Journal for Geometry and Graphics
Volume 10 (2006), No. 1, 115–124.

# RISKO, an Educational Geometry Drawing Software With an Innovative Interface

**Eduardo Toledo Santos, Leandro Lourenzoni, André Luís Lima de Oliveira**

*Escola Politécnica, University of São Paulo*
*Av. Prof. Almeida Prado, trav. 2, n. 83, 05508-900 São Paulo, Brazil*
*email: eduardo.toledo@poli.usp.br*

**Abstract.** The traditional interfaces used in most graphical software have many shortcomings when applied to drawing programs in the context of geometry instruction. Among them are: High complexity, availability of shortcuts to solve problems that the student is supposed to construct by geometry; and none resemblance to real drawing instruments, preventing the student from learning how to properly handle them. A new interface for an educational geometry drawing software called RISKO was designed to mitigate these problems and is presented in this paper. Following a pure Direct Manipulation paradigm and using a concrete, real-world metaphor style, this new interface features drawing tools (drafting triangles, compass, pencil), which behave almost like their real counterparts. Due to its characteristics, the proposed interface is intended to be very intuitive so that it does not require any operating instructions to the user and has no error messages. Also, it is able to support all compass-and-ruler geometric constructions, while training the apprentice in the proper methods for handling real drawing instruments.

*Key Words:* Graphical User Interfaces, Real World Metaphor, Educational Software, Geometry, Drawing Software

*Categories and subject descriptors:* H.5.2 [Information Interfaces and Presentation]: User Interfaces — Graphical user interfaces (GUI)

*MSC 2000:* 51M04

## 1. Introduction

There is a growing interest in using e-learning for geometry and/or technical drawing teaching, both in distance education and as a supplement to face-to-face courses. Although there are many drawing programs being used for educational purposes, some shortcomings may be

ISSN 1433-8157/$ 2.50 

identified in their user interfaces when used as a (partial) replacement for paper and pencil practice in geometry instruction.

Most of those programs adopt a conventional Graphical User Interface (GUI), based on windows, icons, menus and pointing (WIMP model). Due to this, like any other software, they demand some time from users to be mastered. Likewise, most of them have too many features, allowing a full range of graphical constructions (tangent, parallel and perpendicular lines, middle point, bisectors, inscribed and circumscribed circles etc.) to be executed with a few mouse clicks. Finally, their interfaces are very abstract, far apart from the drawing instruments the students use in classroom.

A non-intuitive GUI is a significant problem in that context because, among other reasons, as it is widely known, the credits assigned to graphics courses, with a few exceptions, have been reduced worldwide. Hence, it is very undesirable to have a student spending time to learn how to use a new program instead of practicing the target topics of a geometry course in the short time available. Furthermore, as this kind of software usually covers only a small part of the contents of a course, its use is occasional and, therefore, it is not worth investing too much time for learning it.

Different from a CAD (Computer Aided Design) system, where drawing efficiency is among the most important requirements, an educational tool must sacrifice execution speed in favor of pedagogic issues, if needed. Therefore, a plethora of features for allowing easy execution of any foreseeable construction is not only undesirable because it precludes the student from performing a graphical construction using geometric reasoning, but also because it makes the software learning curve steeper as there are more commands to memorize and locate on the interface.

Concerning the abstraction aspect of the WIMP GUI, it is not known to which extent the use of an artificial computer interface for drawing can hamper the abilities of an apprentice regarding the correct manipulation of the drawing instruments (compass, triangles[1], ruler and pencil). The authors plan to investigate this issue in the future.

This paper describes an interface based on a real world metaphor featuring drawing tools. This interface is under development by the authors and will serve a geometry educational software, accessible through the WWW. The Java programming language is being used for the implementation of both a standalone and an applet version of it.

## 2. The new interface

To fulfill the needs stated before, a direct manipulation, concrete and real-world metaphor interface was proposed as an alternative GUI to an educational geometry drawing software called RISKO (which stands for Realistic Interface for Simulating a Kit of Objects and sounds like the Portuguese word for *trace*). This interface has no menus or buttons. It resembles a sheet of paper on a drawing table with some drawing instruments over it: a triangle set (45-90° and 30-60-90° degrees), a compass and a pencil with an eraser tip. There is also a magnifying glass for zooming. Fig. 1 illustrates its present (v1.0) appearance.

The instruments are supposed to be used like their real counterparts: the pencil is used to draw points and lines or to scribble and to write (using the keyboard); triangles are used to support the pencil when drawing straight lines. Together, triangles may be used to draw lines parallel or perpendicular to other lines or at some angles. The pencil's eraser end is used to delete points, lines or arcs with a single click. A portion of a line or of an arc may

[1] All mentions to the word *triangle* in this paper refer to the drafting tool, not the shape.

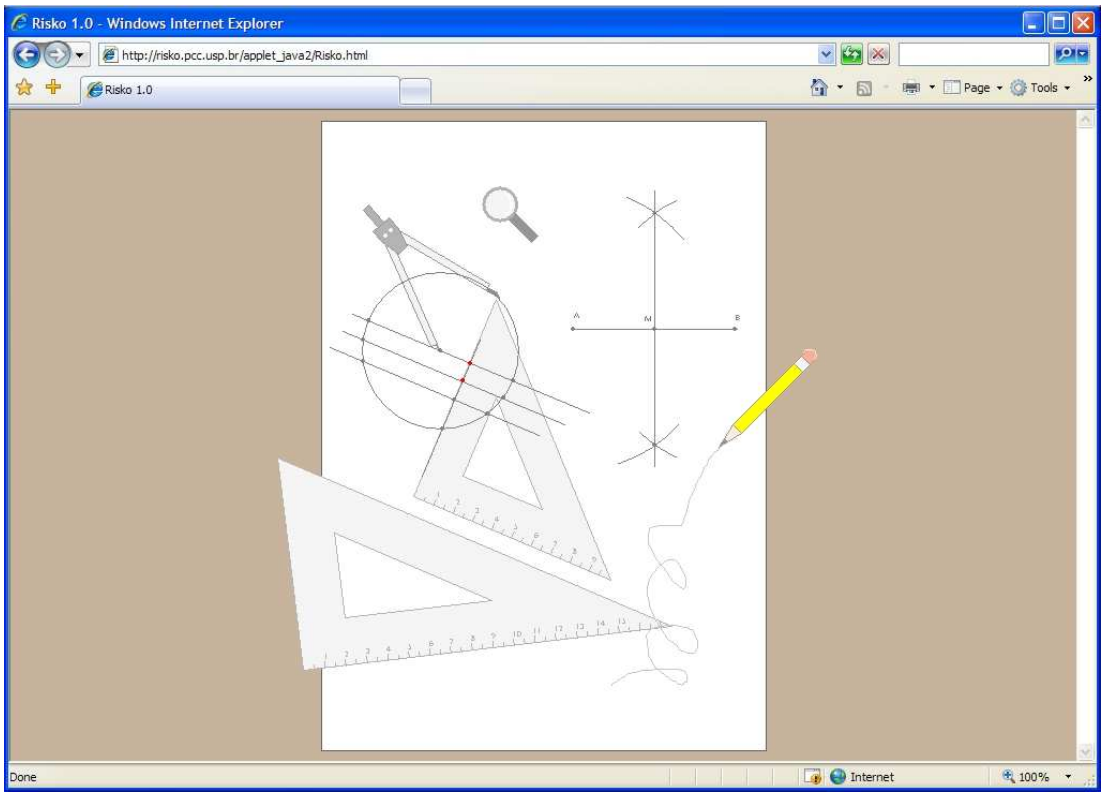


Figure 1: RISKO's drawing table metaphor interface

be erased by rubbing the eraser over it. The compass is used to draw circles or arcs and also to carry distances. The white area represents a A4-size paper sheet. Its scale matches the millimeter marks on the triangles, allowing drawings to be drawn (and printed) to scale. The magnifying glass is the most "unreal" element in the interface, as it is used to support the zoom action common in graphics software: the drawing will dynamically zoom in and out around the lens center, exposing a larger or smaller area of the desktop. In this operation, the instruments keep their original size and position. Note that the millimeter marks on the edges of the triangles, as well as the compass' radius, are kept in scale with the rest of the drawing when zooming (see Fig. 2).

A question for every designer of a geometry-drawing program is if the available functionality in the software is enough to allow the user to perform *all* the geometric constructions possible with a compass and a ruler. In the present case, this is a non-issue as the drawing objects implemented by the interface cover all their drawing-related perceived affordances[2].

The proposed interface incorporates the *local tool* concept [1]: unlike tool palettes, where there can be only one tool active at a time, local tools can be picked up, used and put down anywhere in the work surface, retaining their attributes (pencil color, compass radius, etc.). Evidence [1] was found that this interaction style is easily learned with minimal instruction, even by very young children.

In addition, the *composite tool* concept [3] also can be identified in the proposed interface by the manner the drawing instruments interact with each other to increase their usefulness: one triangle can slide on the edge of the other to draw parallel or perpendicular lines (interacting with a third object – the pencil), etc.

[2] *Affordance* is a property of an object that indicates how to interface with that object [4].

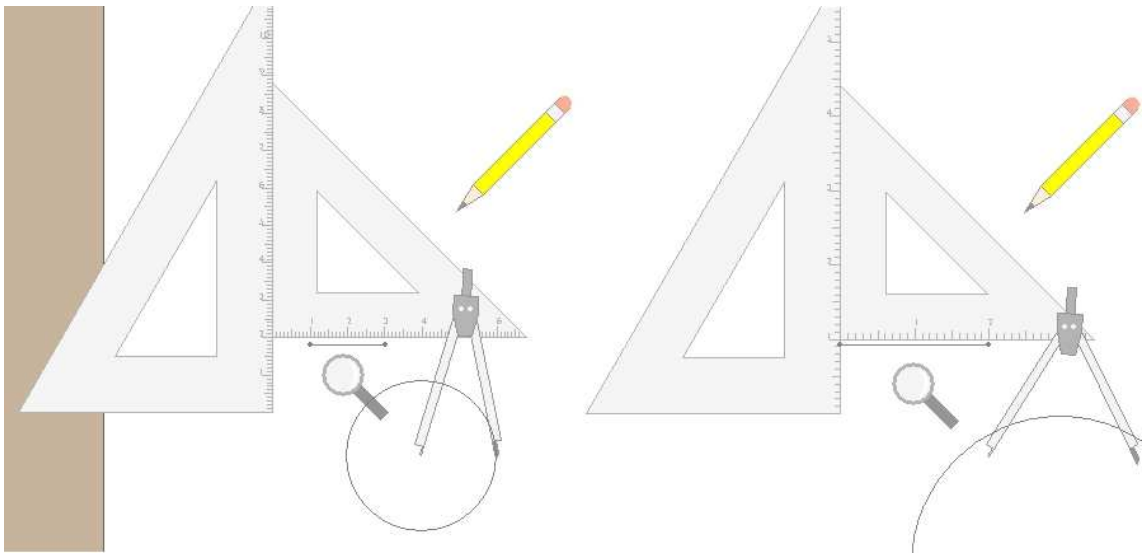

Figure 2: Zooming operation and its effects on drawing, millimeter marks and compass radius

### 2.1. Direct Manipulation

As mentioned before, the proposed interface adopts a Direct Manipulation paradigm. Direct Manipulation Interfaces have three defining principles [10], which most GUIs implement:

- Continuous representation of the objects and actions of interest;
- Physical actions or presses of labeled buttons instead of complex syntax;
- Rapid incremental reversible operations whose effect on the object of interest is immediately visible.

This interface model was chosen because all its advantages [10] are desirable in this project:

- Novices can learn basic functionality quickly;
- Intermittent users can retain operational concepts;
- Error messages are rarely needed[3];
- Users can immediately see if their actions are furthering their goals;
- Users experience less anxiety because the system is comprehensible and because actions can be reversed so easily;
- Users gain confidence and mastery because they are the initiators of action, they feel in control, and the system responses are predictable.

### 2.2. Real-World Metaphors

Metaphors are associations between the user's world and concepts of the computer universe. They are extensively used in modern interfaces due to their intuitiveness (e.g., Windows and Macintosh's folders, files and trash cans, representing office objects). Metaphors can be classified according to its attributes [11]:

- Real-world vs. non-real-world metaphors;
- Concrete vs. conceptual metaphors;
- Spatial vs. time-based metaphors;

[3]Indeed, in RISKO there are no error messages at all!

- General vs. application-dependent metaphors;
- Flexible and composite metaphors vs. rigid metaphors.

The metaphor implemented in RISKO's interface is based on the real-word and is concrete.

Real world metaphors are those based in things or concepts pertaining to common reality (e.g., dentist) as opposed to non-real-world ones (e.g., gnome). Concrete metaphors are those based on things (e.g., pencil) while conceptual metaphors are abstract (e.g., peace).

Concrete real-world metaphors may be the most appropriate for naive and casual users [11], which are characteristics of our target audience (apprentices, first time users).

If an interface has only representations of real-world objects users will naturally know what to do with them [6] as they are acquainted with those objects and already know their perceived affordances [7].

The virtual instruments in the new interface closely emulate their real counterparts (both visually and behaviorally) and their manipulation is intended to be intuitive enough to dismiss any explanation or instruction.

## 3. Dealing with input limitations

Although bimanual interaction has some advantages [2], they require special input devices, which are not generally available to all users, restricting the potential audience for our tool. This limitation is unacceptable for our goals. Therefore, the interface design had to cope with a one-handed input. Due to this fact, a major issue is that the interaction in the proposed interface is through a single mouse, while the user utilizes both hands when interacting with drawing instruments in the real world. Sliding a 45° triangle on the edge of a 30-60° triangle for obtaining parallel lines or holding a triangle in place and, at the same time, drawing a line at its edge with the pencil are operations that demand two hands. Likewise, some operations on only one instrument may require both hands (e.g., adjusting compass radius).

Other issues relate to the required precision for positioning instruments aligned with points or lines and drawing points at exact places.

Some clever techniques were employed to circumvent these problems and, at the same time, keeping the interaction intuitive, as it is shown in the following sections.

### 3.1. Heavy objects for one-handed input

A *heavy instrument* metaphor was adopted for allowing the user to position one drawing instrument in place and them using another tool leaning on the first one without displacing it (e.g., a triangle sliding against the edge of another triangle or a pencil drawing a line against a triangle edge). An object cannot be pushed by another and will move only if directly dragged by the user.

Objects can be moved by the familiar *click'n'drag* action (click and hold down the mouse button -- move the pointer, dragging the instrument along -- release the mouse button). A triangle, when moved, shows realistic behavior, as if the user were dragging it with a finger tip at the cursor position. Collision detection is performed so that triangles do not overlap or the pencil lead does not pass through triangle's edges when tracing. Compass, pencil and magnifying glass can all be moved in the same way. The paper sheet can also be panned through the software window using the same action, with the tools over it following the move.

### 3.2. Instrument's hot zones

The specified interface is supposed to be portable to most platforms, requiring only a one-button mouse, which also contributes to easier operation. On the other hand, this requirement made it more difficult to the software designers to fit all the necessary controls in the interface. To achieve the desired goal, a *hot zone* concept was implemented for operating the drawing instruments. Their behavior depends on which part of them, or zone, the user clicked on when selecting it (Fig. 3). The zone areas are highlighted when the mouse cursor is over them.

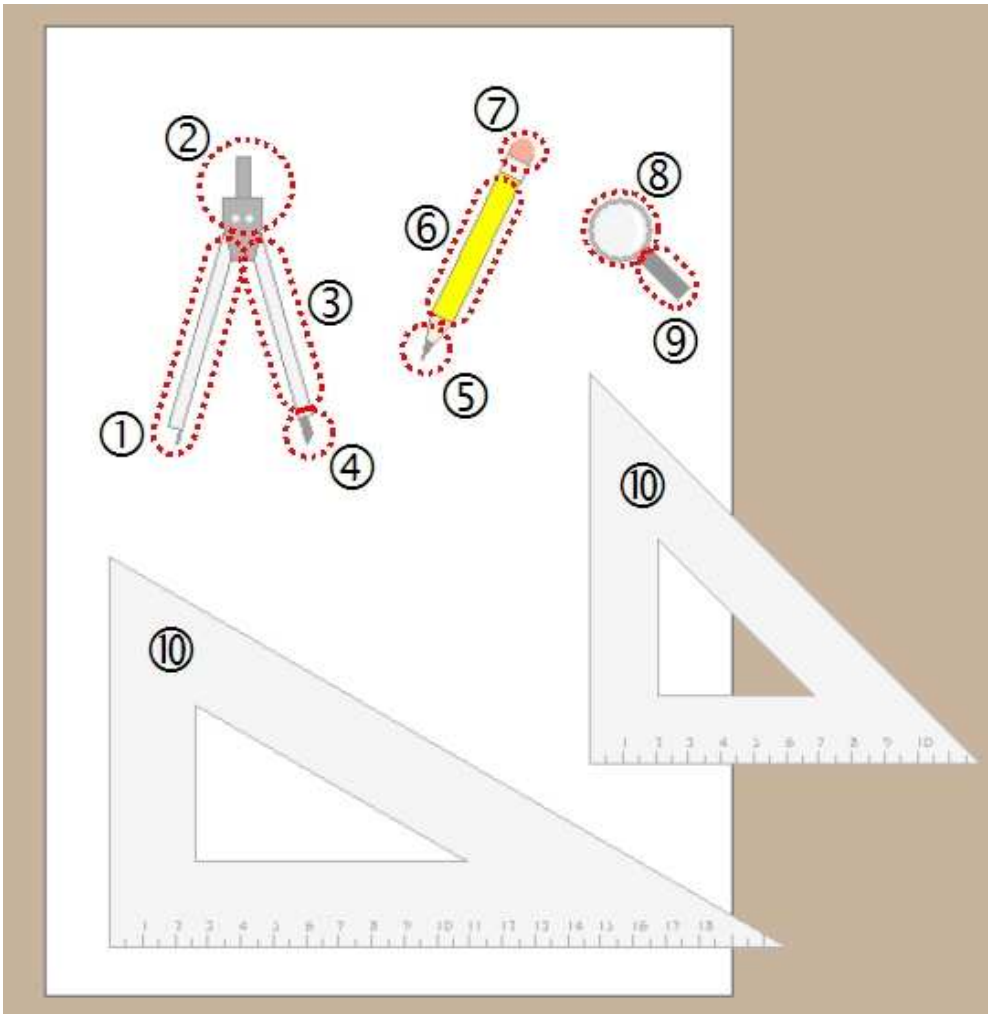


Figure 3: *Hot areas* of the drawing instruments

The pencil/eraser has only three functionalities: to be moved around the paper sheet, to draw lines when it is dragged (if active) and to erase points, lines and arcs, if dragged by the eraser tip. Three *hot zones*[4] were mapped to this object to implement these operations, as shown in Table 1.

The compass has four affordances, mapped as shown in Table 2.

The magnifying glass has two hot zones: the handle (hot zone number 9 in Fig. 3) is used for moving it around the paper sheet. Clicking inside the lens (hot zone number 9)

[4]Actually, there is a fourth *hot zone* mapped to the white ring between the eraser and the pencil body, which allows adjusting the pencil angle. This affordance is not obvious but is not essential either, adding to the tool usability in some cases.

Table 1: Pencil/eraser affordances

| *# in Fig. 3* | *Hot zone* | *Action* |
|---|---|---|
| 5 | Lead | Draw point or line or writes at the lead position |
| 6 | Body | Move the pencil without drawing |
| 7 | Eraser end | Delete what is under the eraser area |

Table 2: Compass affordances

| *# in Fig. 3* | *Hot zone* | *Action* |
|---|---|---|
| 1 | Spike Arm | Move compass without drawing or changing its radius |
| 2 | Handle/Hinge | Pivot compass around spike position keeping radius |
| 3 | Lead Arm | Adjust Radius, keeping spike position |
| 4 | Lead | Draws arc |

and moving the pointer towards the upper border of the paper commands a zoom in action; moving downward activates a zoom out, both centered in the middle point of the lens.

The 45°- and 30-60°-triangles have the same behavior and feature only one hot zone, corresponding to their entire bodies (hot-zone number 10), which is used for moving them around.

### 3.3. Highlighting and snapping for precise positioning

For precise positioning of the pencil lead or the compass lead and spike, as well as the triangles' edges on drawn points, a highlight and snapping feature was implemented. While these user-controlled parts (tips and edges) are over (or very near) any drawn point, it is highlighted, changing its color to bright green. This behavior signals a potential snapping action; if the user releases the mouse button while the point is green, the instrument tip will be positioned exactly on that point, which changes its color to red indicating the snap activation. Likewise, a line will highlight when the pencil tip is over it, allowing the creation of a point right over it.

If a triangle's edge gets snapped to a point, it will pivot around it. If the edge is snapped to two points, its movement will be restricted to slide aligned with the two points. To release from the snap, the user must simply drag the instrument away from the snapping point(s). Only two points, at most, will be highlighted at the same time.

Points are automatically created at all intersections (between lines, arcs, circles and themselves) allowing easy alignment with them.

The ruler marks (on one of each triangle's edges) are active in this same sense and, therefore, pencil and compass tips can be precisely snapped to them.

The precise alignment of a triangle edge with a drawn line is automatic: the user has just to align them close enough and the software will snap them together. Putting a triangle edge on the horizontal position (i.e. aligned with the bottom border of the paper sheet) works in the same way: just level it well enough and RISKO will adjust its position to perfectly

horizontal.

Although these features are not present in real objects, they are simple enough for the user to quickly realize this mechanism after playing with the interface for a few moments. They are necessary for compensating the lack of precise control of the movement of the virtual tools.

## 4. Proposed interface benefits

The introduction in this paper mentioned the following deficiencies in conventional graphic software GUIs that would be overcome with an interface as proposed in this research:

- *Lack of intuitiveness:* the feature-rich interfaces of today's CAD and drawing software requires heavy training or a very efficient online help to be operated effectively by first time users, leading to geometry learner's waste of time and facing of additional difficulties. It commonly takes several weeks of training for operating a CAD system and from 6 to 12 months to become productive with it [9];
- *Excess of features:* suppose the instructor wants the student to practice how to construct a circle tangent to a line and to another circle, using ruler and compass. Most CAD software is able to draw a solution to this problem with the user just selecting the circle tool and clicking on the given line and circle. This is not, of course, the way the instructor intended this geometry exercise to be solved by the apprentice;
- *Lack of practice with drawing instruments:* conventional interfaces, with menus and buttons, are so far apart from real drawing instruments, like compass and ruler, that, by using these interfaces only, the students may never learn how to draw parallel or perpendicular lines by correctly manipulating drawing instruments (a necessary ability, as a computer is not always at hand).

Besides reduction of those shortcomings, in-class geometry teaching practice has some peculiarities, which would be better dealt with such an educational tool:

- Solving geometry exercises in a blackboard requires great ability and precision, which are difficult to achieve when using large, clumsy wooden triangles and chalk compass. Teaching in computer-equipped classrooms demands replacing blackboard and chalk with whiteboard and marker, making things worse due to the whiteboard's slippery surface. Using a video projector for demonstrating the geometry constructions on the proposed software solves these problems, adding to precision and clearness;
- Some geometry constructions span vertically in the sheet and frequently requires more space than it is available when drawn in the blackboard. A computer projection can be easily panned or zoomed to fit the space on the screen;
- Learning pace varies very much among students in the same class, in particular for subject matters like Descriptive Geometry which require spatial abilities, a skill known to differ a lot in the population [5]. That fact demands careful explanations from the instructor, being sometimes boring to some clever students but perhaps, at the same time, too fast for some of their less skilled classmates. A tool that allows each student to review outside class the resolutions of exercises according to its own pace is essential in mitigating this problem.

## 5. Future Developments

The development of RISKO is not finished yet. It is always a challenge to devise ways to implement new features in the software and keep following the pure direct manipulation style and the fidelity of the adopted metaphor. For example, today the drawings produced with the software can be saved and load pressing the F9 and F10 keys. This is just a temporary work-around as we do not want to disrupt the adopted interface paradigm by putting buttons on the screen. The proposed solution for saving and loading files is to make a logical mapping between sheets of paper on a desktop and the drawing files on the computer disk. If the user, using the magnifying glass, makes a zoom-out wide enough, he will see that eventually there are more paper sheets on the virtual desk, at least one of them blank (Fig. 4). If the user starts drawing in this blank sheet, a new one will appear and a new RISKO file (`.rsk` extension) will be created in RISKO's directory in the computer. Each drawing file in this directory corresponds to a different sheet of paper over the virtual desktop.

To allow drawing with different line widths and colors, a colored pencil set box will be another object to be incorporated in the interface (Fig. 4).

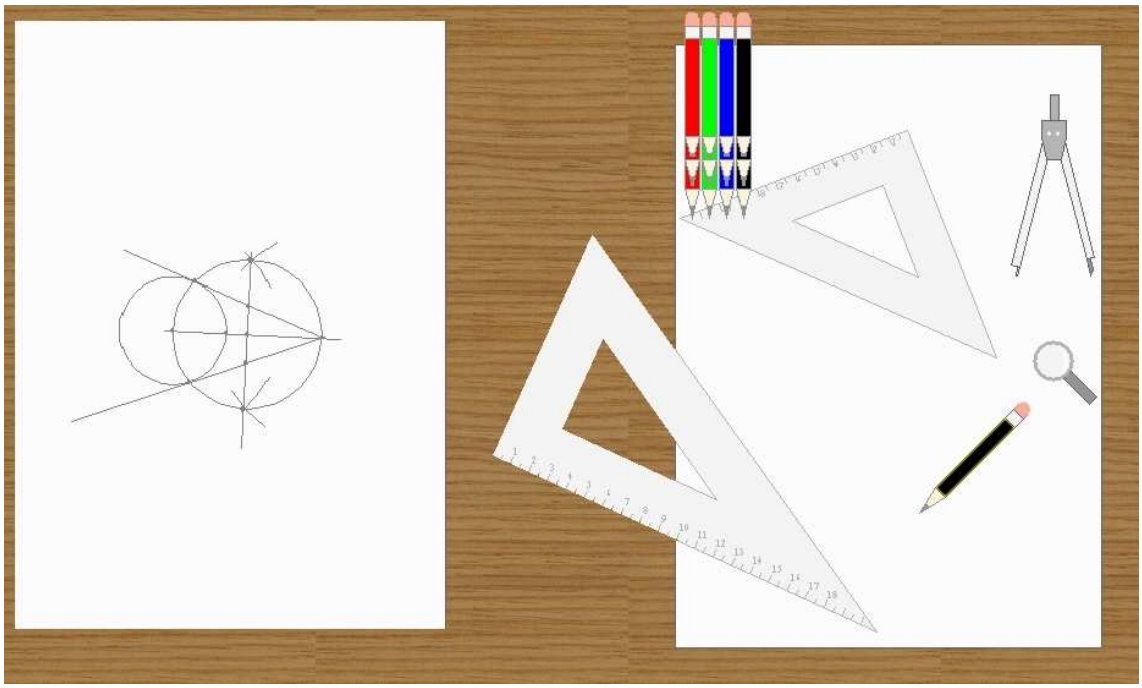

Figure 4: Each file on disk is a A4 paper sheet in RISKO's virtual desktop

## 6. Conclusion

A new interface for a geometry drawing software was presented. It is based on powerful interface related concepts like direct manipulation, metaphors, local and composite tools. The main goals of this interface are:

- To be immediately understandable, even for first time and occasional users, requiring no instructions at all;
- To allow all possible compass-and-ruler constructions to be executed without offering the powerful commands usually available in drawing software;

- To train apprentices in the use of drawing instruments in spite of using a virtual environment;

A preliminary version (v0.8) of this tool has been evaluated by 17 users [8]. Most of them (82.4%) reported that using the system was easy and 82.4% said they liked or liked very much the software. Currently our students have the option of using paper and pencil or RISKO in classroom; many students prefer using the software.

RISKO is available for free in the project website at `http://risko.pcc.usp.br`.

## Acknowledgments

The authors would like to thank FAPESP — Fundação para o Amparo da Pesquisa do Estado de São Paulo, Brazil for supporting this research through grants 03/04530-2 and 04/03707-9.

## References

[1] B.B. Bederson, J.D. Hollan, A. Druin, J. Stewart, D. Rogers, D. Proft: *Local tools: an alternative to tool palettes*. Proc. 9th Annual ACM Symposium on User Interface Software and Technology (UIST'96), 1996, pp. 169–170.

[2] C.G. Butle, R.S. Amant: *HabilisDraw DT: A Bimanual Tool-Based Direct Manipulation Drawing Environment*. CHI'04 Extended Abstracts on Human Factors in Computing Systems, 2004.

[3] J.M. Daughtry, R.S. Amant: *Power tools and composite tools: integrating automation with direct manipulation*. Proc. 2003 Internat. Conf. on Intelligent User Interfaces, Miami, Florida, 2003, pp. 233–235.

[4] D.A. Norman: *Affordance, conventions, and design*. Interactions **6**, n. 3, 38–43 (1999).

[5] R. Gòrska, C. Leopold, S. Sorby, K. Shiina: *International Comparisons of Gender Differences in Spatial Visualization and the Effect of Graphics Instruction on the Development of these Skills*. Proc. 8th ICECGDG, Austin, Texas, 1998, pp. 261–266.

[6] S. Michels: *Co-writing, look and feel*. Master's thesis. Tilburg University, The Netherlands, 1995.

[7] D.A. Norman: *The Design of Everyday Things*. MIT Press, 2000, 257 p.

[8] A.L.L. Oliveira: *Avaliação comparativa de diferentes modelos de interfaces gráficas empregadas no ensino de geometria, segundo os conceitos de usabilidade*. Master's thesis. Escola Politécnica, University of São Paulo, Brazil, 2005.

[9] L.A. Piegl: *Ten challenges in computer-aided design*. Computer-Aided Design **37**, n. 4, 461–470 (2005).

[10] B. Shneiderman: *Direct Manipulation for Comprehensible, Predictable, and Controllable User Interfaces*. Proc. Internat. Conf. on Intelligent User Interfaces – IUI'97, Orlando 1997, pp. 33–39.

[11] K. Vääinänen, J. Schmidt: *User Interfaces for Hypermedia: How to Find Good Metaphors ?* Adjunct Proc., Conf. on Human Factors in Computing Systems 1994 — CHI'94, pp. 263–264

Received August 1, 2004; final form May 2, 2006