Classifying Sketches of animals using an Agent-Based System

Graham Mackenzie and Natasha Alechina

Nottingham University, Nottingham, NG8 2BB, UK,
gjm@cs.nott.ac.uk,
WWW home page: http://www.cs.nott.ac.uk/~gjm

Abstract. A technique for the classification and understanding of child-like sketches of animals, using a live pen-based input device is demonstrated. A method of segmenting the sketch using curve differentials in addition to pen speed is shown. Once this stage is completed, an agent-based search of the resultant data begins in an attempt to isolate recognisable high-level features within the sketch. The context between probable matches for these features is used by the agents in assessing their success in identifying features correctly.

1 Introduction

This paper describes work in progress on classifying objects in free-hand drawings or sketches. We have concentrated on recognising drawings of animals: classifying a free-hand sketch as a cat, or a dog, or a fish, etc.

The drawings we use are discretised pen drawings (not optical images) although the general methodology proposed in the paper could be easily adapted for optical images. At the moment the sketches are represented as sets of components which in turn are arrays of coordinate points. We plan to move to the UNIPEN format[5] in the near future.

This work is intended to explore possibilities of using agent techniques in conjuction with context resolution more generally in computer vision. The decision to use sketch based input was taken simply to reduce the problem in scope - leaving the agent-based context resolution system independant of less relevant low-level computer vision problems.

Our method of classification uses a hierarchical database of animal body parts (heads, tails, paws,...) which is described in section 4. The classification proceeds as follows:

– parse a sketch into a set of likely meaningful components on the basis of their geometry;
– match those components to generalised animal body parts contained in a database;
– on the basis of the matches of components, match the overall sketch to a particular animal; if this proves difficult, return to the previous step.

Parsing of the sketch into components is done using two different techniques. This is described in 2. Classifying components of a sketch as heads, tails or other animal parts is a hard combinatorial problem. To avoid brute force matching of every component to every entry in the database we use a technique involving ‘agents’, able to communicate via a blackboard-like architecture [6, 10]. Each component in the database has an associated agent. Agents move around the sketch getting ‘more confident’ if they seem to be successfully matching their component and also if they encounter other agents in the neighbourhood which they are more comfortable with, (e.g. eyes and ears) and don’t encounter agents they are ‘repulsed’ by (e.g. an ear and a tail). Having reached a certain level of confidence the agents ‘settle’ on their their component. Completely unsuccessful agents are removed from the sketch and replaced by agents corresponding to other components. Another agent then recognises the arrangement of other agents as an animal and gives us a final result.

Unlike most visual classifiers this one attempts to achieve recognition by 'understanding' the image presented to it, at least to a degree. Once a successful match is achieved, we can expect that most of the component parts of the sketch will also have been identified correctly.

More details on the agents control is given in section 5. The algorithm for comparing a stroke in the database and a stroke in the sketch is described in section 3. Some simple experimental results are described in section 6.

2 Parsing a sketch

The input to our system is via a 'pen-based' interface, rather than a scan of a previously drawn sketch. Data arrives as a series of cartesian points grouped into strokes, which together comprise a complete sketch. What we need to do, in essence, is search through the strokes finding and separating out probable high-level features from the collection of strokes. We make the fairly strong assumption that there must be some change of direction or other interference within a pen-stroke to denote a transition from one high-level feature to another, or else a completely new pen-stroke. By separating every combination of change of directions we have a collection of stroke fragments, of which some tiny portion will be complete, identifiable high-level features.

Each sketched 'stroke' is first of all processed to locate probable corners or changes from one subsection of the sketch to another. This is achieved using a combination of two separate methods:

– we measure the speed of the pen at each point of the sketcher's stroke. The sketcher's pen speed drops significantly at corners.
– By measuring the perpendicular distance from a chord across a small section of the stroke to the stroke itself at sample points along the sketch we determine points on the stroke when a significant change of direction has occurred.

Each potential corner is flagged. It is preferable to be over-sensitive in corner-detection, rather than under-sensitive at this stage, since spurious corners can be ignored during the matching stage. We then examine each drawn stroke's relationships with others - attempting to establish points in the sketch that were intended by the sketcher to connect, and ignoring those when sections of line were supposed merely to come close together. Points of probable intersection are flagged along with the corners. Clusters of close flagged points are 'joined' to form a single point to reference that particular event.

What results is a network of edges drawn by the sketcher, and nodes that represent a either a possible change from one primitive component of the sketch to another, or an event of potential significance within one such component.

Once we have this 'graph-like' structure describing a sketch we can consider plausible paths through it as possible components representing high-level features.

We introduce a simple grammar used to describe these components in a concise manner. Each component extracted from the sketch consists of sections of line and the joins between sections. Our grammar, therefore contains these two basic symbols. Each line is expanded to provide information regarding its length (relative to the sketch size) and its curvature (direction and magnitude). Each 'join' provides the angle between the two adjoining line sections.

A grammar for the description of components could therefore be expressed as:

```
component = line | component join component
line = (length, curvature)
join = (angle)
```

where length, curvature and angle are real numbers.

Each line-section is treated thus:

Given a list of coordinate points $C[1..n]$ between two 'join' points A and B we

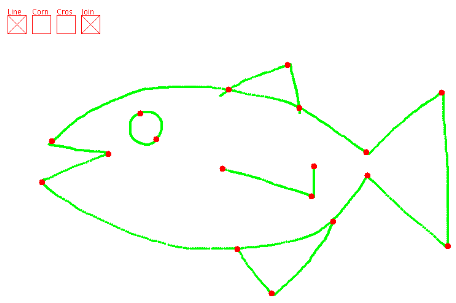

Fig. 1. Parsing a sketch to identify points of interest.

- assess the curvature: measure the perpendicular distance between each point in C and the straight-line AB and find the maximum value.
  Curvature direction is determined by assessing whether any point $C[i]$ between A and B is to the 'right' or 'left' of AB from the perspective of travelling along AB.
- find the length: the distance along C from A to B (not just the length of AB).

For each corner, we must record the change in direction at corners: examine a corner Y by inspecting straight lines XY and YZ (where X, Y, and Z are corners on C). The angle at Y is ordained to be the change in direction required to change from bearing XY to bearing YZ.

3 Matching

It is easy enough to define a matching algorithm that will return a positive match between two components if, and only if, they are exactly the same (after scaling). Assuming the starting points of both components are fixed, we could match their grammatical representations section by section and return a positive match if they are identical.

In reality we want to match successfully two components that the artist/sketcher intended to represent the same concept. This means that the matching process should be relaxed: lengths of the sections, angles between them, and curvature of the lines may vary within some limits, which depend on how precise a match we need for a particular feature.

We mentioned earlier that when parsing we aim to be over-sensitive in our detection of corners and other 'points of interest'. When matching two components with non-equal sizes we can remove corners from the more numerous in an attempt to facilitate a match, imposing a penalty on the match likelihood for each corner removed.

Currently, the matching algorithm returns three parameters: how well the lengths of components matched, how well the curvatures of components matched, and how well the angles between components matched. The outcome of matching is determined by all three parameters. Experiments are used to determine what degree of similarity on each of the criteria we need to successfully match a given feature.

4 Database of components

A database of matchable components is needed to compare the components found in a sketch with. Since the larger part of the aim is to classify the overall sketch, our database must do more than identify individual components - we must concern ourselves with the relations between those components that lead us to build up the 'big picture'.

As section 5 describes, 'agents' are responsible for applying the above matching test on the components in the sketch, so each database entry must contain all the information that a agent might require for its search to be successful.

Often, a given component we want to match as a general case will have many different incarnations, which are completely incompatible with the matching algorithm. The database must therefore be hierarchical in nature - with sub-classes of features for recognising using low-level matches and parent classes that simplify relationships between agents.

Certain database entries (those at the very top level, for example) are designed such that an agent constructed from it can be satisfied by diverse range of possible sketches. For these agents no components-based models are required whatsoever, since the range would be too great. Only the layout of other agents within the system affects the judged success of these agents.

Each database entry includes some of the following:

– a model, for the purposes of matching components at the low-level;
– a parent, from which it can inherit rules from the more general case that still apply;
– neighbour relationships: rules to define what features are to expected in the vicinity of, and what constitutes a 'correct' layout relationship between features.
– alternatives, for the purposes of introducing different agents.

5 Agent control

We use agents to reduce the search space of features that could be part of a sketch by using relationships between parts already identified or partially identified. Examining conflicts and complementary features as we continue the recognition process allows us to reject certain branches of the space of features and order our search more appropriately. It is the fact that only relevant calls to the curve matching routine, and the agent's contextual comparisons are made that reduces the combinatorial explosion experienced with the 'brute force' approach of matching each component with each feature in the database.

Knowledge-based solutions to image understanding problems are relatively scarce [3] but have been explored [9, 4]. Utilising a similar approach to SIGMA [9], we attempt to improve over time our interpretation of an 'image' by allowing collaborative agents brought into play by the identification of certain components to search the available data for additional components that are expected to be present.

We define an 'arena' for our agent system to exist in. The potential features extracted from the graph are added to the arena 'floor', where all agents have access to them.

Roaming the arena are several agents, each tasked individually with finding a specific high-level feature. An agent, in this context, is an independent but communicative 'sub-program' endowed with its own objective and set of skills. By striving to reach its own goals it aids the whole process of identifying sketches. The agents 'wander' the floor exerting an influence on, and being influenced by the other agents in the image, searching for features that their grammar compliant word matches.

Agents are aided in their task by their own sense of achievement and restlessness - a measure of their confidence in having found their individual goal. The mobile agents gradually become more and more restless over time if they are unsuccessful in their task - leading them to become less careful about steering clear of other agents that it might otherwise consider inappropriate to consort with, and take more risks regarding unusual positioning within the image. The more restless an agent is, the less influence it has over other agents, whereas if an agent is satisfied with where it is other agents will consider its input more reputable.

An *agent's confidence* at its current location is a combination of three factors; a measure of how near it is to the feature it is trying to find (c_{target}), and a measure of how compatible its current position is with other agents with respect to the local (c_{local}) and global (c_{global}) rules. These values are computed using the two 'sensors': the target sensor and agent sensor.

The *target sensor* searches the vicinity of the agent (the part of the arena floor around the agent's location) for patterns that match its target specifications. This means running matching tests provided by this agent against fragments found around the agent. Output from the matching sensor directly effects the confidence level that agent advertises, and therefore its subsequent motion.

The *agent sensor* is responsible for interacting with the other agents that are active within the system. This allows an implicit co-operation between agents - one agent knows it should (typically) be 'above' a second agent within the image, so it is 'pushed' that way, and its confidence level increases to reflect that it believes it is in the correct area of the image - at least with respect to the former agent. Two agents can also be repulsed or attracted, depending upon issues regarding their respective targets (insisting on their closeness, for example) and the agent with the least confidence will be most affected.

An agent contains 'rules' regarding its relations with other agents in the image. These rules are split into two sets: local and global. A local rule applies to agents which are in the vicinity of the given agent. A global rule applies to the entire system of agents. Such rules could be that one feature must remain below another, or that one feature must remain horizontally between two others etc.

To compute the exact value of the agent confidence, the system is currently using the following formula. Let $\mathcal{F}$ be the set of potential features, $\mathcal{A}$ the set of all agents, $A \in \mathcal{A}$, R the range within which agents and features are noticed by the agent A, $DIST(x1, x2)$ a function that gives the distance between x1 and x2 in the arena, $CLOSE(x1, x2)$ gives a normalised measure of the distance between x1 and x2 in the sketch, where higher is closer, $A.MODEL$ the feature the agent is looking for, $MATCH(x1, x2)$ is the probability that a potential feature $x1$ is an example of an abstract feature $x2$, $A.RULES$ the set of local rules belonging to A, $RULE(a1, a2)$ the result of an application of a rule to agents $a1$ and $a2$ (higher is a better compliance with the rule).

$c_{target}(A) = \frac{\Sigma_{x \in X}(CLOSE(x,A) \times MATCH(x,A.MODEL))}{|X|}$
where $X = \{x \in \mathcal{F} \text{ and } DIST(x, A) < R\}$ (the set of features visible to A).
$c_{local}(A) = \frac{\Sigma_{a \in X}(\Sigma_{RULE \in A.RULES}(CLOSE(a,A) \times RULE(a,A))/|A.RULES|)}{|X|}$
where $X = \{a \in \mathcal{A} \text{ and } DIST(a, A) < R\}$ (the set of agents visible to A).
$c_{global}(A)$ is computed similarly to $h_{local}(A)$ but with respect to the set of global rules associated with A.

The agent will move in the direction that causes an increase in the value of c_{target}, c_{local} and c_{global} if possible. If one value increases radically whilst the others do not, it is an indication that perhaps it has found a false success.

Only a certain amount of agents are allowed into the image at any one time. Their introduction into the system of active agents follows a period of queueing, waiting to be 'voted' on by the agents that are already active within the image. A voting cycle is set up, and on each iteration of it, an active agent requests an agent that would help satisfy it, that is currently not active. The queue of inactive agents is ordered by the number of votes each agent has received and when a space in the image becomes available, the highest voted agent is released from the queue. Such a 'space' would be made after an agent is rejected from the image. This occurs if an agent's satisfaction drops too low. The dissatisfied agent is removed from the image and added to the back of the queue of waiting agents.

6 Experimental results

We demonstrate some of the results from testing of the low-level, component matching technique below. For this purpose, a user was asked to draw several primitive 'fish-tail' (figure 2) and 'dog-paws' (figure 3), which were matched against a small library of alternative components using the technique described in section 3.

As is apparent each of the matching tests produces different results. The three tests complement each other, as each is more or less applicable in differing circumstances. For example, using the length-based test several 'dog-paws' are mis-identified as 'fish-tails' simply because the ratio of lengths of each section of the component are likely to be the similar in this case.

The 'dog-paw' test demonstrates that the perpendicular distance test (curvature) is advantageous for models with obvious curves, as their are no incorrect matches. Models with no curves will not benefit from this test.

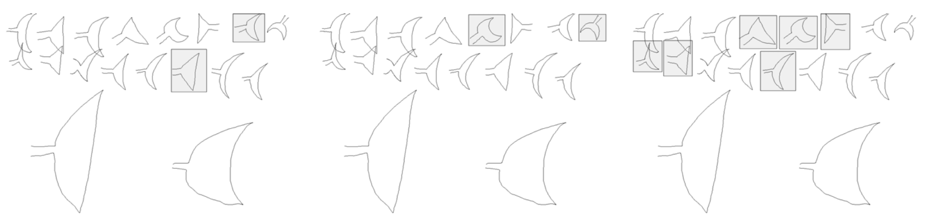

Fig. 2. Highlighted components were matched to the wrong model using the length *(top-left)*, Perpendicular distance *(top-right)* and bearing *(bottom)* match tests respectively for the 'fish-tails' test

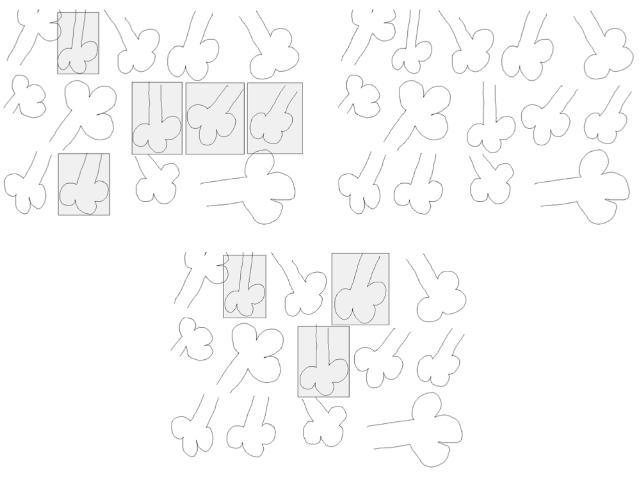

Fig. 3. Highlighted components were matched to the wrong model using the length *(top-left)*, Perpendicular distance *(top-right)* and bearing *(bottom)* match tests respectively, for the 'paws' test

Preliminary tests using the agents resulted in some promising results, an example is shown in figure 4 - a hastily drawn sharp-toothed fish. The appropriate agents, given a small match database and simple ruleset are introduced and they quickly find their valid component match, aided by their own interaction, as described above. Sketches do not have to be oriented specifically - the agents should arrange themselves regardless of the orientation of the overall sketch.

In the image, the black lines comprise the sketch, the dark grey lines represent strong potential matches that have been ignored and the faint grey lines show weak or where there is no match between agent and feature in the image. The agent's location is denoted by the word that describes its target. The image shows the agents after they have settled. Although the 'side-fin' and 'top-fin' agents have correctly found their targets, it is evident from the

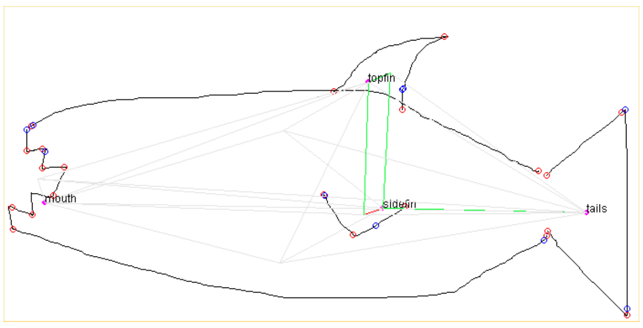

Fig. 4. Appropriate agents working on an example sketch

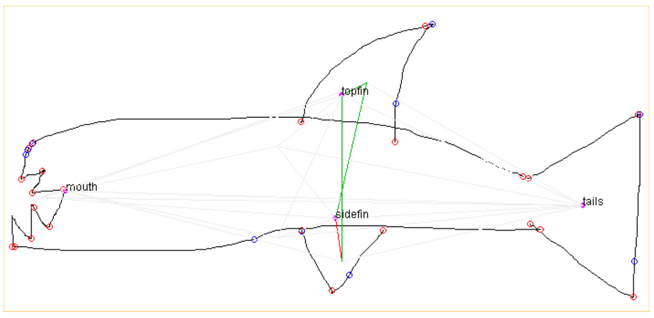

Fig. 5. A feature is in an unexpected location

strongly coloured lines leading from each agent to their other possible feature matches that they each might have settled on the other's target, if based on sketch data only.

Figure 5 shows an example where a feature has appeared in an unexpected location. This demonstrates the interplay between agents - although the 'side-fin' agent has identified the fin on the bottom of the fish it is kept from settling there by rules regarding the location of *side-fins*. The relevant rule states that *side-fins* 'prefer' to be in a line with tails and mouths - both of which have matched comfortably. A decision now needs to be taken as to whether to reject this agent and replace it with one from the queue, or whether to relax the problematic rule and allow the agent to settle.

Figure 6 shows how the location of a missing feature can be inferred using the agent interaction - the mouth agent has moved to the approximate location of the undrawn mouth of the fish using only rules regarding the locations of the other agents - specifically that it must preferably be in line with tail and sidefin, and be far away from the tail.

7 Related Work

We utilise a similar approach to SIGMA [9], by allowing collaborative agents brought into play by the identification of certain components to search the available data for additional components that are expected to be present.

An interesting comparison could be drawn between our system and the *SketchIT* [12] system. However, that system relies on having a limited model database of individually drawn schematic representations (mechanical engineering components, electrical components, UML components, etc) and uses knowledge of the *function* of the system to aid in the recognition process.

Some research has been done on interpreting freehand sketches [12, 8, 7, 2, 1] all of which requires a small restricted 'visual grammar' to operate or recognise simple, one-stroke geometric shapes. Our system segments completely before attempting recognition,

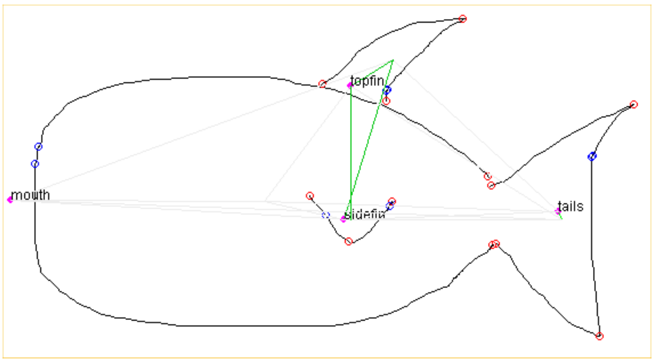

Fig. 6. A feature's location is inferred by an agent

not requiring the sketcher to draw strokes in a particular order and the recognition solution discussed should scale-up easily.

8 Conclusions and future work

We have described an approach to classifying hand drawings of animals using agents which move around a sketch trying to find 'their' feature (an eye, or a tail) and attracting and repulsing other agents. A prototype version of the system (parsing a sketch into components, matching components to features in a database, small hierarchical database, agent control mechanism) has been implemented and tested on small samples of data. The experiments testing the matching implementations demonstrate the need for a mechanism to combine the three simple match tests in a manner appropriate to the model being matched against. Early results of using agents are encouraging.

The subject of our future work is to find out whether the system scales when applied to a large collection of data. Much work is also needed in experimentally adjusting optimal matching parameters for various features as the database of features is being extended.

References

1. J. Arvo and K. Novins. Fluid sketches: Continuous recognition and morphing of simple hand-drawn shapes.
2. C. Calhoun, T. Stahovich, T. Kurtoglu, and L. B. Kara. Recognizing multi-stroke symbols. Technical report, 2002.
3. D. Crevier and R. Lepage. Knowledge-based image understanding systems: A survey. *Computer Vision and Image Understanding*, 67(2):161–185, 1997.
4. B.A. Draper, R. Collins, A. Brolio, A. Hansen, and E. Riseman. The schema system. *The International Journal of Computer Vision*, 2:209–250, 1989.
5. I. Guyon, L. Schomaker, Plamondonm R., M. Liberman, and S. Janet. Unipen project of on-line data exchange and recognizer benchmarks. 1994.
6. B. Hayes-Roth. A blackboard architecture for control. *Artificial Intelligence*, 26:251–321, 1985.
7. J. A. Landay and B. A. Myers. Interactive sketching for the early stages of user interface design. In *Human Factors in Computing Systems*, 1995.
8. J. A. Landay and B. A Myers. Sketching interfaces: Toward more human interface design. *IEEE Computer*, 34(3):56–64, 2001.
9. T. Matsuyama and V. Hwang. Sigma: A knowledge-based aerial image understanding system. *Plenum*, 1990.
10. H. P. Nii. Blackboard systems: The blackboard model of problem solving and the evolution of blackboard architectures. *AI Magazine*, 7(2):38–53, 1986.
11. S. Russel and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, 1995.
12. T. F. Stahovich. *SketchIT: A Sketch Interpretation Tool for Conceptual Mechanical Design*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, 1995.