From strokes to model : a bottom-up approach for sketching activities analysis

Stephane Safin, Roland Juchmes, Catherine Elsen & Pierre Leclercq

University of Liège, Belgium. Lucid-ULg : Lab for User Cognition and Innovative Design

Introduction

This paper proposes a framework for sketches analysis that could be useful for sketch-based interfaces development, in the particular context of preliminary activity-centred design. We will first describe our previous works on sketch-based support tools for the preliminary design phases and the context of these researches. The next chapter declares the concrete objectives of our approach, then the methodology of sketches analysis we propose is presented with a concrete example, and finally the last section is dedicated to the expected results and specifications of a sketch-based support tool that would follow our framework.

1. Context: Helping Architects in the First Stages of Design

The task of designing is becoming more and more complex: the multiplicity of constraints to deal with, the variety of actors and the economic context put the designers under high pressure, every loss of time leading to loss of money. That environment negatively influences the time accorded to the preliminary phases of design. Yet, this stage is not trivial : the better business decisions are often quickly drawn on a napkin corner !

Errors are a core part of conceptual design in architecture, but, after the preliminary phase, as long as the design progresses, they become more and more costly and hard to recover, That's this paradox we want to surpass, by proposing tools that could support designers in the early stages of design, in order to evaluate the efficiency of their design and to detect potential errors at the end of the preliminary design.

Our first works, mainly focused on architectural design, leaded to a software, named EsQUIsE [1,2,3], that aims at tracking, analyzing and interpreting architectural free-hand sketches and at proposing to designers real-time feedbacks.

The underlying principle is simple (fig. 1) : the architect draws his sketches and the software creates an operational model of the object. This model feeds evaluators (like the powerful evaluators available on later stages of design), that give feedback to the designer throughout alternative representations of its object. These alternative representations help designer to change his point of view on the design and to detect potential errors.

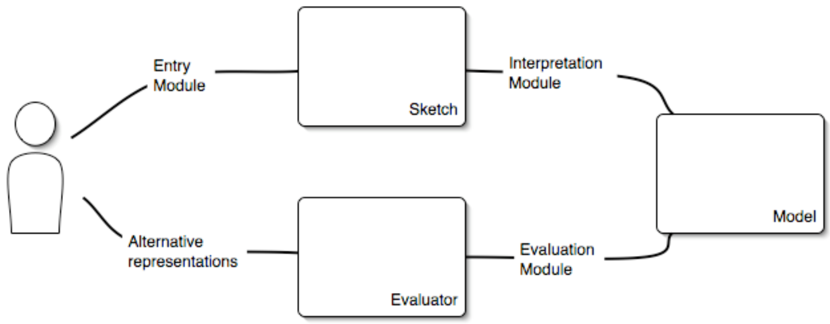

Fig1. EsQUIsE principle and operation

The software architecture is grounded on a multi-agent system, that could be very simply described by 3 modules. Firstly, the entry module : it consists in analyzing sketches in order to construct the geometrical model of the drawing, in other words, the internal representation of the structure of the drawing. Because of the sketches lack of precision, a "fuzzy graphic" model approach has been developed to take into account the considerable margin of error in the identification of the drawing geometry. Secondly, the interpretation module translates this geometrical information into a functional model of the planned architectural object (analysis of the semantic content of the drawing) : annotations and implicit information allow the system to compose in real time a complete model of the building designed. Thirdly, the evaluation module offers to the users non intrusive feedbacks about the artifact, like for instance (and at the present time) 3D visualizations, thermal consumptions, topological organization or energetic needs (in heating or air-conditioned).

Our motivations for the development of an analysis grid are (a) to test this logic on another domain of design and (b) to try to refine it in order to make our software evolve.

2. Goals of the analysis grid

In order to provide efficient help to designers, it is important that the system creates from the input data an accurate *model* of the object. This model is driven by the objectives of the evaluator : at each help, at each feedback required by the user corresponds a defined model, constructed on specific information gathered out from the data at our disposal, here the sketches and their content. Whatever the evaluator is (ergonomic, economic, productibility, ...), the model must contain a (partial) semantic understanding of the object. Thus the model is built on two kinds of information :

- observable traces of activities, e.g. the drawing;
- semantic knowledge on the domain or common sense, that have to be previously coded (or learned by the system thanks to advanced AI techniques).

Consequently to these properties, our analysis grid aims at the following objectives.

- i) to find relations between strokes and concepts, through sketch analysis. Are these relations univocal, equivocal, ... ?
- ii) starting from the graphic inputs, how could we build one specific model of the object ?
- iii) What implicit information do we have to add to the explicit content of the graphical entity, to get the object ?

3. Proposed Methodology

In order to create a model of the object from the sketching activity observation, we assume to get through different levels of analysis, from the strokes characterization to the complete object comprehension. The issue raised is to know what information and knowledge a computer system must know to make the link between all those levels. For this purpose, we propose a methodology in four steps, from a local (micro) to a global (macro) point of view on the object, and from a graphic to a semantic description of this object (fig. 1). Our purpose is to decompose the sketch analysis in simple steps. It is thus easier to identify knowledge necessary to pass from one step to another, than global knowledge necessary to fully understand a sketch. This assumption is directly linked to the design of a multi-agent system.

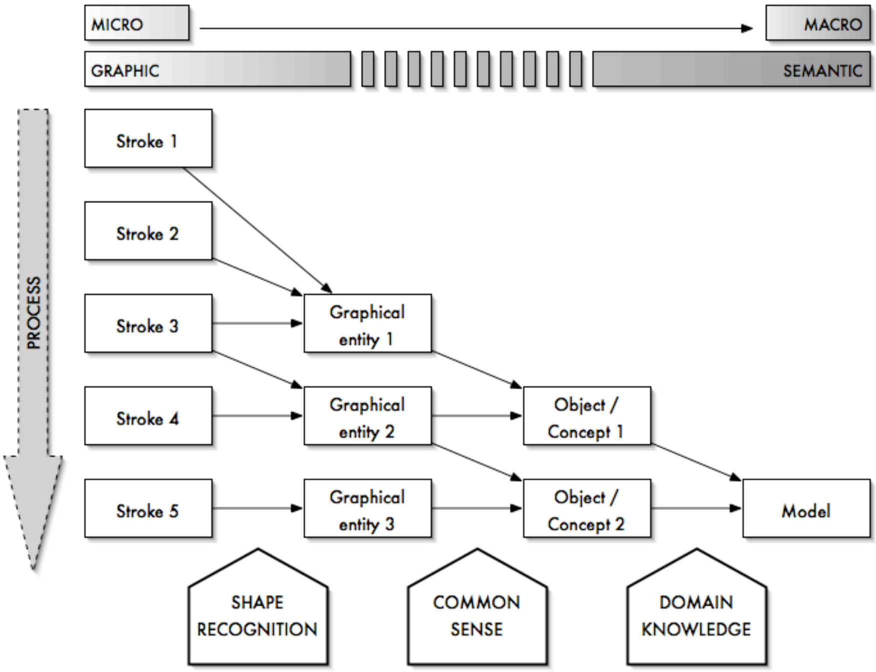

Fig2. Analysis grid

The first level of analysis is the *stroke*. A stroke has an unambiguous definition: it begins when the pen is placed on the sheet of paper and ends when it is taken off. The stroke is our unit of analysis. The drawing process is viewed as a succession of strokes, independently of time. In real time systems, we use properties of the strokes like speed, pressure level or tilt of the stylus to sort strokes and help the recognition phase. In this paper, we analyze videos, so only the relations between strokes are coded.

The second level of analysis is the *graphical entity*. A graphical entity is a group of stokes that can be visually identified with respect to their spatiotemporal relations (e.g. parallelism, perpendicularity, alignment…). A graphical entity can be a closed shape, a sign or a symbol, dashed lines, arrows etc. This definition presents some ambiguities : a great number of insignificant graphical entities exist in a drawing (e.g. contours), finding rules to detect important ones is not a trivial problem. From the strokes to the graphical entities, the system must have a *shape recognition.* In architecture, the most important is certainly to recognize outlines and structured symbols. Symbol of bathtub has to be recognized as an independent object. Contours must be identified.

The third level is what we have called objects or concepts : a lemon, a pinnacle, movement etc. From the graphical entities to objects, the system must have a *common sense*, i.e. must give some properties and a signification to the objects. For example, the system should understand that a contour may represent a room. It also has to know that bathtub symbol represents a bathtub.

Finally, the last level of abstraction is the functional model of the design : what it is made up and how it works. From the object to the model the system needs to have some *domain tacit knowledge*, allowing to make semantic connections between objects. For instance, a room with a bathtub should be identified as a bathroom. To feed the evaluators properly there should be *domain explicit knowledge* (driven by the evaluators properties). The model has to know for instance that in a bathroom the windows sills are usually higher to avoid to be seen, and that the temperature has to be higher.

This analyzis methodology is driven by the architecture of a multi-agent system where low-level agents, working with strokes, identify graphical entities. Other agents, taking into account the context of the design, in turn link these entities to concepts or objects. High-level agents combine these concepts to produce the interpretation of the design. The main hypothesis of this structure is that complexity of design can be managed by a bottom-up approach. This assumption presents some limitations, e.g. is it possible to identify graphical entities without situated domain knowledge?

4. Example of this Analysis

We made a trial of our analysis grid on a short moment of the video material sent to us. For this trial, we have considered a very general definition of the model to be constructed, a simple rough fonctionnal model of the object. We analyzed one drawing of the participant A (fig. 2). This drawing has been chosen because it is a quite simple but complete representation of the object.

Fig3. Analyzed drawing

Our analysis is represented by a tree. Each stroke and each graphical entity is labeled, by order of appearance. Each object and each part of the model is labeled depending on what it represents. Arrows determine the links of composition. The whole tree of strokes, entities, objects and model are available in appendix 1.

The strokes properties have also been noted. For this analysis we have just qualified the type of strokes (curve, line…) and their geometric relations to other strokes (parallelism, symmetry…).

5. First Results and Specifications

Although we have analyzed only a single drawing with our method, we have watched the whole set of drawings in the three activities, in order to evaluate the feasibility of our method and to draw some more general conclusions. Although it is quite complex and time consuming, this short study showed us that this method is usable. We expect all the drawings to be analyzable with this method. But some definitions (and principaly the graphical entities) have to be specified more accurately. Several preliminary conclusions can be drawn, and several issues raised :

- The method allows us to be informed on the dynamic of object design. The designer globally draws the object from the top to the bottom (the pinnacle to the glass under the squeezer) and processes by functional elements (the pinnacle, than the lemon, than the base, than the pulp collector…). Do these clues allow us to understand at least partially the design process?
- We can see that almost all graphical entities are built from consecutive strokes (In the schema on appendix 1 we can see than only a few arrows from strokes to graphical entities cross each other). This let us think that a real-time model composition is possible.
- Another result is the characterization of the nature of graphical entities encountered in design. In this case, the drawing can be decomposed in three type of graphical entities : contours (closed or almost closed shapes), isolated strokes or symbols (e.g. arrows), and set of related strokes that doesn't create a contour (e.g. the right "foot" of the support surface). This let us think that it is difficult to have a precise definition of graphical entities prior to observations. These graphical entities emerge from a domain exploration.
- This methodology allows us also to explore the links between graphical entities and objects. These links define the common sense the system should have. For instance, the concept of movement is associated to arrows in the sketches. Therefore arrows have to be recognized as movements. The lemon object is of course present in the three activities, but we notice that it is also graphically expressed in those three activities. The pinnacle has some graphical stable properties between drawings and even between designers and therefore could be recognized by a computer system.
- But it appears also that the links between graphical entities and objects are closely related to the domain. For instance, the pinnacle could be recognized as a rocket in the domain of spatial construction. This leads to two conclusions. At first, it seems difficult to create a generic recognition: models, objects and even graphical entities are domain-specific. Secondly, the grid underlines two levels of analogies on the objects and on the graphical entities. For instance, the lemon (object) could be related to a banana or an orange, and its shape (graphical entity) could be related to an umbrella or a hat.
- We also identified two types of object transformations. From a graphical entity representing an object, adjunction of other graphical entities (strokes inside the shape) can change the same object. These transfor-

mations, as described by Goel [4] on the succession of sketches, are from two types : lateral (change from an idea to another) or vertical (change by growing in precision). For instance, adding details on the pinnacle doesn't change the very nature of its shape but helps to understand its function (vertical transformation, strokes 1 to 7). Changing the triangle base of the squeezer to a concept-like shape (two feet) change the very nature of the object (lateral transformation, strokes 16 to 24). But, if we look just at the structure of strokes and graphical entities, there is no difference between those two types of transformation. Thus a graphical understanding is not sufficient to track the concept evolution. The notion of object and its semantic are necessary.

Finally, we expect that this methodology could help us to define an (almost) exhaustive knowledge necessary to understand the concept of lemon squeezer. By multiplying the studies, we may be able to identify all the different types of graphical entities, all the common sense necessary to identify objects from these entities, and all the knowledge necessary to create a complete model of the object. Our purpose is of course not to create a complete knowledge on lemon squeezers, but if we can build a complete theoretical multi agent system for this narrow domain, we could expect to be able to do so for larger domains, like architecture. Our goal is here to discuss the pertinence of this method on a single domain.

In parallel, it would also be very interesting to better understand the multiple inter-relations existing between the sketches, as well as the relations that exist between the different components of one sketch and the chronological evolution of all of these inputs relations. This "reverse" analysis of the drawing could help us to compose a design process history, from the first idea emergence, through the sequence of the various design steps. This type of analysis could be done in two ways : intra-drawing and between drawings. For instance,

- By understanding the succession of strokes in the time, we could detect patterns of graphical entities design. These patterns can help us to develop shape recognition modules.
- By understanding the changes in the model we could approach the design process, understanding the concepts evolution and the transformations between drawings..
- By understanding the collection of drawing we could detect the main strokes, that are recurrent, and that inform us on the core strokes of the objects being designed.

References

1. Juchmes, R., Leclercq, P., Azar, S. (2005). A freehand sketch environment for architectural design supported by a multi-agent system. Special issue of *Computers and Graphics on Calligraphic Interfaces*, vol 29, num. 6.
2. Leclercq, P. (2004) Invisible Sketch Interface in Architectural Engineering. *Graphic recognition, recent Advanced and Perspectives, Lecture Notes in Computer Science*. LNCS 3088. Springer Verlag, Berlin, 353-363
3. Leclercq, P., Juchmes, R. (2002). The absent interface in design engineering. *AIEDAM Artificial intelligence in engineering design and manufacturing. Special issue : Human-Computer Interaction in Engineering Contexts*. Vol. 16, num. 5.
4. Goel, V. (1995). *Sketches of Thought*. Bradford-MIT Press, Cambridge.

Appendix 1 : Analysis tree