

An Agent-Based Paradigm for Free-Hand Sketch Recognition

D.G. Fernández-Pacheco¹, J. Conesa¹, N. Aleixos², P. Company³, and M. Contero²

¹DEG. Universidad Politécnica de Cartagena, 30202 Cartagena, España

²Instituto en Bioingeniería y Tecnología Orientada al Ser Humano
(Universidad Politécnica de Valencia), España

³EMC. Universitat Jaume I, 12071 Castellón, España

{daniel.garcia,julian.conesa}@upct.es, naleixos@dig.upv.es,
pcompany@emc.uji.es, mcontero@dig.upv.es

Abstract. Natural interfaces for CAD applications based on sketching devices have been explored to some extent. Former approaches used techniques to perform the recognition process like invariant features extracted with image analysis techniques as neural networks, statistical learning or fuzzy logic. Currently, more flexible and robust techniques are being introduced, which consider other information as context data and other relationships. However, this kind of interfaces is still scarcely extended because they still lack scalability and reliability for interpreting user inputs.

Keywords: CAD, Natural interfaces, Agent-based systems, Gesture and symbol recognition.

1 Introduction

New devices created to substitute the traditional pen and paper support Calligraphic Interfaces [1], but automatic gesture recognition is a complex task since different users can draw the same symbols with a different sequence, shape, size or orientation, and still remains as a very active research field. Due to these inconveniences, many recogniser algorithms are not robust or present ambiguity in the results. To avoid this variability, techniques based on digital image processing can be applied. Moreover, the problems related to the sketch scale or orientation has to be solved by means of techniques that remain invariant in the face of these features [2]. Shape description from morphological features as length, width, perimeter, area, inertial moments, bounding box, etc., presents the drawback of dependency on the scale or orientation, which increases the percentage of mistakes in the classification stage of the algorithm [3], so invariant morphological features are more suitable for this purpose. For example, Fourier descriptors are an example of invariant features [4], and they have been largely used as a general technique for image information compression or classification of regular shapes [5] or handwriting characters [6]. There are also a set of regular invariant moments widely used as contour-based shape descriptors. These are the descriptors derived by Hu [7]. Along the time, some works are carried out using these

techniques in sketch recognition to detect symbols, diagrams, geometric shapes and other user command gestures. Some of them are scale and rotation dependent, others use invariant features related to shape factor ratios, Fourier descriptors or invariant moments, and others can accept multiple strokes but in a strict input order ([8-13]). As traditional techniques for classification, we can find the fuzzy logic, distances from ideal shape, linear and non-linear discriminant analysis, etc. ([14-17]). These traditional techniques have the drawback of being rigid and the more symbols are contained in the catalog the worse results are obtained.

As an alternative to traditional techniques, we can find a technology based on agents. The agent based systems have been widely used for process simulation, process control, traffic control and so on, but their use is being extended more and more to recognition processes for supporting natural interfaces. The main benefits we can take profit of are the flexibility of these systems and the autonomy of agents. This technology is being used even more for natural interfaces, as for instance Juchmes et al. [18] who base their freehand-sketch environment for architectural design on a multi-agent system. Also Achten and Jessurum [19] use agents for recognition tasks in technical drawings. So do Mackenzie et al. [20] in order to classify sketches of animals. Other examples can be found in Azar et al. [21], who base their system for sketch interpretation on agents, or in Casella et al. [22], who interpret sketched symbols using an agent-based system.

Our current goal is defining a new paradigm which is a) aimed at interpreting hand-drawn sketches used by product designers during the creative stages of design, and b) able to design a scalable solution. Hence, our aim is defining an agent-based structural approach, so as to take profit of the flexibility of agents systems and the autonomy of individual agents. Our structural approach works in three levels: a) *basic* agents in the low level act to partially recognise/classify simple strokes and distribute some recognition tasks, b) *primitive* agents in the intermediate level use these results for finding the syntactic meaning of strokes, and c) the upper level contains the *combined* agents aimed at obtaining the semantic meaning of groups of strokes.

This paper is organised as follows: first the recognition process is explained, second the structure of agent levels is shown and, finally, some reasoning about this methodology is given.

2 The Recognition Process Paradigm

To take advantage of the flexibility and autonomy of hierarchical multi-agent architecture for recognition process, a hierarchical decomposition for symbols is defined. So, symbols are made out of lines, where lines are like “phonemes” of an alphabet. One or several phonemes make a symbol (i.e. a “word”), that belongs to a set of symbols (a “dictionary”). In this way, our structural approach uses *basic* agents to recognise/classify lines, *primitive* agents for finding their syntactic meaning, and *combined* agents to obtaining the symbols they belong to.

What we call lines or “primitive symbols” are simple geometric shapes that appear repeatedly in the symbols. On the other hand, strokes are sets of points that are digitised by the input device between a consecutive pen-down and pen-up event. For the

sake of simplicity, along the rest of this paper, we have assumed the equivalence between lines and strokes. This simplistic assumption is valid as far as the current goal is just testing whether the design and implementation of the agents-architecture is feasible for interpreting hand-drawn symbols. Figure 1 shows the set of lines/strokes used as phonemes.

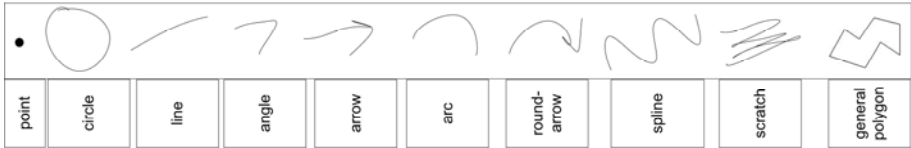


Fig. 1. Phonemes

However, it is clear that the final system will not continue assuming that each pen stroke represents a single line. Overtracing and segmentation must be considered. Overtracing is the use of multiple strokes to represent a single line. Readers interested in this topic can find a recent contribution by Ku et al. [23]. Segmentation is the process of dividing a complex stroke into its geometrical primitives. Segmentation of sketches is an open problem in the process of sketch recognition. Recent contributions can be found in [24-25].

To compose phonemes into words, i.e. to identify symbols, we must find the semantic meaning of every group of phonemes. This means that the last stroke for a symbol occurs when the recognition process finishes with a valid result. An improvement we accomplish is allowing *interspersing* of strokes from different objects. Readers interested in this topic can find recent advances in [26].

Finally, we have defined a catalog or “dictionary”, as shown in figure 2. Again, we have just defined just a small catalog of symbols, although representative enough to test the implementation of our agents-architecture. The symbols considered are representative of the modeling tasks that allow the users to construct parametric geometry from sketches and to create basic solid models. They include command gestures (as extrusion, scratch-out, etc.), symbols conveying information about dimensions and geometric constraints (collinear, parallel, etc.), and symbols conveying other geometrical information (as axis, etc.).

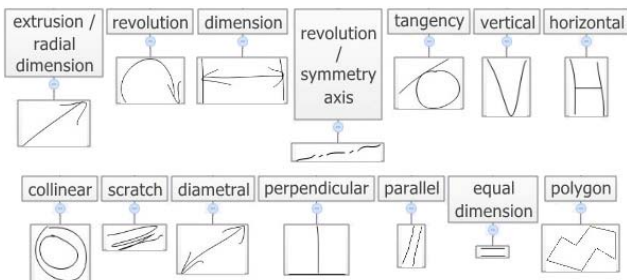


Fig. 2. Dictionary of gestures/symbols

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 [27] or Lin et al. [28], 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 [29]), to our knowledge, nothing has been done in the particular ambit of engineering symbols.

3 Agent Structure

As our goal is defining an agent-based paradigm proficient to design scalable solutions, we have defined an agents-architecture that works in three levels: low level for the *Basic Agents* aimed at interface, pre-processing and feature extraction; intermediate level for the *Primitive Agents* in charge of syntactic meaning of every single stroke, and the upper level that holds the *Combined Agents* in charge of semantic meaning of several strokes forming a symbol. All agents in the system run in parallel.

3.1 Basic Agent Level

Four agents work at this level:

- The **Broker Agent** (BA) that distributes the tasks between all agents living in the different levels of the system.
- The **Interface Agent** (IA) that supports the interaction with the user and provides the stroke points to the remaining basic agents.
- The **Pre-processing Agent** (PA) that beautifies every stroke in order to remove the noise and prepares them for further operations.
- The **Feature Agent** (FA) that extracts relevant features by means of image analysis techniques.

Regarding to the BA, it manages the delivering task in the system. Figure 3 shows the operating diagram of the BA for the Basic and Primitive agent levels.

When a user draws a stroke, the IA displays it and sends the digitised points to a global data structure. The BA is notified and sends the stroke points to the PA that produces the beautified stroke. Ideally, stroke points should be uniformly distributed. But the faster the speed a stroke is introduced at, the fewer points are digitised. That causes a varying concentration of points. To solve this problem, the PA detects and removes isolated points by estimating the distance between each point and their neighbors. Then PA passes a smoothing mask along the points of each stroke to avoid the effect of tremors. At the end, the line equation between two consecutive digitised points is calculated, and the gaps are filled in with new points.

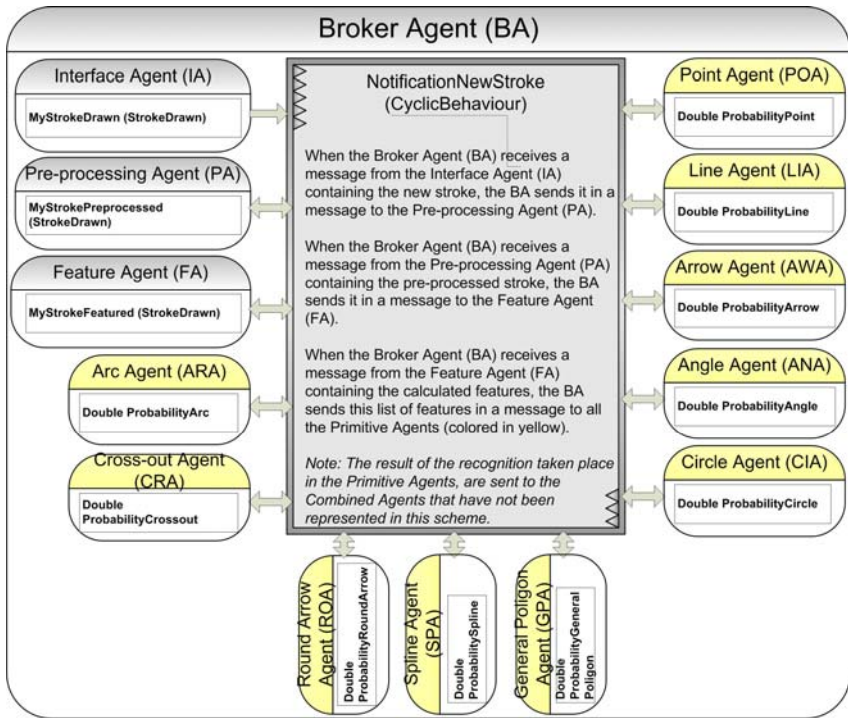


Fig. 3. Broker Agent operating diagram for Basic and Primitive agent levels

When beautification is done, BA asks for feature extraction to the FA. Once the FA has finished, the BA is informed and it passes the features to the Primitive Agents. When finished, the Primitive Agents pass their results to the BA, which will let them available for the Combined Agents to perform the second recognition step.

The feature extraction is sequential. The FA normalises the stroke points to a fixed number of points (FFT algorithm requires an input with $2n$ points) and then performs its analysis to calculate the FFT [30], and other features invariant to position, scale and orientation as Hu moments [7], perimeter, circularity, and so on [31]. Centroid, radius signature and the arc length versus cumulative turning angle signature [32] are also calculated. To note that not all the features will be used by all the Primitive Agents. They will use the significant ones to find their relevant cues, in order to recognise the primitive symbol they are in charge of.

3.2 Primitive Agent Level

Recognition of simple strokes is performed in this level. At least one Primitive Agent has been implemented for each primitive symbol (figure 4). The names of these agents have been chosen in the following way: the two first digits for the phoneme to recognise and the third digit is an A for Agent. So, the Primitive Agent that finds a point will be called POA (PO for point and A for Agent).

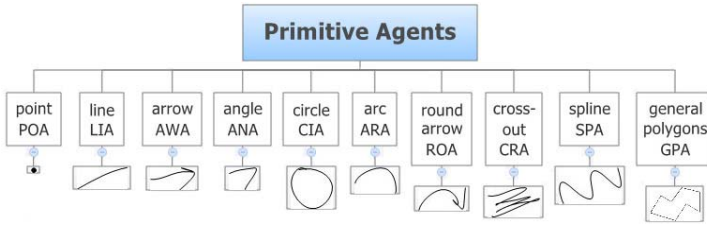


Fig. 4. Primitive Agents level

These agents compete against each other, to give at least one valid solution to the Combined Agents. Using the features provided from the Basic Agent level, each Primitive Agent finds cues to detect its primitive symbol. When these agents finish, three conditions may happen. If agents did not find results, that is, no primitive symbol was recognised, then, the system remains waiting for the user to sketch again. If just one primitive symbol was recognised, then the upper level tries to give significance with the existing context to recognise one combined symbol. Finally, if more than one primitive symbol was recognised, then the upper level must give significance to one of them depending on the existing context. The others are ignored.

3.3 Combined Agent Level

The recognition of symbols is performed in this level. As in the Primitive level, there is at least one Combined Agent for each symbol to recognise (figure 5).

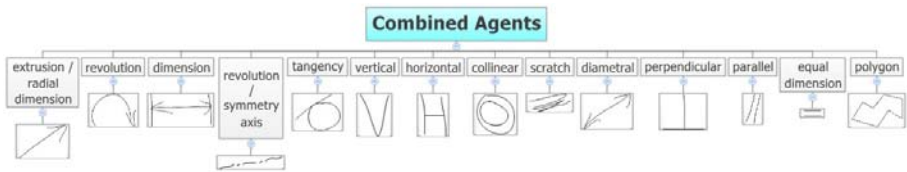


Fig. 5. Combined Agents level

One iteration of the symbol recognition process starts when a stroke is drawn. It is first beautified and normalised, and, after that, their features are extracted by means of image analysis techniques. Then, it is recognised in two steps: as a simple stroke, and as a component of a symbol. In the first step, every stroke is matched to a specific phoneme. In the second step, consecutive phonemes are collected until they form a full symbol, i.e. until they reach a full semantic meaning. For the second step, the system checks out the following cases:

- If the phoneme is a single-stroke symbol (i.e. scratch or polygon) it is assigned and the recognition of the current symbol finishes.
- If it is suitable to pertain to some multiple-stroke symbol, then, it is added, and the system must wait for the next stroke to be introduced:

- If the next stroke is suitable to form a combined symbol altogether, then, it is added and the current symbol finishes.
- If the next stroke is suitable to pertain to, but there is not semantic meaning enough to form a combined symbol altogether, then, it is added and the search continues.
- If the next stroke is not suitable to pertain to, it is ignored, and the search continues.

Note that every stroke may be suitable to pertain to more than one multiple-stroke symbol, and the same symbol may be drawn by way of different strokes (figure 6). Hence, all of the valid candidates are searched *in parallel*, until one of them reaches a full semantic meaning. The recognition system implements the blackboard paradigm, where the decision is taken depending on the set of cues that have the maximum significance. The user can draw the strokes in any order, so the recogniser does not depend on the sequence.

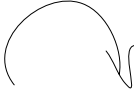



Symbol for revolution command	Generated by means of the combination of the following primitive symbols
	One "arc" primitive symbol and two "line" primitive symbols 
	One "arc" primitive symbol and one "angle" primitive symbol 
	One "round-arrow" primitive symbol 

Fig. 6. Possibilities of sketching the revolve command gesture

4 Agent Architecture Implementation

The agents-architecture is described in Figure 7. The Basic Agents (already detailed in figure 3), are colored in yellow and delimited by dotted lines. The intermediate level manages the Primitive Agents (PA), and the upper level manages the Combined Agents (CA). Their output is given to the BA, which passes the final result to the external CAD editor or Sketched-Based Modeling system though the IA agent.

The platform is implemented on top of the Jade agent-based platform [33] using Java 1.6. Communication is in charge of messages that are encoded in FIPA-ACL messages [34], 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 developer to monitor the exchange of messages using the *sniffer* built-in agent.

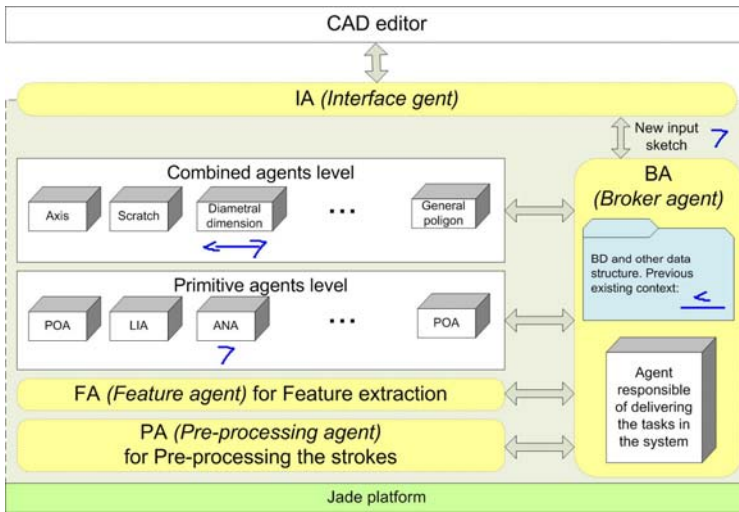


Fig. 7. The agent sketch system architecture

5 Reasoning about This Paradigm

The proposed paradigm consists of two main parts: the recognition of simple strokes, and the recognition of (multi-stroke) symbols.

To recognise simple strokes, several Primitive Agents compete to each other. Every agent collects cues related to its own stroke type and scores the probability of the stroke to belong to such a type. The phoneme reported to be the most probable is selected.

At this stage, the main difference from traditional recognition techniques as fuzzy logic, neural networks or other statistical learning is that specific algorithms are implemented to detect different phonemes. Traditional techniques are more rigid as they use the same classification method to recognise different symbols, that is, the same neural network or the same discriminant analysis is executed to classify different things. Whereas with this paradigm the classification method may be customized for each type of symbol to recognise, what can foresee a higher success ratio.

In the recognition of (multi-stroke) symbols, the main difference from traditional techniques is that in order to recognise a symbol, a specific code is executed, since there is one combined agent for each symbol. The agents are autonomous and adaptive too, and therefore capable of changing their decision depending on the context.

Another benefit of programming with agents is the flexibility of the system. We have tested that we are able to add new symbols to our catalog, just adding the corresponding Combined Agent for its recognition. Adding a new symbol to the recognition system (as for example a X symbol), is as easy as adding its corresponding Combined Agent to the upper level, as it may use the implemented alphabet to classify the symbol.

If new strokes (phonemes) are also required, they are just added to the alphabet implementing its corresponding Primitive Agent. Moreover, in a traditional environment, the recognition process is carried out sequentially, but in here we have determined the specific tasks of the recognition process and have assigned them to different agents that can act independently, avoiding the sequential process and arranging the recognition

process to execute in a more logical way. Hence, this method is not dependent on the number of strokes neither on the sketching sequence order.

6 Conclusions

Our current goal is defining a scalable solution aimed at interpreting hand-drawn symbols used by product designers during the creative stages of design.

At this end, we have described a hierarchical multi-agent architecture organised in three levels: the Basic Agents, the Primitive Agents and the Combined Agents. An experimental version was successfully implemented to demonstrate the validity of the architecture. Besides, the goal of flexibility was successfully checked too.

The next step will be benchmarking the effectiveness of this strategy compared to other current successful approaches. Finally, although the informal tests seem quite promising, the scalability of the approach should be properly checked.

Acknowledgments. 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

1. Contero, M., Naya, F., Jorge, J., Conesa, J.: CIGRO: A minimal instruction set calligraphic interface for sketch-based modeling. In: Kumar, V., Gavrilova, M.L., Tan, C.J.K., L'Ecuyer, P. (eds.) ICCSA 2003. LNCS, vol. 2669, pp. 549–558. Springer, Heidelberg (2003)
2. Kan, C., Srinath, M.D.: Invariant character recognition with Zerkine and orthogonal Fourier-Mellin moments. *Pattern Recognition* 35, 143–154 (2002)
3. Blasco, J., Aleixos, N., Moltó, E.: Machine vision system for automatic quality grading of fruit. *Biosystems Engineering* 85(4), 415–423 (2003)
4. Gonzalez, R.C., Woods, R.E.: *Digital Image Processing*, 2nd edn. Prentice Hall, Upper-Saddle River (2002)
5. Mokhtarian, F., Abbasi, S.: Robust automatic selection of optimal views in multi-view free-form object recognition. *Pattern Recognition* 38, 1021–1031 (2005)
6. Chen, G.Y., Bui, T.D., Krzyzak, A.: Rotation invariant pattern recognition using ridgelets, wavelet cycle-spinning and Fourier features. *Pattern Recognition* 38, 2314–2322 (2005)
7. Hu, M.: Visual pattern recognition by moment invariants. *IRE Trans. Inf. Theor.* IT-8, 179–187 (1962)
8. Rubine, D.H.: Specifying Gestures by Example. *Computer Graphics*. In: Proceedings of the SIGGRAPH 1991, vol. 25(4), pp. 329–337 (1991)
9. Ajay, A., Vo, V., Kimura, T.D.: Recognising Multistroke Shapes: An Experimental Evaluation. In: Proceedings of the ACM (UIST 1993), Atlanta, Georgia, pp. 121–128 (1993)
10. Gross, M.D.: Recognising and Interpreting Diagrams in Design. In: Proceedings of ACM (AVI 1994), Bari, Italy, pp. 88–94 (1994)
11. Zang, D., Lu, G.: A Comparative Study of Fourier Descriptors for Shape Representation and Retrieval. In: The 5th Asian Conference on Computer Vision, ACCV 2002, Melbourne, Australia, pp. 1–6 (2002)
12. Harding, P.R.G., Ellis, T.J.: Recognising Hand Gesture Using Fourier Descriptors. In: Proceedings of the 17th International Conference on Pattern Recognition, vol. 3, pp. 286–289 (2004)
13. Zion, B., Shklyar, A., Karplus, I.: Sorting fish by computer vision. *Comput. Electron. Agric.* 23, 175–187 (1999)

14. Fonseca, M.J., Jorge, J.: Using Fuzzy Logic to Recognise Geometric Shapes Interactively. In: Proceedings of 9th IEEE Conference on Fuzzy Systems, vol. 1, pp. 291–296 (2000)
15. Xiangyu, J., Wenyin, L., Jianyong, S., Sun, Z.: On-Line Graphics Recognition. In: Pacific Conference on Computer Graphics and Applications, pp. 256–264 (2002)
16. Zhengxing, S., Liu, W., Binbin, P., Bin, Z., Jianyong, S.: User Adaptation for Online Sketchy Shape Recognition. In: Lladós, J., Kwon, Y.-B. (eds.) GREC 2003. LNCS, vol. 3088, pp. 305–316. Springer, Heidelberg (2004)
17. Park, C.H., Park, H.: Fingerprint classification using fast Fourier transform and non-linear discriminant analysis. *Pattern Recognition* 38, 495–503 (2005)
18. Juchmes, R., Leclercq, P., Azar, S.: A freehand-sketch environment for architectural design supported by a multi-agent system. *Computers & Graphics* 29(6), 905–915 (2005)
19. Achten, H.H., Jessurun, A.J.: 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 (eCAADe 2002), Warsaw, pp. 246–253 (2002)
20. Mackenzie, G., Alechina, N.: Classifying sketches of animals using an agent-based system. In: Petkov, N., Westenberg, M.A. (eds.) CAIP 2003. LNCS, vol. 2756, pp. 521–529. Springer, Heidelberg (2003)
21. Azar, S., Couvreur, L., Delfosse, V., Jaspartz, B., Boulanger, C.: An agent-based multimodal interface for sketch interpretation. In: Proceedings of International Workshop on Multimedia Signal Processing (MMSP 2006), British Columbia, Canada (2006)
22. Casella, G., Deufemia, V., Mascardi, V., Costagliola, G., Martelli, M.: An agent-based framework for sketched symbol interpretation. *Journal of Visual Languages and Computing* 19, 225–257 (2008)
23. Ku, D.C., Qin, S.F., Wright, D.K.: Interpretation of Overtracing Freehand Sketching for Geometric Shapes. In: WSCG 2006. Full papers proceedings, vol. G41, pp. 263–270 (2006)
24. Pu, J., Gur, D.: Automated Freehand Sketch Segmentation Using Radial Basis Functions. *Computer-Aided Design* (2009), doi:10.1016/j.cad.2009.05.005
25. Company, P., Varley, P.A.C., Piquer, A., Vergara, M., Sanchez-Rubio, J.: Benchmarks for Computer-based Segmentation of Sketches. In: Pre-Proceedings of the Eighth IAPR International Workshop on Graphics Recognition GREC 2009, pp. 103–114 (2009)
26. Sezgin, T.M., Davis, R.: Sketch recognition in interspersed drawings using time-based graphical models. *Computers and Graphics* 32(5), 500–510 (2008)
27. Zhu, S.C., Mumford, D.: A stochastic grammar of images. *Foundations and Trends in Computer Graphics and Vision* 2(4), 259–362 (2006)
28. Lin, L., Wu, T., Porway, J., Xu, Z.: A stochastic graph grammar for compositional object representation and recognition. *Pattern Recognition* 42(7), 1297–1307 (2009)
29. Ku, D.C., Qin, S.F., Wright, D.K.: Interpretation of Overtracing Freehand Sketching for Geometric Shapes. In: WSCG 2006. Full papers proceedings, vol. G41, pp. 263–270 (2006)
30. Tao, Y., Morrow, C.T., Heinemann, P.H., Sommer, H.J.: Fourier-Based Separation Technique for Shape Grading of Potatoes Using Machine Vision. *Transactions of the ASAE* 38(3), 949–957
31. Wojnar, L., Kurzydłowski, K.J.: Practical Guide to Image Analysis. ASM International, pp. 157–160 (2000) ISBN 0-87170-688-1
32. Yu, B.: Recognition of freehand sketches using mean shift. In: Proc. of IUI 2003, pp. 204–210 (2003)
33. Bellifemine, F., Poggi, A., Rimassa, G.: Developing multi-agent systems with JADE. In: Castelfranchi, C., Lespérance, Y. (eds.) ATAL 2000. LNCS (LNAI), vol. 1986, pp. 89–103. Springer, Heidelberg (2001)
34. FIPA ORG, FIPA ACL Message Structure Specification, Document no. SC00061G (2002)