

A new paradigm based on agents applied to free-hand sketch recognition

D.G. Fernández-Pacheco^a, F. Albert^b, N. Aleixos^{b,*}, J. Conesa^a

^a *Graphical Expression Department, Technical University of Cartagena, Cartagena 30202, Spain*

^b *Instituto Interuniversitario de Investigación en Bioingeniería y Tecnología Orientada al Ser Humano, Universitat Politècnica de València, Valencia 46022, Spain*

**Corresponding author. Tel.: +34 96 387 95 14; fax: +34 96 387 75 19.*

E-mail address: naleixos@dig.upv.es (N. Aleixos)

Abstract

Important advances in natural calligraphic interfaces for CAD (Computer Aided Design) applications are being achieved, enabling the development of CAS (Computer Aided Sketching) devices that allow facing up to the conceptual design phase of a product. Recognizers play an important role in this field, allowing the interpretation of the user's intention, but they still present some important lacks. This paper proposes a new recognition paradigm using an agent-based architecture that does not depend on the drawing sequence and takes context information into account to help decisions. Another improvement is the absence of operation modes, that is, no button is needed to distinguish geometry from symbols or gestures, and also "interspersing" and "overtracing" are accomplished.

Keywords: Natural interfaces, free-hand sketch recognition, agent-based systems.

1 Introduction

Some studies in graphic communication field (Barr, 2004; Rose, 2005) promoted by the Engineering Design Graphics Division of the American Society for Engineering Education and by the American Society of Mechanical Engineers (ASME) have concluded that the abilities "to create solid 3D models on a computer" and "to produce free-hand sketches of engineering objects" are mainly the two most highly valued skills that engineering students should be competent in. In turn, the use of sketches during the process of developing new industrial

products and the benefits obtained from its utilization have also been analyzed by other authors (Plimmer & Apperley, 2002; Tversky, 2002).

The arrival of CAD (Computer Aided Design) supposed an enormous change on the lately phases of the product design process, but the impact on the initial phases where the ideas are more important than geometry details, that is, the conceptual design stage, was no significant. Pencil and paper are still used on this conceptual phase to communicate ideas by means of hand-drawn sketches. The main reason for this is due to the lack of CAS (Computer Aided Sketching) tools provided by CAD applications at a commercial and academic level. On other hand, it has been revealed that using CAS tools is, at least, as useful as conventional pencil and paper for developing spatial vision skills in novel engineering students (Contero, Naya, Company, Saorin, & Conesa, 2005). In addition, CAS tools have not been developed as expected since, among others, the hardware necessary to implement them is not well established yet (e.g. Tablet PCs or UMPCs), and the limited capabilities they offer (e.g. strictness in the drawing sequence or low success recognition ratio) do not improve the traditional sketching carried out still nowadays with conventional tools.

So, the challenge of replacing conventional pencil and paper sketches with a digital sketching environment exists. This new environment must be designed in such a way that it favors a “natural” process that does not hinder the user, while also producing its output in the form of a digital design model that can be reused in the remaining phases of the design process.

2 Related work

In the course of creating a product different types of sketches can be used by engineers/designers. Concretely, the classification proposed by Ferguson (1994) distinguishes between “thinking sketches”, which are used to focus and guide non-verbal thought; “talking sketches”, which support the design considerations between colleagues; and “prescriptive sketches”, which are the conveyed instructions that are given to the draughts person, who is the responsible for producing the final version of the engineering drawings. The presented paper is focused on the thinking sketches, where ideas are more important than detailed geometry. At the moment, the field of Sketch-Based Interfaces and Modelling (SBIM) is an emerging area of research, being the creation of 3D models from thinking sketches one of the main objectives.

In order to transform a thinking sketch into a 3D model during the development process of industrial products two approaches can be used: a) geometric reconstruction based modelling techniques by means of the extraction of regularities or “clues” and the assignation of depth information to each one of the vertices, and b) gesture-based modelling methods, where interaction with the user through gestures that are recognized as commands to generate solids from 2D sections is performed. The first approach remains out from paper scope, being the second one the base of the presented work. Some existing systems that make use of gesture-based modelling methods are the SKETCH (Zelevnik, Herndon, & Hughes, 1996), TEDDY (Igarashi, Matsuoka, & Tanaka, 1999) or GIDeS (Pereira, Jorge, Branco, & Nunes, 2000) systems.

Various methods to detect symbols, diagrams, command gestures and geometric shapes are used in sketch recognition. Vuong, He, and Hui (2010) propose a handwriting mathematics system which makes use of an elastic matching algorithm for mathematical symbol recognition. In turn, Apte, Vo, and Kimura (1993) develop a recognizer of hand drawn geometric shapes from several strokes introduced consecutively, basing their classification on thresholds to some established ratio filters. Basing on similar features and making use of a classic linear discriminator, Rubine (1991) classifies single-stroke sketches as digits, letters and basic commands. A prototype for the recognition of glyphs that can be interpreted within the context of several design domains (such as building drawings, mechanic devices or electric circuits) is later described by Gross (1994). Paulson and Hammond (2008) implement the PaleoSketch recognizer, which classifies between eight primitive forms using the “normalized distance between direction extremes” and “direction change ratio” features. Fonseca and Jorge (2000) turn to other features invariant to rotation, such as perimeter, convex hull and area scalar ratios to recognize some shapes. Otherwise, Sun, Liu, Peng, Zhang, and Jianyong (2004) recognize simple geometric shapes and adjust them to the user intention, calculating the average distance from the vertices of the preset shape to the vertices of the stroke. The use of Fourier descriptors in recognizers have demonstrated to achieve better results than methods based on shape descriptors as presented above. Hence, Harding and Ellis (2004) use Fourier descriptors for recognizing hand gesture trajectories. Park and Park (2005) also use them to describe fingerprints that are classified by means of non-linear discriminant analysis, as well as Licsar and Sziranyi (2004) who employ these descriptors in an augmented reality environment to detect the user’s hand movement. In turn, Zhang and Lu (2002) also make use of these descriptors to retrieve images from databases.

Automatic gesture recognition is a complex task since different users can draw the same symbols with a different sequence, shape, size or orientation. In this way, there are several limitations that are present in actual recognizers:

- Recognizer algorithms are not robust and can present ambiguity in the results depending on the user.
- Normally a training previous stage is needed.
- The number of sketched strokes is limited and the input sequence is previously set, reducing freedom to the designer.
- The input stroke is always classified even if it does not exist in the database, assigning the most similar to it and consequently executing a non intended action. The “no-gesture” possibility is missed.
- To change between geometric, command and constraint modes the user normally turns to events as pressure level of the pencil or specific menu buttons.
- The dictionary has a limited number of symbols or gestures.
- There is no possibility of interrupting temporarily the drawing sequence to complete another part of the sketch (known as interspersing).
- They do not behold sketching in an artistic way (overtracing).

In short, actual recognizers are not much robust, no efficient, rigid and addressed to the solution the recognizer has to find out, fixing previously the domain of work. Due to this reason a methodology change in recognizers is needed in order to distinguish and classify sketches like a human person would do it, that is to say, using context information, being flexible at middle and final decisions, and reaching at conflicting situations which force to change the decision by means of consensual agreements. This way of behavior can be assigned to the recognizers by the use of expert systems, object based systems, and agent-based systems, among others.

In the case of agent-based systems, they have been widely used in different applications for process simulation and control such as, for example, the intelligent system for anomaly detection in a combined cycle gas turbine plant developed by Arranz, Cruz, Sanz-Bobi, Ruíz, and Coutiño (2008), or the system for distributed automatic meeting scheduling proposed by Zunino and

Campo (2009). Other works in these fields are done by Gao, Shang, and Kokossis (2009), that make use of agents to develop an intelligent system to support coordinate manufacturing execution and decision-making in chemical process industry, and Wu et al. (2010) benefit from this methodology and present a multi-agent system design and evaluation for collaborative wireless sensor network in large structure health monitoring. Another control application by using agents is proposed by Jiang and Yung (2011), who help senior executives to maintain remote control of their company by means of an agent-based intelligent system. An evaluation of the use of agents is also proposed by Dimou, Symeonidis, and Mitkas (2009), who present an integrated framework based on the Agent Performance Evaluation (APE) methodology for evaluating the performance of agent-based systems.

However, the use of agent-based systems is being extended more and more to recognition processes for supporting natural interfaces. For instance, Achten and Jessurum (2002) use agents for recognition tasks in technical drawings, and Juchmes, Leclercq, and Azar (2005) base their freehand-sketch environment for architectural design on a multi-agent system. Also Mackenzie and Alechina (2003) turn to multi-agent systems in order to classify sketches of animals. Other examples can be found by Azar, Couvreur, Delfosse, Jaspard, and Boulanger (2006), who base their system for sketch interpretation on agents, or by Casella, Deufemia, Mascardi, Gennaro, and Martelli (2008), who interpret sketched symbols using an agent-based system. Finally, Hong, Landay, Long, and Mankoff (2003) describe the drawbacks of sketch recognizers at this time.

The previous systems do not take into consideration other essential symbols used to create 2D sections or 3D geometry (e.g. geometric and dimensional constraints). For this reason the main objective of the recognition paradigm proposed in this paper is to support a calligraphic interface that permits generating geometry from sketches employing standardized symbols as dimensional and geometric constraints, sections, command gestures and different type of lines. This environment will consider context information and will have a flexible structure in order to increase easily the number of symbols in the dictionary. The user will be able to draw freely, taking no account of the number of strokes or the input sequence of them. Furthermore, the annoying change of mode currently implemented in most CAD applications by a button that permits to switch the input mode from geometry to gesture or symbol mode, and also editing mode, is avoided and deduced by the interface. The “interspersing” (interrupt temporarily the

drawing sequence to complete another part of the sketch) and “overtracing” (the use of multiple strokes to represent a single line) methods are also implemented.

This paper is organized as follows. In section 3 we present an overview of the proposed paradigm for recognition of free-hand sketched symbols. A detailed description of the architecture and the implemented agents is covered in section 4. Section 5 describes the experimental work carried out and finally, section 6 shows the conclusions and the expected further work in this field.

3 The recognition process paradigm

To introduce the proposed paradigm spoken and graphical languages are considered, taking into account their similitude. In general, a symbol is made up of one or several simple strokes, as a word of a dictionary is formed by diverse phonemes of an alphabet. So then, one or several phonemes or simple strokes (from now on “primitive symbols”) constitute a word or complex symbol (from now on “combined symbol”), which belongs to a set of accepted words (a dictionary). Consequently, the recognition process will have to consider two levels: a lower level where the phonemes are recognized, and an upper level where the words are deduced. In the Fig. 1 an example of the paradigm is shown, taking into account several combinations of primitive symbols (as they were “phonemes”) for the same combined symbol (as it were a “word” of a dictionary). At the moment of the definition of these primitive symbols, they are chosen as the usual and logical ways of drawing of the designer. It has no sense, for example, to sketch the tangency symbol by means of one stroke only, as it is illustrated in Fig. 1.

Figure 1

Attending the paradigm, the recognition of the primitive symbols will be carried out by the “Primitive Agents”, and the “Combined Agents” will be the chosen for taking care of the recognition of combined symbols. When talking about a primitive symbol (phoneme or simple stroke), a series of digitalized points between a “pen down” and “pen up” event is considered.

Definition of geometry, commands for editing or generating geometry, and depiction of geometric/dimensional constraints are some requirements of a CAD interface. In order to provide all this functionality to the implemented interface, a small index or “dictionary” of primitive and combined symbols has been defined in Fig. 2. These symbols are the more frequently used in

modelling tasks, allowing the user to build parametric geometry from sketches and creating basic solid models. Modelling commands (such as extrusion and revolution), symbols to indicate geometrical restrictions (like concentric and parallel), and annotations (like radial dimension), and symbols to provide artistic support are included.

Figure 2

The primitive symbols of the proposed dictionary can be assembled to form combined symbols in the same manner the phonemes of an alphabet are assembled to constitute words with a semantic meaning. This implies that the last stroke that makes up a combined symbol takes place when the recognition process ends with a valid result. In turn, the user should have some freedom when sketching, not being forced to introduce the strokes in a strict order, neither to introduce all the strokes of a combined symbol at once but allowing the user to interrupt his/her action and return back to it later. In order to support this functionality, an initial implementation of the “interspersing” of strokes from different objects (Sezgin & Davis, 2008) has been implemented in the recognizer, enabling the searching of all the possible solutions on the storehouse of strokes/phonemes (database containing all the user inputs not recognized yet).

“Overtracing” (Ku, Qin, & Wright, 2006) has been also supported by means of the artistic combined symbols, allowing the use of multiple strokes to represent a single line or arc, similar to artistic drawing.

Defining a full catalog is beyond the scope of the present work, but will be mandatory for the system to cope with actual design scenarios. At this end, we should perhaps consider the convenience of following an approach similar to the grammar of images described by Zhu and Mumford (2006) or by Lin, Wu, Porway, and Xu (2009), which solve a hierarchical decomposition from scenes to objects. However, we should remember that most engineering symbols are already perfectly defined and catalogued, as they are standardized (see “technical drawings” at www.iso.ch or www.asme.org).

To note that expanding the catalog of symbols implies revisiting the set of phonemes to ensure, on behalf of efficiency, that we still have the simplest although complete set of phonemes. This is a complex task in itself, and although similar problems have been solved (i.e. in the ambit of character recognition), to our knowledge, nothing has been done in the particular ambit of engineering symbols.

4 The agent-based architecture

The multi-agent architecture offers, by its very nature, the possibility of focusing the recognition tasks of a symbol according to the group of symbols we are interested in detecting. Thus, it is possible to group the symbols in different fields defined in engineering. As an example, we can sketch symbols which allow to carry out modelling operations (extrusion or revolution), symbols related to the structural analysis (fixed unions, articulated unions, sliding unions), symbols related to the industrial design (coils, bearings, toothed wheels), symbols associated with the restrictions of the design (horizontal, vertical, parallel, perpendicular, equal dimension, concentric, tangency) or annotation symbols (radial dimension, diametral dimension, linear dimension).

Given that the main aim is to define a paradigm based on agents which allows designing scalable solutions, the proposed platform presents a structure made up of two levels: on the one hand, a central core where the agents responsible for basic tasks of the system and the primitive agents reside; and on the other hand, the diverse modules where the combined agents form groups.

Looking over at these basic agents, we can find an Interface Agent (IA) which is in charge of the interaction with the user and sends the digitized points to the rest of the basic agents, a Preprocessing Agent (PA) which smoothes and eliminates the noise of each introduced stroke in order to be subsequently analyzed, the Feature Agent (FA) which extracts the most important features through image analysis techniques, the Broker Agent of the Central Core (BACC) which, as its name suggests, assigns the tasks to the agents of the system, and the Intelligent Agent (INA) which receives information from the modules and takes the final decision.

In a similar way, each module is managed by a Broker Agent (BAM) which assigns the tasks to the combined agents responsible for the recognition of each one of the symbols defined in the module and informs the central core of the result of its agents in order to take the final decision.

As the functional diagram of the architecture shows in the Fig. 3, when a user draws a stroke, the Interface Agent (IA) shows it and sends the digitalized points to a data structure. The Broker Agent of the Central Core (BACC) receives the notification and sends the points of the stroke to the Preprocessing Agent (PA). Ideally, these points should be uniformly distributed. However, the faster the stroke is drawn, the fewer points are digitalized, which causes a variation in the points concentration, as well as the drawing may be more or less trembling and halting. The

Preprocessing Agent (PA) must filter and eliminate that noise in order to get an ideal stroke ready for the recognition. Once the preprocessing has been completed, the BACC requests the Feature Agent (FA) to extract the features, which, once it has finished, sends back those features to the BACC. With the extracted features the BACC sends the information to the corresponding primitive agents, which send back the results of the first recognition stage to the BACC. In possession of these results the BACC takes the final decision about if the sketched input matches with a primitive symbol. If so, it sends the results to the broker agents of the modules, which will be referred as BAMs. Specifically, the broker agents BARM, BAMB, BAIDM, BAAM, BATM and BASAM can be easily distinguished in Fig. 3. These broker agents will resend the information to its combined agents, starting the second recognition stage. After finishing, the BAMs agents receive the information from the combined agents and send the results to the Intelligent Agent (INA) in the Central Core, which will take the final decision of the recognition process.

Figure 3

The proposed architecture has been implemented on top of the Jade agent-based platform (Bellifemine, Poggi, & Rimassa, 2000) using Java 1.6. Communication is in charge of messages that are encoded in FIPA-ACL messages (FIPA, 2002), which is a communication language natively supported by Jade. FIPA-ACL specifies both the message fields and the message performatives, which consist of communicative acts such as requesting, informing, making a proposal, accepting or refusing a proposal, and so on. Jade offers the means for sending and receiving messages (both internal and to and from remote computers), in a way that is transparent to the agent developer. It maintains a private queue of incoming ACL messages per agent. Jade also allows the MAS (Multi-Agent System) developer to monitor the exchange of messages using the sniffer built-in agent.

4.1 The Central Core

The first recognition stage of the paradigm, that is to say, the syntactic recognition of the phonemes, takes place precisely in the Central Core. Here, the Feature Agent will extract the most important features of the sketched stroke. Later the Primitive Agents will try to get a

positive matching using these extracted features and send the results to the broker agent (BACC), which will take the final decision for this first recognition phase.

4.1.1 The Feature Agent

Once the sketched stroke is captured by the IA and preprocessed by the PA, the BACC sends the beautified points to the Feature Agent (FA). This Feature Agent has a sequential functioning, in which we have to distinguish two clearly distinctive parts. On the one hand, the “segmentation” of the stroke is carried out extracting the primitive geometrical forms which make it up and providing information about them which will be used in the decision stages afterwards. This segmentation process, which is currently a well-known problem in the sketch recognition field (Pu & Gur, 2009; Company, Varley, Piquer, Vergara, & Sanchez-Rubio, 2009), is achieved by means of the TCVD (Tangent Corner Vertices Detection) algorithm (Fernandez-Pacheco, 2010; Albert, Fernandez-Pacheco, & Aleixos, 2011), of which has been proved its high success ratio in corner and tangent vertices detection. Fig. 4 shows some strokes and its segmentation by the TCVD algorithm.

Figure 4

On the other hand, a series of features or basic “cues” of the introduced stroke are extracted. Only features that are not dependent on scale, position and orientation have been chosen, like the FFT (Tao, Morrow, Heinemann, & Sommer, 1995) of the radius signature (also called polar signature) and of the “arc length versus cumulative turning angle” signature (also known as direction signature). A non-linear discriminant analysis will be used later to classify the symbol/gesture according these features.

4.1.2 The Primitive Agents

Once the extraction of the features of the input stroke is finished, this information is sent to all the Primitive Agents. At least an agent for each primitive symbol has been implemented. Each one of the primitive agents will use the information that it considers significant in order to find relevant clues which allow it to recognize the primitive symbol which it is dealing with. In this way, each primitive agent evaluates the stroke introduced and quantifies its information from two processes:

- A recognition based on the geometry of the stroke that returns a positive or negative coincidence after the syntactic analysis (Match or No_match of the found vertices and the approximated primitives). See Fig. 5.

Figure 5

- A quantitative value as a result of a maximization function based on a non-linear Bayesian discriminatory analysis whose input parameters are the features extracted by the FA (Fernandez-Pacheco, Contero, Naya, & Aleixos, 2008).

These primitive agents evaluate and quantify the coincidence degree of a stroke with its corresponding primitive but they do not take a final decision. The results are sent to the broker agent of the central core, which will decide if the syntactic recognition has a valid candidate.

4.1.3 The Broker Agent of the Central Core

The processes described so far as the data gathering, pre-processing, extraction of features, evaluation and quantification of coincidence with primitives, require the different agents in charge of these tasks to act in a coordinated way. The Broker Agent of the Central Core (BACC) will be responsible for managing all these processes, indicating to each agent when it has to send the requested information, and collecting all the data that each agent returns once its execution is finished.

However, the most important task of the BACC is to decide which primitive symbol has been sketched using the information provided by the primitive agents. Three possible cases can be distinguished:

- Case 1: The candidate primitive agent indicates positive matching in the syntactic recognition and returns the first or second maximum value of the maximization function in relation to the values given by the other primitive agents. In this case, the BACC validates the recognition of the primitive symbol corresponding to that primitive agent.
- Case 2: If the conditions required in the *case 1* are not fulfilled and there are not crosses in the stroke, then the input stroke is considered as a geometric entity; otherwise it is rejected.

- Case 3: Any other case which is not contemplated in the previous ones will cause that the system considers the stroke as geometry if it does not contain any crosses; otherwise it is rejected.

Once the first recognition stage of the paradigm has been accomplished, if the input stroke is validated with a primitive symbol from the dictionary (cases 1 or 2) the BACC will send the result to each one of the modules of the system, starting the second recognition stage of the paradigm where the semantic recognition of the words takes place.

4.2 The modules of the system

The second recognition stage of the paradigm is mainly carried out by the different modules of the system. All these modules have a similar structure (Fig. 6). Inside each one an agent for each symbol with semantic meaning is defined. These symbols are denominated combined symbols, and each agent responsible for their recognition will be designated as its combined agent. These combined agents will compete with each other in order to recognize the combined symbol from the primitive symbols that make it up.

In turn, each module is managed by the Broker Agent of the Module (BAM) which, on the one hand, resends the information of the recognized primitive symbol from the BACC in the central core to the combined agents of its module and, on the other hand, once this information is processed by all the combined agents it sends the results of the module that it manages to the Intelligent Agent in the central core (INA).

Figure 6

4.2.1 The combined agents

From the information that they receive of the analysis carried out by the primitive agents, the combined agents look for the recognition of their combined symbol. These combined agents dispose a list or storehouse where the primitive symbols will be added according as they are recognized, as they were phonemes, until a complete symbol can be conformed, that is to say, until a full semantic meaning is reached.

Hence the system checks the following situations:

- a) If the phoneme is a symbol constituted by only one stroke and it has full meaning by itself, that is, it does not pertain to any more complex symbol, it is assigned and the recognition process finalizes.
- b) If the phoneme is feasible of pertain to other symbols which are formed by several strokes, it is added to the storehouse and the system must wait for the following stroke is introduced, distinguishing three possible cases:
 - 1) If the following stroke can form jointly a combined symbol, then it is added to the storehouse and the actual symbol concludes, updating the status of the agent to “accepted”.
 - 2) If the next stroke is feasible of pertain, but there is not enough semantic meaning to form jointly a combined symbol, then it is aggregated and the status is updated to “in process”.
 - 3) If the consecutive stroke is not able to pertain, the corresponding combined agent updates its status to “rejected”.

Looking closely into the searching process of the combined agents, three levels can be easily differenced:

1. Contextual Analysis

This analysis is based on a set of conditions between the different recognized primitive symbols. As an example, the extrusion combined agent will try to get a positive match looking at the three possible cases shown in the Fig. 7, with no matter of the sequence of introduction of the primitive symbols.

Figure 7

Thereby the different cases are analysed as follows (see Fig. 7):

- a) Case 1: This is the easiest event to be solved by the combined agent. When a primitive “arrow” is recognized, the extrusion combined symbol is immediately identified.
- b) Case 2: In this case the symbol is defined by the union of a primitive “line” and a primitive “angle”. To make sure that the primitives identify correctly the extrusion symbol, the combined agent verifies the following conditions:

- 1) The distance between vertices 2-4 must be lower than the minor distance between vertices 3-4 and 5-4.
 - 2) The angle described by vertices 1-2 must be in the midst of the angles formed by vertices 3-4 and 5-4.
 - 3) The length of the line 1-2 must be at least twice the length of the mayor line between vertices 3-4 and 5-4.
- c) Case 3: The symbol is defined by the combination of three primitive “lines”. In this case the conditions verified by the combined agent are the following ones:
- 1) The distance between vertices 2-4 must be lower than the distance between vertices 3-4.
 - 2) The distance between vertices 2-6 must be lower than the distance between vertices 5-6.
 - 3) The angle described by vertices 1-2 must be in the midst of the angles formed by vertices 3-4 and 5-6.
 - 4) The length of the line 1-2 must be at least twice the length of the mayor line between vertices 3-4 and 5-6.
 - 5) The vertex 4 must be close enough to vertex 6.

2. Postulations for the symbol to have a complete meaning.

Besides the contextual analysis, other aspects are taken into account by the combined agents in order to achieve a positive recognition, as the need of introducing twice a combined symbol, or sketch it near a closed contour.

Taking this into consideration, the combined agents can be classified in three groups:

- Modelling agents: these agents are executed on a previously defined surface, and designate an immediate action. If no prior selection of a surface (closed outline) is introduced, the broker agent of the central core will inform the combined agents and other possibilities for the recognized symbol will be considered. The extrusion and revolution combined agents are included in this group.

- Agents with references: these agents have to be recognized twice consecutively. They set up a relationship between two geometric entities, like parallelism, perpendicularity, equal dimension, tangency and concentricity.
- Independent agents: when no previously introduced symbol or selected surface is needed, agents are considered independent and its recognition supposes an immediate action. This is the case of the following combined agents: vertical, horizontal, radial dimension, linear dimension, diametral dimension, symmetry axis and section.

3. Semantic interpretation.

It must be taken into consideration that, as shown in Fig. 8, a stroke can either represent a symbol by itself or be part of a more complex symbol made up by different primitive symbols, with no matter the sequence in which they are introduced.

For instance, the diametral dimension combined symbol can be recognized from one of the configurations of primitives shown in Fig. 8.

Figure 8

However, considering the dictionary of symbols defined for the annotation module shown in Fig. 9, the diametral dimension symbol could also be part of a more complex symbol, and it will be able to be recognized later by the lineal dimension combined agent.

Figure 9

Briefly, each combined agent will analyze whether the set of primitives in the storehouse identifies its combined symbol, or if it is able to identify the symbol with the subsequent primitives. If that is the case, the primitive symbol will be kept in the list. Otherwise it will be rejected.

Once the aforementioned analyses have been carried out, the combined agents update their status by means of a flag which can take three possible values:

- Rejected: if the primitive cannot be part of the combined symbol.
- In process: if the primitive together with the primitives in the list is capable of constituting the combined symbol.

- Accepted: if the primitive by itself or together with the primitives in the storehouse identifies the combined symbol.

This status update will be informed to the broker agent of the module by each one of the combined agents of a determined module.

4.2.2 The Broker Agents of the Modules

Apart from the combined agents, each module counts on a Broker Agent (BAM) which manages the module and carries out two principal functions. On the one hand, it receives from the BACC the recognized primitive symbol and sends this information to the combined agents of the module.

On the other hand, it manages the results given by each one of the combined agents and informs the intelligent agent in the central core about the status of the module. This status is set by means of a comparison of the flags of the combined agents of the module.

When a broker agent of a module receives a new primitive symbol, this one is sent to the active combined agents of the module, that is, the combined agents whose status flag is “in process” or “accepted”. Once all the status flags of the combined agents are received, the broker agent sends a report to the intelligent agent. This report can contain three states:

- Rejected: if all the flags of the combined agents of the module indicate “rejected”.
- In process: if, at least, a flag of a combined agent of the module indicates “in process”.
- Accepted: if one flag of a combined agent indicates “accepted” and the rest have the “rejected” status.

An example of the states through which the flags of the combined agents and the broker agent of the annotation module are set during the recognition of the lineal dimension combined symbol is given in Fig. 10.

Figure 10

In this example, initially all the agents have the “in process” state, even the broker agent of the module. After the two first strokes, the flag of the radial dimension combined agent gets the “accepted” state, since the combination of an angle and a line, in the sketched relative positions,

defines its symbol. Meanwhile, the flags of the rest of the combined agents of the module are not modified since the combination of the strokes can still be part of their symbols. The BAM remains the “in process” state, granted that at least a combined agent continues being “in process”.

The sketching of the third stroke makes the radial dimension combined agent go to the “rejected” state because the new stroke introduced is not part of its symbol, whereas the diametral dimension combined agent gets the “accepted” state as all the strokes introduced define its symbol. In turn, the lineal dimension combined agent remains the “in process” state waiting for new strokes that define its symbol, and therefore the broker agent of the module does not need to modify its flag.

At the fourth step only the lineal dimension combined agent remains the “in process” state awaiting a new stroke, while the rest of the combined agents are in “rejected” state given that the set of strokes in the storehouse can be part of their combined symbol. In this step the BAM keeps its state.

When the fifth stroke is sketched, the lineal dimension combined agent turns to the “accepted” state as all the strokes define its symbol. Thereby the BAM gets the “accepted” state since there is only one combined agent with the “accepted” flag and the rest with “rejected” state.

4.3 The intelligent agent

The Intelligent Agent (INA) of the central core is the responsible of managing the reports coming from the broker agents of the different modules installed in the platform and of taking the final decision about the recognition of the symbol introduced, establishing a kind of negotiation between the INA and the BAM agents of the different modules.

The procedure carried out by the intelligent agent is based on the information given by the primitive agents and the state of the flags of the BAM agents in the different modules. At the beginning of a recognition process, the BAM agents are initialized to the “in process” state.

For the correct operation of the algorithm it is required to define a storehouse, in which all the strokes the user sketches are introduced. Every time a new stroke is introduced, and after being pre-processed, the BACC agent sends its features to the BAM agents of the active modules, that

is, the modules whose status flag is “in process” or “accepted”. Then, each BAM agent establishes contact with the active combined agents of its module and updates its flag according to the status they reach, informing immediately to the INA agent.

Once the INA agent has received the updating messages from all the active BAM agents it decides if a full semantic recognition has been achieved (one of the modules has the “accepted” state and the rest are “rejected”), if partial semantic recognition is present (at least a module with the “in process” state exists), or on the contrary geometry must be considered (all the modules are in “rejected” state). The outline followed by the INA agent is showed by means of the Fig. 11, in which the next considerations are taken:

a) If the stroke is not recognized by any of the primitive agents:

1. If any of the BAM agents with “in process” state contains a combined agent whose state is “accepted”, its symbol is recognized and the new stroke is considered geometry.
2. Otherwise, the storehouse and the new stroke are considered geometry.

Once the verification is finished the elements of the storehouse are erased.

b) If the stroke is recognized by some of the primitive agents:

1. If the stroke is recognized by the scratch primitive agent:

- i. If any of the BAM agents has a combined agent whose state is “accepted” its symbol is recognized.
- ii. Otherwise, the storehouse is considered geometry.

In any of the previous cases, and a posteriori, the scratch primitive symbol is recognized and the storehouse is erased.

2. Otherwise, the stroke is stacked in the storehouse and the updating of the BAM agents is awaited:

i. If the state of all the BAM agents is updated to rejected:

- a. If any of the BAM agents, before adding the last stroke, had a combined agent whose state was “accepted”, its symbol is recognized, the storehouse

is erased, all the modules are initialized with the “in process” state and the last stroke is newly introduced in order to initialize the recognition process again.

- b. Otherwise, the storehouse is considered geometry and its content is erased.
- ii. If there is any BAM agent whose flag has the “in process” state, the next stroke is awaited.
- iii. If all the BAM agents have the flag in the “rejected” state, except only one that keeps its flag in the “accepted” state, it is assumed that the storehouse is recognized as a combined symbol. The combined agent whose flag is kept in “accepted” state, indicates the kind of the recognized symbol. Once the verification is finished, the storehouse is erased.

Every time the storehouse is erased, the INA agent sends a reset message to the BACC, which will resend the request to the BAM agents of the modules, causing the broker agents and each one of the combined agents to initialize their flag to the “in process” state.

Figure 11

For a better comprehension of the operation of the multi-agent platform, in the Fig. 12 the recognition of a lineal dimensional symbol is analyzed, going into detail on the states through which the flags of the combined and BAM agents are set during the recognition process. In order to facilitate its understanding the “interspersing” mode is not activated in this example.

- Initial state: all the agents are initialized to the “in process” state.
- Step 1: the first stroke makes several combined agents of the restriction module, and the section and artistic arc agents, go to the “rejected” state, given that the introduced vertical line cannot be part of these combined symbols. Since at least one combined agent of each module continues with the “in process” state, all the BAM agents of the modules remain “in process”. Meanwhile, due to at least one BAM agent remains “in process”, the INA agent continues waiting for the entry of a new stroke.
- Step 2: The new primitive line, parallel to the previous one, makes the perpendicular, radial dimension, diametral dimension, extrusion, revolution, symmetry axis and artistic line combined agents go to the “rejected” state, and only the BAM agents of the

restriction and annotation modules remain “in process” state since they have at least one combined agent in this state. The INA agent keeps waiting for a new stroke.

- Step 3: The new stroke introduced causes the horizontal agent goes to the “accepted” state, granted that the set of primitives in the storehouse makes up its symbol. The BAM agent of the restriction module gets also the “accepted” state as its horizontal combined agent has recognized the symbol and the rest of the agents of the module have been “rejected”. As at least a BAM agent remains “in process”, the INA agent continues waiting for the entry of a new stroke.
- Step 4: The recognition as an angle of the new stroke introduced makes the horizontal combined agent go to the “rejected” state along with the BAM agent of its module. Meanwhile the lineal dimension combined agent and the BAM agent of the annotation module remain with the “in process” state. In turn, the INA agent keeps waiting for the entry of a new stroke since at least a BAM agent continues being “in process” state.
- Step 5: The stroke recognized as an angle finishes the symbol of the lineal dimension combined agent, turning into the “accepted” state along with its BAM agent. The INA agent recognizes the lineal dimension symbol since there is only a BAM agent in “accepted” state and the rest are “rejected”. Later it deletes the “storehouse” and initializes all the agents to the “in process” state.
- Step 6: A new stroke, recognized as a primitive arrow, starts the recognition process again, giving rise to all the agents of the restriction and artistic modules go to the “rejected” state, keeping the agents of the annotation module with “in process” state and changing the extrusion and BAM agents of the modelling module to the “accepted” state. The INA agent waits for the entry of a new stroke given that at least a BAM agent remains “in process” state.

Figure 12

The “interspersing” mode, that is, the possibility of leaving the symbol unfinished and going on with it later, has been implemented in a first phase by means of the inclusion of an “interspersing parameter”. This parameter indicates the number of strokes that can be awaited for the acceptance of a symbol. In order to comprehend the operating mode of “interspersing”, an

example is illustrated in Fig. 13. Initially, the interspersing value is set to one, indicating that the user can interrupt the sketching of a symbol drawing one stroke with no relation to the previous ones, and come back later to the unfinished symbol to complete it. In the third step of Fig. 13, the introduction of a geometry arc causes the possibility of an interspersing process, and the interspersing parameter is reduced in one unit. Consequently, all the possible combined agents remain with the “in process” state, although the introduced arc cannot belong to the current symbol, waiting for a new input. The introduction of a circle in the fifth step and the no opportunity of interspersing (its parameter has reached the zero value) implies the confirmation of the diametral dimension symbol. Once the recognition is done, the interspersing parameter is reset to the initial value.

When selecting the interspersing parameter value, it is not recommended a value above one unit, since in this case, the broker and combined agents would wait for many input strokes as the interspersing parameter indicates. This could cause the user not to see his/her actions executed immediately, and the feedback function of the interface would be interrupted, generating undesired delays and frustration.

Figure 13

5 Experimental work and results

The paradigm proposed in this article has been implemented and evaluated in two differentiated stages. A first stage where the syntactic recognition carried out by the primitive agents is evaluated and a second phase where the semantic and final recognition by the combined agents and the intelligent agent of the system is also rated.

In the tests performed for the first stage evaluation 10 cad users have taken part, introducing each one of these users several reiterations of all the primitive symbols of the dictionary (see Fig. 2). About 2200 primitive symbols with different orientations and sizes have been used for the evaluation. Aside another set of primitive symbols have been used to train the system off-line in order to calculate the optimized values of some classification functions which will be used as cues by the primitive agents later in the on-line process.

The results of this first recognition stage are shown in Table 1, which shows success ratio in classification for each phoneme in the diagonal cells (highlighted in black), and the percentage of misclassification in the rest.

Table 1

From these results, the average success ratio achieved in the recognition of the primitive symbols of the dictionary was of 96.41%. Main errors in classification correspond to cases in which the sketched symbol is wrongly digitalized. Also errors can be due to symbols that are wrong interpreted as for instance: an unclosed circle that can be confused with an arc; an arc with high radius value that can be converted to a line; an arrow or round-arrow symbols that can be confused depending if the main long stroke is more or less curved, independently of the user's intention.

By other side, the tests accomplished for the evaluation of the second recognition stage, or final evaluation of the interface, were conducted with 9 CAD users. Each user introduced several occurrences of each one of the different combined symbols of the Fig. 2 (a total of 2500 symbols) with different orientations and sizes. Once introduced, the implemented interface classified them obtaining the results shown in Table 2. For the evaluation of the interface, any possibility of the combined symbols was allowed, with no matter the sequence of introduction of the strokes or the combination of them (for example, an angle can also be introduced by means of two lines).

Hence, the results of the final recognition stage are shown in Table 2, which shows success ratio in classification for each combined agent in the diagonal cells (highlighted in black), and the percentage of error in the rest of the cells. The average success ratio achieved with the proposed recognizer is 96.8%.

Table 2

As can be confirmed from Table 2, the implementation of the proposed paradigm makes possible the correct classification of the sketched symbols in a calligraphic interface, with the exception of those cases in which the sketched form is not recognized as a word of the dictionary, being classified as geometry. In this case the user should delete it and sketch again the intended form in a better way.

Anyway, thanks to the cues extracted by the feature agent and the primitive agents, and the context information analyzed by the combined agents, the consideration of the non-intended “no gesture” possibility, impossible to detect in other recognizers, has been solved.

An overview of the implemented interface for the recognizer proposed in this paper can be seen in Fig. 14, where the recognition of several gestures intended by the user are successfully detected.

Figure 14

6 Conclusions

The paper proposes a new recognition paradigm for a sketch-based environment using an agent-based architecture. The recognition process has been supported by two levels of agents: a lower level where the “Primitive Agents” reside and are in charge of the syntactic recognition, that is, the low level recognition, and an upper level where the “Combined Agents” remain and carry out the semantic recognition using contextual information, that is, the high level recognition. The proposed recognizer has been designed in order to be easily extensible by means of new primitive agents (if the required phoneme is not included in the dictionary a new primitive agent can be created) or new combined agents (the creation of one combined agent is necessary for each new added symbol/gesture).

The recognizer for natural interface developed in this work is also endowed with the “interspersing” (the user can leave the symbol unfinished and go on with it later on) and “overtracing” (the user can draw several strokes to represent one single line or arc, similar to artistic drawing) utilities. The proposed recognizer does not depend on the number of strokes and neither on the sketching sequence of user inputs, providing more flexibility and freedom to the drawing process. Besides, no operation modes are required, in the way that no buttons are needed to change the input mode to introduce geometry, symbols or commands, making the environment more intuitive without the existence of complex menus.

This paradigm has been implemented in a Java platform using JADE, obtaining a success ratio of 96.41% for primitive symbols and 96.8% for combined symbols. A total of 2500 symbols have

been sketched by different users with different orientations and sizes. The obtained results reveal the interest of the proposed paradigm in the sketch-based environment.

Acknowledgements

The Spanish Ministry of Science and Education and the FEDER Funds, through the CUESKETCH project (Ref. DPI2007-66755-C02-01), partially supported this work.

References

- Achten, H. H., & Jessurun, A. J. (2002). An agent framework for recognition of graphic units in drawings. In *Proceedings of 20th International Conference on Education and Research in Computer Aided Architectural Design in Europe* (pp. 246-253). Warsaw.
- Albert, F., Fernandez-Pacheco, D. G., & Aleixos, N. (2011). A New Method to Find Corner and Tangent Vertices in Sketches using parametric cubic curves approximation. Manuscript submitted for publication.
- Apte, A., Vo, V., & Kimura, T. D. (1993). Recognising Multistroke Geometric Shapes: An Experimental Evaluation. In *Proceedings of the ACM (UIST'93)* (pp. 121-128). Atlanta, Georgia.
- Arranz, A., Cruz, A., Sanz-Bobi, M. A., Ruiz, P., & Coutino, J. (2008). DADICC: Intelligent system for anomaly detection in a combined cycle gas turbine plant. *Expert Syst Appl*, 34 (4), 2267-2277.
- Azar, S., Couvreur, L., Delfosse, V., Jaspert, B., & Boulanger, C. (2006). An agent-based Multimodal interface for sketch interpretation. In *IEEE Workshop on Multimedia Signal Processing* (pp. 488-492). New York: IEEE.
- Barr, R. E. (2004). The Current Status of Graphical Communication in Engineering Education. In *34th ASEE/IEEE Frontiers in Education Conference* (Vol. 3, pp. S1D8-S1D13). Savannah.
- Bellifemine, F., Poggi, A., & Rimassa, G. (2000). Developing multi-agent systems with JADE. In *Seventh International Workshop Agent Theories Architectures and Languages (ATAL2000)* (Lecture Notes in Computer Science ed., Vol. 1986, pp. 89-103). Berlin: Springer-Verlag.
- Casella, G., Deufemia, V., Mascardi, V., Gennaro, C., & Martelli, M. (2008). An agent-based framework for sketched symbol interpretation. *J Visual Lang Comput*, 19 (2), 225-257.
- Company, P., Varley, P. A. C., Piquer, A., Vergara, M., & Sanchez-Rubio, J. (2009). Benchmarks for Computer-based Segmentation of Sketches. In *Proceedings of the Eighth IAPR International Workshop on Graphics Recognition (GREC)* (pp. 103-114). France.
- Contero, M., Naya, F., Company, P., Saorin, J. L., & Conesa, J. (2005). Improving visualization skills in engineering education. *IEEE Comput Graph*, 25 (5), 24-31.

- Dimou, C., Symeonidis, A. L., & Mitkas, P. A. (2009). An integrated infrastructure for monitoring and evaluating agent-based systems. *Expert Syst Appl*, 36 (4), 7630-7643.
- Ferguson, E. S. (1994). *Engineering and the Mind's Eye*: The MIT Press.
- Fernandez-Pacheco, D. G. (2010). *Aplicacion para reconocimiento de bocetos basada en sistemas multi-agente*. Unpublished Ph.D. dissertation, Technical University of Valencia, Valencia.
- Fernandez-Pacheco, D. G., Contero, M., Naya, F., & Aleixos, N. (2008). A Calligraphic Interface in a Sketch-Based Modelling Environment. In *20º Congreso Internacional de Ingeniería Gráfica INGEGRAF*. Valencia.
- FIPA, O. R. G. (2002). *FIPA ACL Message Structure Specification*.
- Fonseca, M. J., & Jorge, J. (2000). Using Fuzzy Logic to Recognise Geometric Shapes Interactively. In *Proceedings of 9th IEEE Conference on Fuzzy Systems* (Vol. 1, pp. 291-296). San Antonio, Texas.
- Gao, Y., Shang, Z. G., & Kokossis, A. (2009). Agent-based intelligent system development for decision support in chemical process industry. *Expert Syst Appl*, 36 (8), 11099-11107.
- Gross, M. D. (1994). Recognising and Interpreting Diagrams in Design. In *Proceedings of ACM (AVI'94)* (pp. 88-94). Bari, Italy.
- Harding, P. R. G., & Ellis, T. J. (2004). Recognising Hand Gesture Using Fourier Descriptors. In *Proceedings of the 17th International Conference on Pattern Recognition* (Vol. 3, pp. 286-289).
- Hong, J., Landay, J., Long, A. C., & Mankoff, J. (2003). Sketch Recognisers from the End-User's, the Designer's, and the Programmer's Perspective. In *AAAI Spring Symposium*.
- Igarashi, T., Matsuoka, S., & Tanaka, H. (1999). Teddy: A sketching interface for 3D freeform design. In *Siggraph 99 Conference Proceedings* (pp. 409-416). New York: Assoc Computing Machinery.
- Jiang, X. N., & Yung, K. L. (2011). A new intelligent system for senior executives to maintain remote control of their company. *Expert Syst Appl*, 38 (1), 736-742.
- Juchmes, R., Leclercq, P., & Azar, S. (2005). A freehand-sketch environment for architectural design supported by a multi-agent system. *Comput Graph-Uk*, 29 (6), 905-915.
- Ku, D. C., Qin, S. F., & Wright, D. K. (2006). Interpretation of Overtracing Freehand Sketching for Geometric Shapes. In *14th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision WSCG* (pp. 263-270). Plzen, Czech Republic: Univ West Bohemia.
- Licsar, A., & Sziranyi, T. (2004). Hand gesture recognition in camera-projector system. In *Computer Vision in Human-Computer Interaction, Proceedings* (Vol. 3058, pp. 83-93). Berlin: Springer-Verlag Berlin.
- Lin, L., Wu, T. F., Porway, J., & Xu, Z. J. (2009). A stochastic graph grammar for compositional object representation and recognition. *Pattern Recogn*, 42 (7), 1297-1307.

- Mackenzie, G., & Alechina, N. (2003). Classifying sketches of animals using an agent-based system. In *Computer Analysis of Images and Patterns, Proceedings* (Vol. 2756, pp. 521-529). Berlin: Springer-Verlag Berlin.
- Park, C. H., & Park, H. (2005). Fingerprint classification using fast Fourier transform and nonlinear discriminant analysis. *Pattern Recogn*, 38 (4), 495-503.
- Paulson, B., & Hammond, T. (2008). PaleoSketch: accurate primitive sketch recognition and beautification. In *Proceedings of the 13th international conference on Intelligent user interfaces* (pp. 1-10). Gran Canaria, Spain: ACM.
- Pereira, J., Jorge, J., Branco, V., & Nunes, F. (2000). Towards calligraphic interfaces: sketching 3D scenes with gestures and context icons. In *8th International Conference in Central Europe on Computer Graphics, Visualization and Interactive Digital Media WSCG*. Plzen, Czech Republic: University of West Bohemia.
- Plimmer, B., & Apperley, M. (2002). Computer-aided sketching to capture preliminary design. In *Proceedings of the Third Australasian conference on User interfaces - Volume 7* (pp. 9-12). Melbourne, Victoria, Australia: Australian Computer Society, Inc.
- Pu, J. T., & Gur, D. (2009). Automated freehand sketch segmentation using radial basis functions. *Comput Aided Design*, 41 (12), 857-864.
- Rose, A. T. (2005). Graphical communication using hand-drawn sketches in civil engineering. *J Prof Iss Eng Ed Pr*, 131 (4), 238-247.
- Rubine, D. (1991). SPECIFYING GESTURES BY EXAMPLE. In *Siggraph 91 Conference Proceedings* (Vol. 25, pp. 329-337). New York: Assoc Computing Machinery.
- Sezgin, T. M., & Davis, R. (2008). Sketch recognition in interspersed drawings using time-based graphical models. *Comput Graph-Uk*, 32 (5), 500-510.
- Sun, Z., Liu, W., Peng, B., Zhang, B., & Jianyong, S. (2004). User Adaptation for Online Sketchy Shape Recognition. *Graphics Recognition* (Vol. 3088, pp. 305-316): Springer Berlin / Heidelberg.
- Tao, Y., Morrow, C. T., Heinemann, P. H., & Sommer, H. J. (1995). Fourier-Based Separation Technique for Shape Grading of Potatoes Using Machine Vision. *Trans of the Amer Soc Agric Eng*, 38 (3), 949-957.
- Tversky, B. (2002). What do Sketches say about Thinking? In *AAAI Spring Symposium Series - Sketch Understanding* (pp. 148-152).
- Vuong, B. Q., He, Y. L., & Hui, S. C. (2010). Towards a web-based progressive handwriting recognition environment for mathematical problem solving. *Expert Syst Appl*, 37 (1), 886-893.
- Wu, J., Yuan, S. F., Ji, S., Zhou, G. Y., Wang, Y., & Wang, Z. L. (2010). Multi-agent system design and evaluation for collaborative wireless sensor network in large structure health monitoring. *Expert Syst Appl*, 37 (3), 2028-2036.
- Zelevnik, R. C., Herndon, K. P., & Hughes, J. F. (1996). SKETCH: An interface for sketching 3D scenes. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques* (pp. 163-170): ACM.

- Zhang, D., & Lu, G. (2002). A comparative Study of Fourier Descriptors for Shape Representation and Retrieval. In *Proc. of 5th Asian Conference on Computer Vision (ACCV)* (pp. 646-651): Springer.
- Zhu, S.-C., & Mumford, D. (2006). A stochastic grammar of images. *Found. Trends. Comput. Graph. Vis.*, 2 (4), 259-362.
- Zunino, A., & Campo, M. (2009). Chronos: A multi-agent system for distributed automatic meeting scheduling. *Expert Syst Appl*, 36 (3), 7011-7018.

List of Figures:

Figure 1. Recognition process paradigm

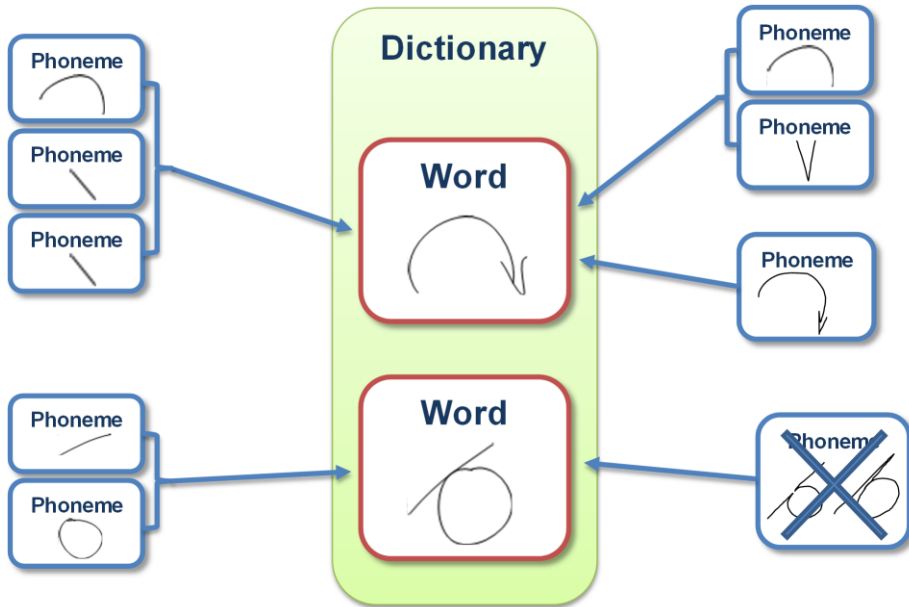


Figure 2. Primitive and Combined symbols of the implemented Dictionary

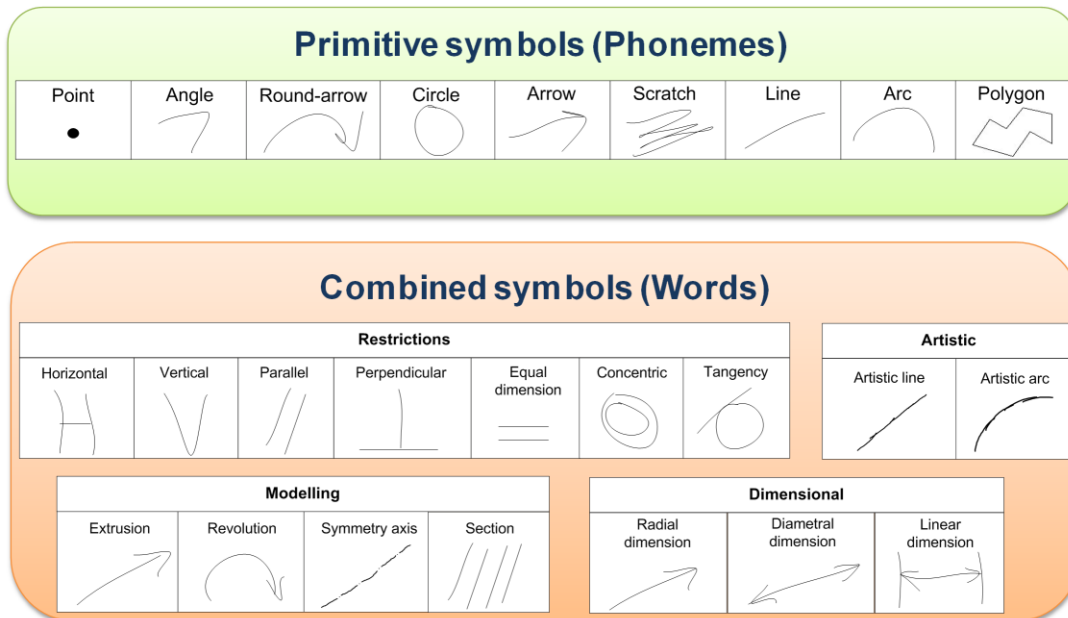


Figure 3. Functional diagram of the agent-based architecture

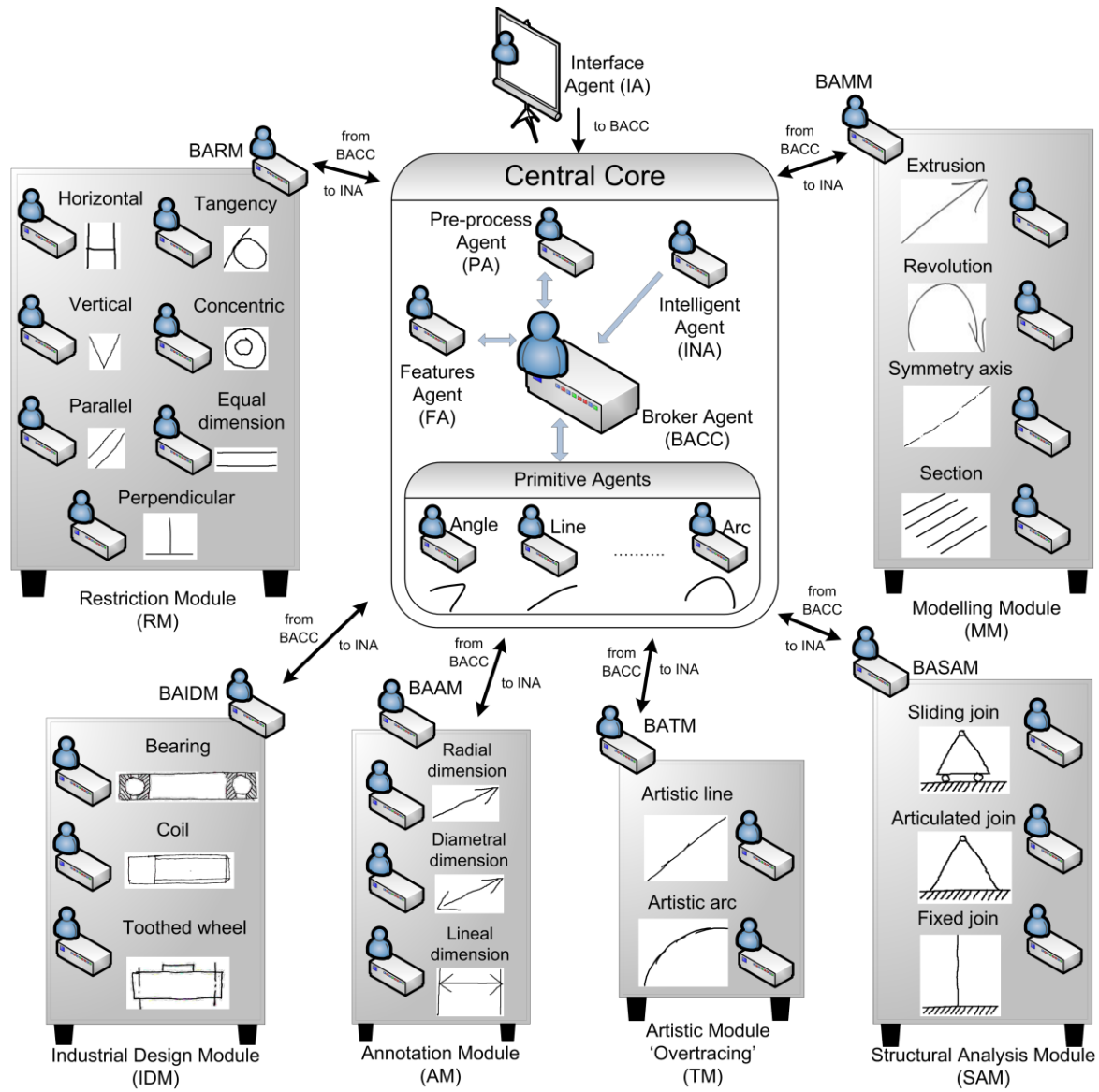


Figure 4. Segmentation of the sketched strokes by the TCVD algorithm

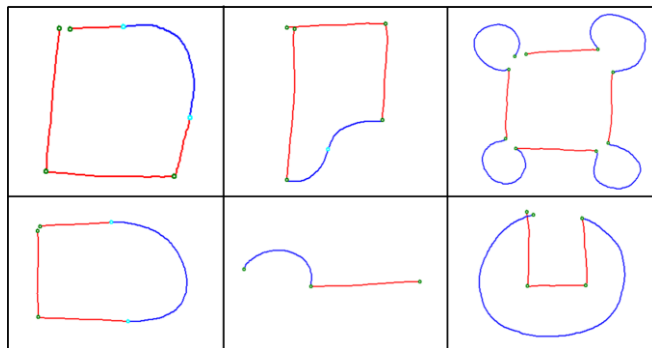


Figure 5. Match or No_match of the found vertices and the approximated primitives

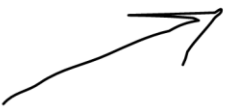
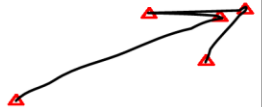

Sketched symbol	Real segmentation	Result from syntactic recognition
		 Positive MATCH

Figure 6. Structure of the modules

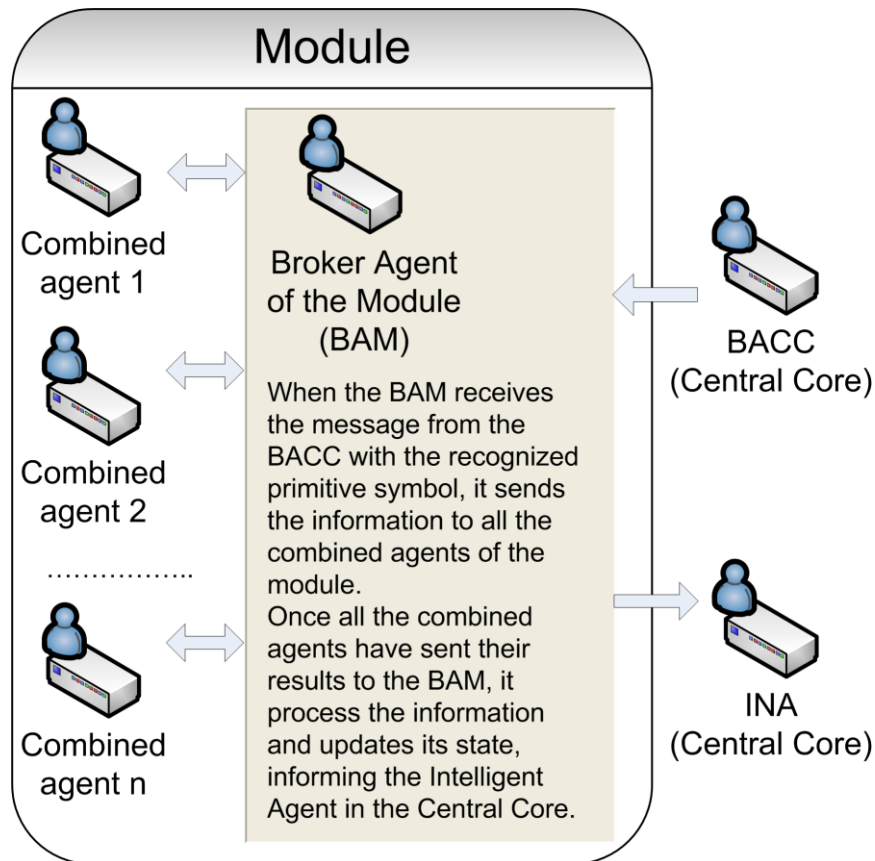


Figure 7. Possibilities of sketching the extrusion combined symbol

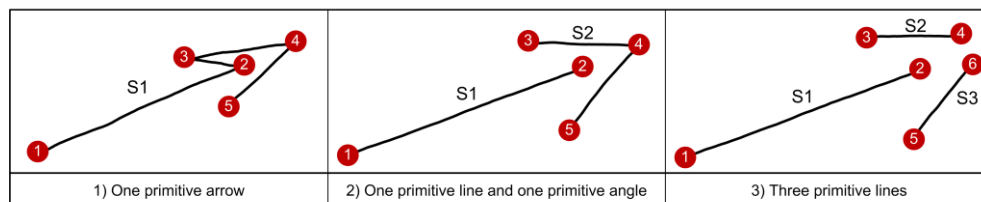


Figure 8. Possible configurations for the diametral dimension combined symbol


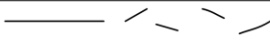
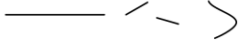
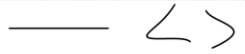


Diametral dimension combined symbol	Different inputs for diametral dimension combined symbol (no matter the sequence)	
	Set of five 'line' primitive symbols	
	Set of three 'line' and one 'angle' primitive symbols	
	Set of one 'line' and two 'angle' primitive symbols	
	Set of two 'line' and one 'arrow' primitive symbols	
	Set of one 'angle' and one 'arrow' primitive symbols	

Figure 9. Dictionary of symbols of the annotation module







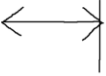
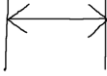
Annotation module		
Radial dimension	Diametral dimension	Lineal dimension
		

Figure 10. An example of the status flags in the annotation module

	Initial state		Step 1		Step 2		Step 3		Step 4		Step 5	
Strokes												
Annotation module	BAM	∞	BAM	∞	BAM	∞	BAM	∞	BAM	∞	BAM	A
	Radial	∞	Radial	∞	Radial	√	Radial	X	Radial	X	Radial	X
	Diametral	∞	Diametral	∞	Diametral	∞	Diametral	√	Diametral	X	Diametral	X
	Lineal	∞	Lineal	∞	Lineal	∞	Lineal	∞	Lineal	∞	Lineal	√

- "Rejected" Status
- "In Process" Status
- "Accepted" Status

- Agents that receive primitives
- Agents that do not receive primitives

Figure 11. Flowchart of the decision-taking of the INA agent

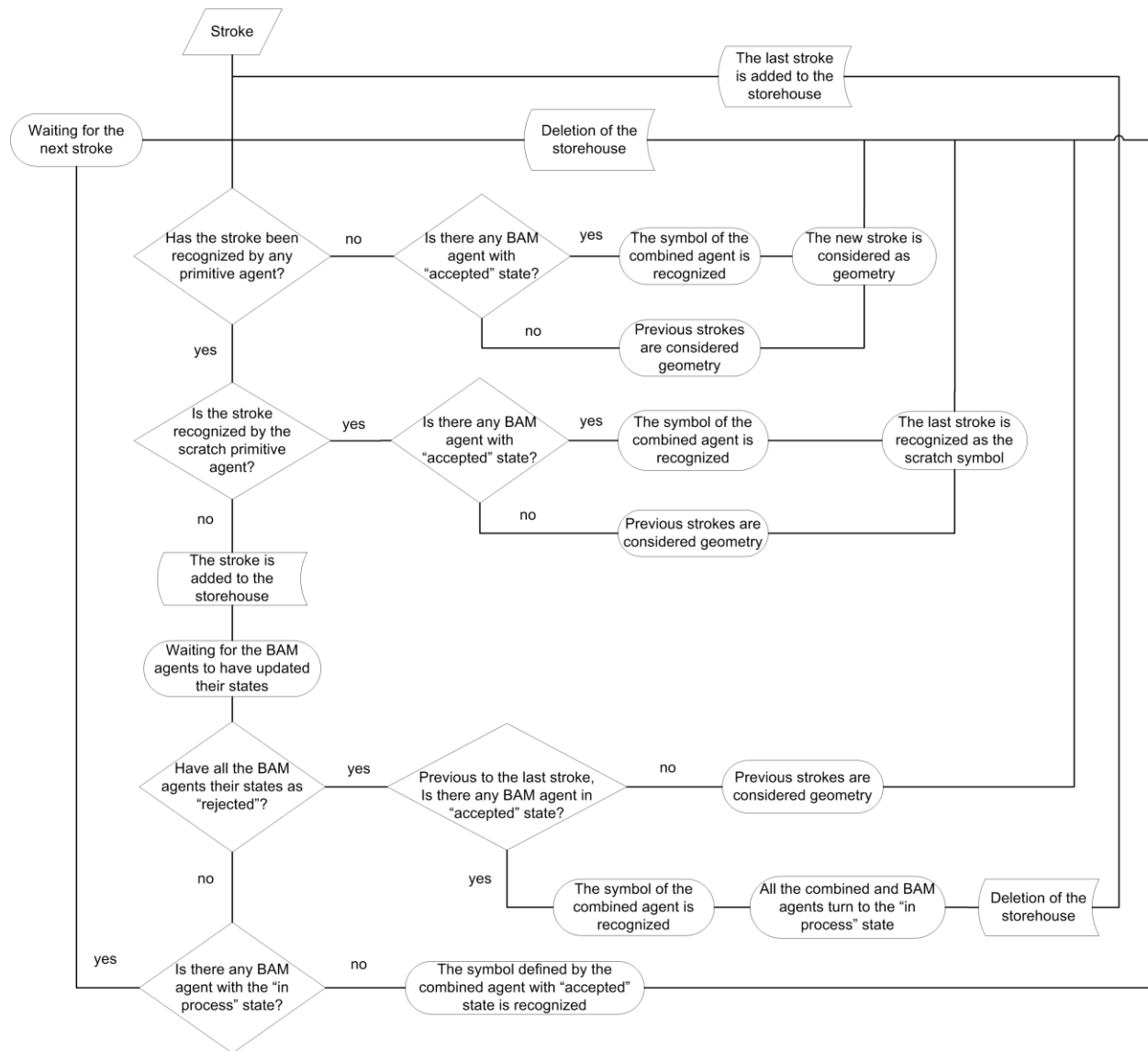


Figure 12. States of the flags during the recognition of a lineal dimension symbol (Active modules: restriction, annotation, artistic and modelling)

	Initial state	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6						
Stroke													
Flags													
Agents of the Restriction Module	BAM	∞	BAM	∞	BAM	√	BAM	X	BAM	X	∞	BAM	X
	Horizontal	∞	Horizontal	∞	Horizontal	∞	Horizontal	√	Horizontal	X	∞	Horizontal	X
	Vertical	∞	Vertical	X	Vertical	X	Vertical	X	Vertical	X	∞	Vertical	X
	Parallel	∞	Parallel	X	Parallel	X	Parallel	X	Parallel	X	∞	Parallel	X
	Tangency	∞	Tangency	X	Tangency	X	Tangency	X	Tangency	X	∞	Tangency	X
	Concentric	∞	Concentric	X	Concentric	X	Concentric	X	Concentric	X	∞	Concentric	X
	Eq. Dimen	∞	Eq. Dimen	X	Eq. Dimen	X	Eq. Dimen	X	Eq. Dimen	X	∞	Eq. Dimen	X
Perpendic.	∞	Perpendic.	∞	Perpendic.	X	Perpendic.	X	Perpendic.	X	∞	Perpendic.	X	
Agents of the Annotation Module	BAM	∞	BAM	∞	BAM	∞	BAM	∞	BAM	√	∞	BAM	∞
	Radial	∞	Radial	∞	Radial	X	Radial	X	Radial	X	∞	Radial	√
	Diametral	∞	Diametral	∞	Diametral	X	Diametral	X	Diametral	X	∞	Diametral	∞
Lineal	∞	Lineal	∞	Lineal	∞	Lineal	∞	Lineal	∞	∞	Lineal	√	∞
Agents of the Modelling Module	BAM	∞	BAM	∞	BAM	X	BAM	X	BAM	X	∞	BAM	√
	Extrusion	∞	Extrusion	∞	Extrusion	X	Extrusion	X	Extrusion	X	∞	Extrusion	√
	Revolution	∞	Revolution	∞	Revolution	X	Revolution	X	Revolution	X	∞	Revolution	X
	Sym. Axis	∞	Sym. Axis	∞	Sym. Axis	X	Sym. Axis	X	Sym. Axis	X	∞	Sym. Axis	X
Section	∞	Section	X	Section	X	Section	X	Section	X	∞	Section	X	
Agents of the Artistic Module (Overracing)	BAM	∞	BAM	∞	BAM	X	BAM	X	BAM	X	∞	BAM	X
	Artistic line	∞	Artistic line	∞	Artistic line	X	Artistic line	X	Artistic line	X	∞	Artistic line	X
	Artistic arc	∞	Artistic arc	X	Artistic arc	X	Artistic arc	X	Artistic arc	X	∞	Artistic arc	X
Action of the INA agent	Waiting for stroke	Waiting for stroke	Waiting for stroke	Waiting for stroke	Waiting for stroke	Waiting for stroke	Recognition as "Lineal dimension"	Waiting for stroke					

- "Rejected" Status
- "In Process" Status
- "Accepted" Status

- Agents that receive primitives
- Agents that do not receive primitives

Initialization

Figure 13. Recognition of a diametral dimension symbol implementing the “interspersing” mode

	Initial state	Step 1	Step 2	Step 3	Step 4	Step 5						
Stroke												
Flags												
Agents of the Restriction Module	BAM	∞	BAM	∞	BAM	x	BAM	x	BAM	x	BAM	x
	Horizontal	∞	Horizontal	∞	Horizontal	x	Horizontal	x	Horizontal	x	Horizontal	x
	Vertical	∞	Vertical	x	Vertical	x	Vertical	x	Vertical	x	Vertical	x
	Parallel	∞	Parallel	x	Parallel	x	Parallel	x	Parallel	x	Parallel	x
	Tangency	∞	Tangency	x	Tangency	x	Tangency	x	Tangency	x	Tangency	x
	Concentric	∞	Concentric	x	Concentric	x	Concentric	x	Concentric	x	Concentric	x
	Eq. Dimen.	∞	Eq. Dimen.	∞	Eq. Dimen.	x	Eq. Dimen.	x	Eq. Dimen.	x	Eq. Dimen.	x
Perpendic.	∞	Perpendic.	∞	Perpendic.	x	Perpendic.	x	Perpendic.	x	Perpendic.	x	
Agents of the Annotation Module	BAM	∞	BAM	∞	BAM	∞	BAM	∞	BAM	∞	BAM	∞
	Radial	∞	Radial	∞	Radial	∞	Radial	∞	Radial	x	Radial	x
	Diametral	∞	Diametral	∞	Diametral	∞	Diametral	∞	Diametral	∞	Diametral	∞
	Lineal	∞	Lineal	∞	Lineal	∞	Lineal	∞	Lineal	∞	Lineal	∞
Agents of the Modelling Module	BAM	∞	BAM	∞	BAM	∞	BAM	∞	BAM	x	BAM	x
	Extrusion	∞	Extrusion	∞	Extrusion	∞	Extrusion	∞	Extrusion	x	Extrusion	x
	Revolution	∞	Revolution	x	Revolution	x	Revolution	x	Revolution	x	Revolution	x
	Sym. Axis	∞	Sym. Axis	∞	Sym. Axis	x	Sym. Axis	x	Sym. Axis	x	Sym. Axis	x
	Section	∞	Section	x	Section	x	Section	x	Section	x	Section	x
Agents of the Artistic Module (Overtracing)	BAM	∞	BAM	∞	BAM	x	BAM	x	BAM	x	BAM	x
	Artistic line	∞	Artistic line	∞	Artistic line	x	Artistic line	x	Artistic line	x	Artistic line	x
	Artistic arc	∞	Artistic arc	x	Artistic arc	x	Artistic arc	x	Artistic arc	x	Artistic arc	x
Action of the INA agent	Waiting for stroke	Waiting for stroke	Waiting for stroke	Interspersing	Waiting for stroke	Recognition as “Diametral dimension”						
Interspersing value	1	1	1	0	0	0						

- “Rejected” Status
- “In Process” Status
- “Accepted” Status
- Agents that receive primitives
- Agents that do not receive primitives

Figure 14. Examples of the sketch recognition interface for several gestures: a) extrusion, b) revolution and c) deletion

Table 2. Results of the semantic (and final) recognition by the combined agents (given in percentages)

		Restriction Module						Modelling Module				Annotat. Module		Artistic Module		Central Core		
		Concentric	Tangent	Vertical	Horizontal	Parallel	Perpendicular	Equal Dimension	Section	Symmetry Axis	Revolution	Extrusion / Radial Dim.	Lineal Dimension	Diametral Dimension	Artistic Line	Artistic Arc	Geometry	Scratch
Recognized Symbols / Sketched Symbols	Concentric	96.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.3	0.0	
	Tangent	0.0	99.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	
	Vertical	0.0	0.0	98.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	0.0	
	Horizontal	0.0	0.0	0.0	96.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.3	0.0	
	Parallel	0.0	0.0	0.0	0.0	100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
	Perpendicular	0.0	0.0	0.0	0.0	0.0	97.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	0.0	
	Equal Dimension	0.0	0.0	0.0	0.0	0.0	0.0	100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
	Section	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
	Symmetry Axis	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	93.0	0.0	0.0	0.0	0.0	0.0	7.0	0.0	
	Revolution	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	98.0	0.0	0.0	0.0	0.0	2.0	0.0	
	Extrusion/Rad.Dim	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	99.0	0.0	0.0	0.0	1.0	0.0	
	Lineal Dimension	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	90.0	0.0	0.0	10.0	0.0	
	Diametral Dim.	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	97.0	0.0	3.0	0.0	
	Artistic Line	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	95.0	5.0	0.0	
	Artistic Arc	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	90.0	10.0	
	Geometry	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100	0.0
	Scratch	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.0	96.0	