Y	Y		Y	Y
GSP		Y	Y		Y	

Note that with Cabri 3D, options 2, 3, and 5 also require a plane to be specified, and circles may also be created as the intersection between two objects such as a plane and a sphere.

Appendix C: Figures and figure descriptions

Cabri II Plus 	B A c0 s0 1.8 cm 10.0 cm² Point Point Circle: A, B Segment: A, B Text (Distance or Length): A, B Text (Area): c0																																								
Cabri 3D 	- point A, on base plane - point B, on base plane - circle c₁, in base plane, centered on point A and through point B - segment s₁, by point A and point B - value X₁, distance between point A and point B - value X₂, area of circle c₁																																								
Geometer's Sketchpad AB = 2.65 cm Area c₁ = 22.12 cm² 	Given: - Point A - Point B Steps: - Let c₁ = circle centered at A passing through B. - Let j = segment between A and B. - Let AB = distance from Point A to Point B. - Let Area c₁ = area of Circle c₁.																																								
Cinderella 	<table><thead><tr><th></th><th>Who?</th><th>What?</th><th>Where?</th></tr></thead><tbody><tr><td>●</td><td>A</td><td>Point(2.04 2.6)</td><td>(2.04 2.6)</td></tr><tr><td>●</td><td>B</td><td>Point(4.24 4.2)</td><td>(4.24 4.2)</td></tr><tr><td>○</td><td>C0</td><td>Circle(A;B)</td><td>$(x - 2.04)^2 + (y - 2.6)^2 = 2.72^2$</td></tr><tr><td>—</td><td>a</td><td>Segment(A;B)</td><td>—</td></tr><tr><td></td><td>A0</td><td>Area(C0)</td><td>23.25</td></tr><tr><td></td><td>Text0</td><td>Text(A;A0)</td><td> C0 = 23.25</td></tr><tr><td></td><td>C</td><td>Mid(A;B)</td><td>(3.14 3.4)</td></tr><tr><td></td><td>dist0</td><td>Distance(A;B)</td><td>2.72</td></tr><tr><td></td><td>Text1</td><td>Text(dist0)</td><td> AB = 2.72</td></tr></tbody></table>		Who?	What?	Where?	●	A	Point(2.04 2.6)	(2.04 2.6)	●	B	Point(4.24 4.2)	(4.24 4.2)	○	C0	Circle(A;B)	$(x - 2.04)^2 + (y - 2.6)^2 = 2.72^2$	—	a	Segment(A;B)	—		A0	Area(C0)	23.25		Text0	Text(A;A0)	C0 = 23.25		C	Mid(A;B)	(3.14 3.4)		dist0	Distance(A;B)	2.72		Text1	Text(dist0)	AB = 2.72
	Who?	What?	Where?																																						
●	A	Point(2.04 2.6)	(2.04 2.6)																																						
●	B	Point(4.24 4.2)	(4.24 4.2)																																						
○	C0	Circle(A;B)	$(x - 2.04)^2 + (y - 2.6)^2 = 2.72^2$																																						
—	a	Segment(A;B)	—																																						
	A0	Area(C0)	23.25																																						
	Text0	Text(A;A0)	C0 = 23.25																																						
	C	Mid(A;B)	(3.14 3.4)																																						
	dist0	Distance(A;B)	2.72																																						
	Text1	Text(dist0)	AB = 2.72																																						
GeoGebra 	Free Objects A = (2.22, 2.02) B = (3.18, 3.1) Dependent Objects a = 1.44 areac = 6.56 c: $(x - 2.22)^2 + (y - 2.02)^2 = 2.09$ distanceAB = 1.44																																								
<table><thead><tr><th>No.</th><th>Name</th><th>Definition</th><th>Value</th></tr></thead><tbody><tr><td>1</td><td>Point A</td><td></td><td>A = (2.22, 2.02)</td></tr><tr><td>2</td><td>Point B</td><td></td><td>B = (3.18, 3.1)</td></tr><tr><td>3</td><td>Circle c</td><td>Circle through B with center A</td><td>c: $(x - 2.22)^2 + (y - 2.02)^2 = 2.09$</td></tr><tr><td>4</td><td>Segment a</td><td>Segment [A, B]</td><td>a = 1.44</td></tr><tr><td>5</td><td>Number areac</td><td>Area[c]</td><td>areac = 6.56</td></tr><tr><td>6</td><td>Text Textc</td><td>"Area = " + areac</td><td>Textc = "Area = 6.56"</td></tr><tr><td>7</td><td>Number distanceAB</td><td>Distance of A and B</td><td>distanceAB = 1.44</td></tr><tr><td>8</td><td>Text TextAB</td><td>$\overline{AB}$ + (Name[A]) + (Name[B]) + " = " + distanceAB</td><td>TextAB = "$\overline{AB}$" = 1.44"</td></tr></tbody></table>				No.	Name	Definition	Value	1	Point A		A = (2.22, 2.02)	2	Point B		B = (3.18, 3.1)	3	Circle c	Circle through B with center A	c: $(x - 2.22)^2 + (y - 2.02)^2 = 2.09$	4	Segment a	Segment [A, B]	a = 1.44	5	Number areac	Area[c]	areac = 6.56	6	Text Textc	"Area = " + areac	Textc = "Area = 6.56"	7	Number distanceAB	Distance of A and B	distanceAB = 1.44	8	Text TextAB	$\overline{AB}$ + (Name[A]) + (Name[B]) + " = " + distanceAB	TextAB = " $\overline{AB}$ " = 1.44"		
No.	Name	Definition	Value																																						
1	Point A		A = (2.22, 2.02)																																						
2	Point B		B = (3.18, 3.1)																																						
3	Circle c	Circle through B with center A	c: $(x - 2.22)^2 + (y - 2.02)^2 = 2.09$																																						
4	Segment a	Segment [A, B]	a = 1.44																																						
5	Number areac	Area[c]	areac = 6.56																																						
6	Text Textc	"Area = " + areac	Textc = "Area = 6.56"																																						
7	Number distanceAB	Distance of A and B	distanceAB = 1.44																																						
8	Text TextAB	$\overline{AB}$ + (Name[A]) + (Name[B]) + " = " + distanceAB	TextAB = " $\overline{AB}$ " = 1.44"																																						

References

- Bainville, E., & Laborde, J.-M. (2004, 2007). *Cabri 3D [computer software]*. Grenoble: Cabrillog.
- Butler, D., Jackiw, N., Laborde, J., Lagrange, J., & Yerushalmy, M. (2009). Design for transformative practices. In C. Hoyles & J.-B. Lagrange (Eds.), *Mathematics education and technology—rethinking the terrain. The 17th ICMI study* (pp. 425–438). New York: Springer.
- Falcade, R., Laborde, C., & Mariotti, M. (2007). Approaching functions: Cabri tools as instruments of semiotic mediation. *Educational Studies in Mathematics*, 66(3), 317–333.
- Goldenberg, E., Scher, D., & Feurzeig, N. (2008). What lies behind dynamic interactive geometry software? In G. Blume & M. Heid (Eds.), *Research on technology in the learning and teaching of mathematics: Cases and perspectives* (Vol. 2, pp. 53–87). Greenwich, CT: Information Age.
- Gray, S., Loud, B., & Sokolowski, C. (2005). Undergraduates' errors in using and interpreting variables: A comparative study. In G. Lloyd, M. Wilson, J. Wilkins, & S. Behm (Eds.), *Proceedings of the 27th annual meeting of the North American Chapter of the International Group for the Psychology of Mathematics Education*. Eugene, OR: All Academic.
- Hohenwarter, M., et al. (2002, 2010). *GeoGebra [computer software]*. Linz: GeoGebra Inc.
- Hoyles, C., & J.-B. Lagrange, J.-B. (Eds.). (2010). *Mathematics education and technology—rethinking the terrain. The 17th ICMI study*. New York: Springer.
- Jackiw, N. (2009). *The Geometer's Sketchpad (version 5) [computer software]*. Berkeley, CA: Key Curriculum Press.
- Jones, K. (2000). Providing a foundation for deductive reasoning: Students' interpretations when using dynamic geometry software and their evolving mathematical explanations. *Educational Studies in Mathematics*, 44, 55–85.
- Jones, K., Mackrell, K., & Stevenson, I. (2009). Designing digital technologies and learning activities for different geometries. In C. Hoyles & J.-B. Lagrange (Eds.), *Mathematics education and technology—rethinking the terrain. The 17th ICMI study* (pp. 47–60). New York: Springer.
- Kortenkamp, U. (1999). Foundations of dynamic geometry. Dissertation, ETH Zürich.
- Kortenkamp, U., & Dohrmann, C. (2010). User interface design for dynamic geometry software. *Acta Didactica Napocensia*, 3(2), 59–66.
- Küchemann, D. (1981). Algebra. In K. Hart (Ed.), *Children's understanding of mathematics: 11–16* (pp. 102–119). London: John Murray.
- Laborde, C. (2007). The role and uses of technologies in mathematics classrooms: Between challenge and *modus vivendi*. *Canadian Journal of Science, Mathematics and Technology Education*, 7(1), 68–92.
- Laborde, C., & Laborde, J.-M. (2008). The development of a dynamical geometry environment: Cabri-géomètre. In G. Blume & M. Heid (Eds.), *Research on technology in the learning and teaching of mathematics: Cases and perspectives* (Vol. 2, pp. 31–52). Greenwich, CT: Information Age.
- Laborde, J.-M., et al. (2003). *Cabri II Plus [computer software]*. Grenoble: Cabrillog.
- Mackrell, K. (2008). Cabri 3D: An environment for creative mathematical design. In P. Liljedahl (Ed.), *Canadian Mathematics Education Study Group proceedings 2007 annual meeting*. Frederickton: University of Frederickton.
- Richter-Gebert, J., & Kortenkamp, U. (2007, 2010). *Cinderella 2 [computer software]*. Heidelberg: Springer.
- Richter-Gebert, J., & Kortenkamp, U. (2010). The power of scripting: DGS meets programming. *Acta Didactica Napocensia*, 3(2), 59–66.
- Schimpf, F., & Spannagel, C. (2010). *Reducing graphical user interfaces in DGS*. Paper presented at the I2Geo conference 2010, Hluboka nad Vltavou.
- Sedig, K., & Liang, H.-N. (2006). Interactivity of visual mathematical representations: Factors affecting learning and cognitive processes. *Journal of Interactive Learning Research*, 17(2), 179–212.
- Sedig, K., & Sumner, M. (2006). Characterizing interaction with visual mathematical representations. *International Journal of Computers for Mathematical Learning*, 11, 1–55.
- Sinclair, M. (2003). Some implications for the results of a case study for the design of pre-constructed dynamic geometry sketches and accompanying materials. *Educational Studies in Mathematics*, 57(1), 91–119.
- Talmon, V., & Yerushalmy, M. (2004). Understanding dynamic behaviour: Parent–child relations in dynamic geometry environments. *Educational Studies in Mathematics*, 57, 91–119.