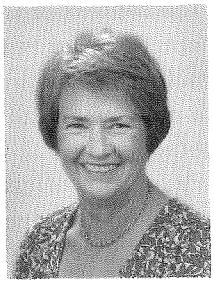
Intelligent Mind-map



Beryl PLIMMER
University of Auckland, New Zealand
beryl@cs.auckland.ac.nz
http://www.cs.auckland.ac.nz/research/hci/mind-maps.html

1. Project Goal

Intelligent support for informal documents on the tablet PC is hampered by inadequate recognition algorithms and ink reflow techniques. This project addressed both of these problems within the context of mind-maps. Mind-maps are especially attractive as an exemplar of an intelligent ink project as they require a combination of ink support techniques that are applicable to a wide range of other ink document domains. Furthermore the cognitive efficacy of the resulting tool can be evaluated against existing computer-based tools and paper equivalents lending weight to the growing body of evidence that computer-based sketching holds benefits beyond paper and current computer tools.

Mind-maps are a well know, particularly useful, informal technique for problem solving and organization of ideas. While there is a range of computer based tools for mind mapping such as Mind Manager (http://www.mindjet.com/) the quintessential unconstrained drawing space that a tablet PC affords is not intelligently supported. For this project this necessitates better ink recognition and reflow algorithms than are currently available.

Our prototype tablet PC based mind-mapping tool is designed to support pedagogically-oriented brainstorming. The scenario of use for this project is an educational application of mind-maps: planning a report. Imagine a student planning a report: they start with the main subject in the centre of the mind map and as ideas occur to them add branches and sub-branches, in no particular sequence. When satisfied with the mind-map they convert it into a linear tree structure to commence the detailed report writing. Each branch of the mind map becomes a section and each sub-branch a sub-section.

To provide such an intelligent tool the following functionality is required:

- Basic ink support (drawing, erasing etc) and file persistence.
- Ink grouping and dividing (text/drawing)
- Text and drawing recognition
- Document structural analysis

There is a natural semantic split between drawing and writing ink which humans achieve effortlessly – however it has proved difficult to achieve acceptable results on computers. Identifying ink features to discriminate between drawing and writing strokes is the subject of a current project being undertaken by Plimmer. The features discovered in this project using rigorous statistical analysis have reduced mis-recognition rates considerably. Our tests on sketched diagram components show that the Microsoft divider is strongly biased towards text: it correctly classifies all writing strokes, but classifies 75% of drawing strokes as writing. The new feature set we have developed has a 10.8% error rate for writing and 8.8% error rate for drawing resulting in overall error rate of 11% [1]. Our objective was to leverage our novel feature recognition based approach in the mind mapping project reported here and extend it with mind-map-specific features to further improve recognition. These extensions, in addition to permitting us to develop a functional mind-map tool, provide us insight into domain specific specialization of our generic recognition algorithms.

2. Technical Breakthrough

In the context of this project we have explored the interaction and recognition requirements for semi-structured hand-drawn diagrams, using mind maps as a motivating exemplar.

Interaction

Providing effective and intuitive interaction was one of the principal objectives of this project. Consider the mind-map in figure 1. As the related work section on sketching has become very busy the user may decide to move one of the branches down a level. One question we grappled with is 'how should the interaction work?' We implemented three types of selection: stroke-by-stroke, the user lassoes the strokes they want to select, with this selection there is no intelligent reflow; node, by clicking a node (contained or uncontained) all the ink of that node is selected, the node moves together and connectors are reflowed to the new node position; branch, this selects a branch, its sub-branches and associated nodes, the entire branch is move together and reconnected to the tree depending on where the user releases it. The initial indications from our project are that all three types of moving support are useful, yet the complex interaction model is hard for users to intuitively grasp.

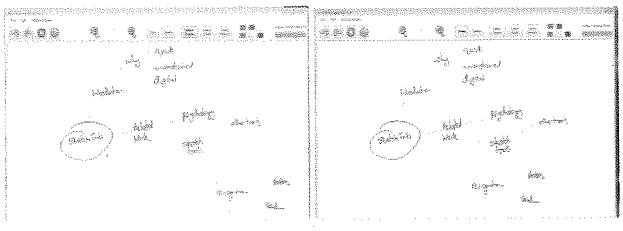


Figure 1. Mind-map a) where related work branch has become too deep b) the recognition branch has been moved in a level

We have thus found the interaction requirements for a mind-map tool are not straightforward, despite the conceptual simplicity of mind-maps. Creating a mind-map is simple however with hand-drawn input we have found that typical tablet PC screens are too small for a substantive mind-map without zooming or scrolling. While such techniques are technically simple they detract from the naturalness of the interaction experience and raise issues as to how additional ink is rendered on a zoomed canvas. A digital whiteboard interface solves the space problem, however the current generation of digital white boards have significant interaction issues in regards to parallax errors. One of the inherent disadvantages of digital ink is that people write larger than the typed equivalent, thus the digital ink product takes more screen real-estate than its widget based equivalent. Following on from our experiences in this project, there is scope for further research into how, from a user's perspective, to best deal with ink recorded at different levels of zooming.

Recognition

The recognition engine in the MM tool supplemented our divider [1] with a set of heuristically derived context rules. One of the most important of these is that although we aim to recognize an ink stroke immediately it is entered (eager recognition), the divider can make more accurate decisions after the following stroke has been drawn and when the full diagram is present. The recognition engine thus parses each stroke at least twice, once immediately it is entered and again following the entry of a subsequent stroke.

A new set of context based ink features were also defined for this project. There are three classes of strokes for mind-maps of the form shown in figure 1, circles (that indicate nodes), connectors and text. It is particularly difficult to algorithmically differentiate some connectors from text letters such as 'l'. To improve the recognition we altered the feature at the root of the divider tree from bounding box width to length of the longest side – this improved the error rate for approximately vertical connectors but resulted in more vertical letter strokes been classified as connectors. We then explored a number of ink grouping techniques to identify mis-classified letter strokes and at the same time identify unconstrained nodes. Letters are generally formed as a part of a word, by grouping the ink strokes accurately we can assume that if some of the ink in the group is letters, it will all be letters. Based on this premise we developed a set of heuristics to group and if necessary reclassify writing strokes. Consider the example in figure 2, as the new letter 'd' is added to the text block the upper and lower bounds of the node box are used to check for horizontal alignment and the right bounds for vertical alignment. Margins are added to the bounding box to allow for the first ascender or descender of a word (that is taller letters such as 'h' or those that descend below the base line such as 'g').

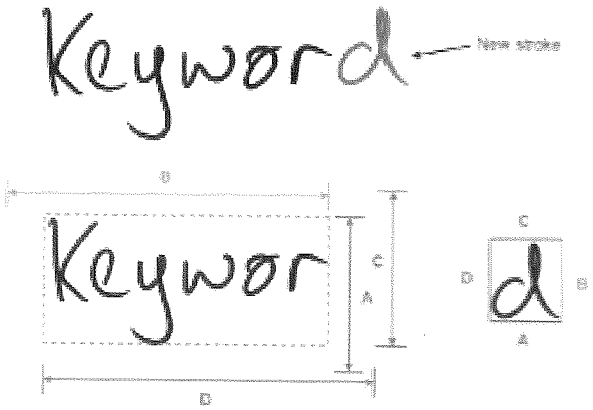


Figure 2. Spatial relationships between existing letters and a new letter added to a word

The node also aids in the reclassification of misclassified text strokes. Figure 3 show how an 'l' that has been classified as a connector can be detected as being in or close to a text box. We can be fairly confident when the entire stroke is within the bounding box (a), however sometimes the letter is partially inside (b) or close to a bounding box (c) one could construct a mind map where the connector has a similar spatial relationship. Heuristically we decided that if 30% of the stroke is within the box it is reclassified as text, and that if a near vertical stroke at the end of a text box meets the usual horizontal alignment criteria it is added to the text.

While these heuristics have significantly improved the recognition rate to approximately 93% there are still too frequent instances of errors. These rules are now being incorporated into our parallel work on ink division and have allowed us to define a number of new context related features for the feature library. The software also groups close words that are not separated by a connector to form multi-word nodes.

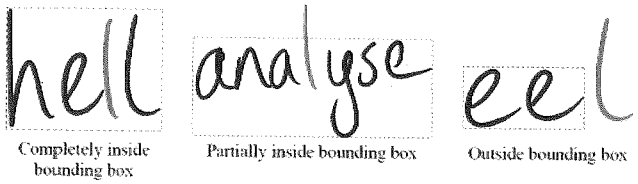


Figure 3. Examples of letters that need context attributes to distinguish them from connectors

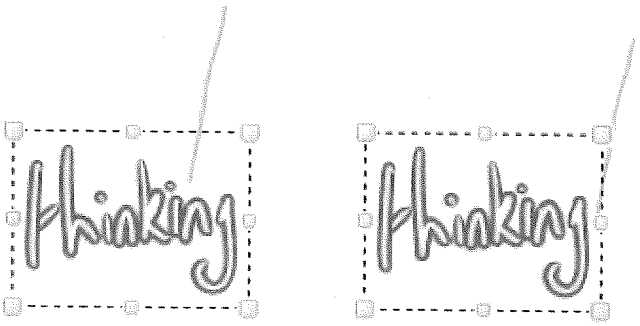


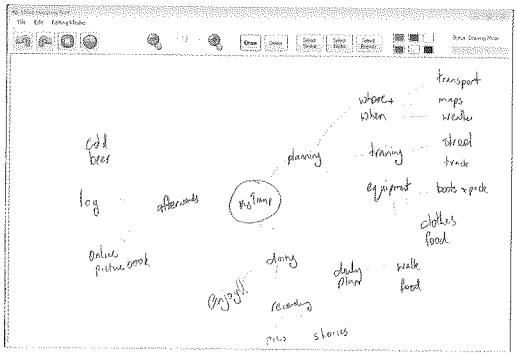
Figure 4. Examples of connectors that have similar context attributes to from the letters in figure 3

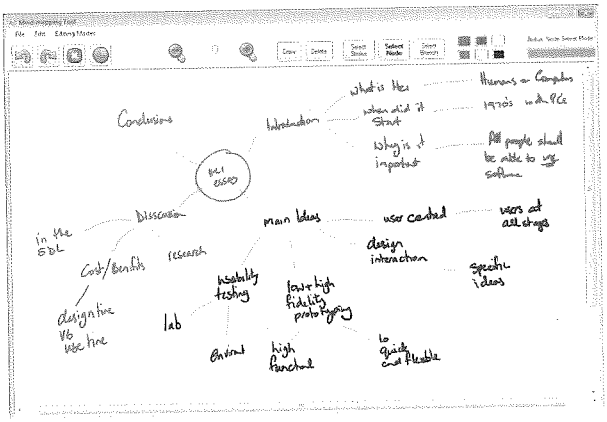
A small user study that we conducted with the tool suggests that such interfaces have a high enjoyment factor. Participants ranked the mind-map creation task very strongly the 'overall ranking for this experience' rated on a 10 point scale mean 8.75 (sd .88). However recognition errors impact the node detection and this in turn affects the stability of the editing support. Based on our experience with the mind-maps and a parallel project for hand-drawn graphs [2] we are currently building a robust software architecture that provides a framework for node-and-edge diagrams. This framework is component based to allow different recognition engines and layout algorithms (ideas explored in the graph project) to be plugged in.

(1) Patel, R., B. Plimmer, J. Grundy, and R. Ihaka, Ink Features for Diagram Recognition, in SBIM. 2007, IEEE: California.

(2) Reid, P., F. Hallett-Hook, B. Plimmer, and H. Purchase. Applying Layout Algorithms to Hand-drawn Graphs. In OzCHI 2007: Entertaining User Interfaces 2007. Adelaide: ACM.

3. Innovative Applications





4. Publications

[1] Chik, V., B. Plimmer, and J. Hosking. "Intelligent Mind-mapping". *in OzCHI 2007: Entertaining User Interfaces 2007. P187-190,Adelaide: ACM. (With acknowledgement to the grant of Microsoft Research Asia Mobile Computing in Education Theme)*

[2] Blagojevic, R., P. Schmieder, and B. Plimmer. "Towards a Toolkit for the Development and Evaluation of Sketch Recognition Techniques". *in Sketch Recognition Workshop, IUI 09. 2009. Florida. (With acknowledgement to the grant of Microsoft Research Asia Mobile Computing in Education Theme)*

[3] Chik V, "Intelligent Mind-mapping", *MEng Thesis, 2007 University of Auckland, available from*

http://www.cs.auckland.ac.nz/~beryl/publications/vincent_chik_mind_map_thesis.pdf

Dividing Text and Drawing Ink

Beryl PLIMMER
University of Auckland, New Zealand
beryl@cs.auckland.ac.nz
www.cs.auckland.ac.nz/research/hci/

1. Project Goal

Inked documents, for instance a sketched diagram, annotations on an article or to-do list, generally consist of both writing and drawing elements. From a recognition perspective the writing and drawing are fundamentally different. Writing strokes are joined together to create letters, words, phrases and sentences: and the writing has an abstract meaning that is not directly related to the ink shapes. In contrast drawing ink is generally in the form of ideograms or pictograms where the meaning is much more directly related to the ink shapes. Because of this fundamental difference it is important to be able to accurately separate writing and drawing ink. This is a very challenging problem. So difficult that in fact most sketch tools do not attempt it, resorting to keyboard input of words or very constrained writing input. Clear, well researched methods of separating writing and drawing would be of great benefit for ink recognition and the pen-input research community and enable effective electronic pen-based solutions for a much wider range of domains.

2. Technical Breakthrough

We have expanded our data collector (Blagojevic, Plimmer et al. 2008) into a toolkit for the development and evaluation of sketch recognition algorithms (Blagojevic, Schiemder et al. 2009). In addition to the data collection, single stroke labeling and dataset generation functions of the first prototype, our new platform includes a group labelling function, the addition of 60 stroke features to our library, and a framework to integrate existing recognisers easily into the tool for running comparative evaluations. We have integrated six existing representative recognizers; these recognizers are written in a range of languages including C#, C++ and Java. We are currently considering what constitutes a fair comparative test of recognisers given their differing characteristics and constraints. We are also working on adding more options to the testing framework to help us to support this.

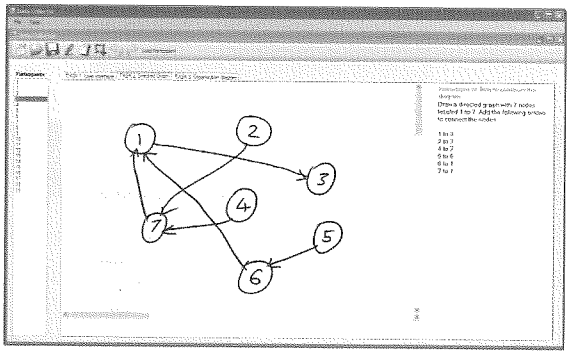


Figure 1. Data Collector interface

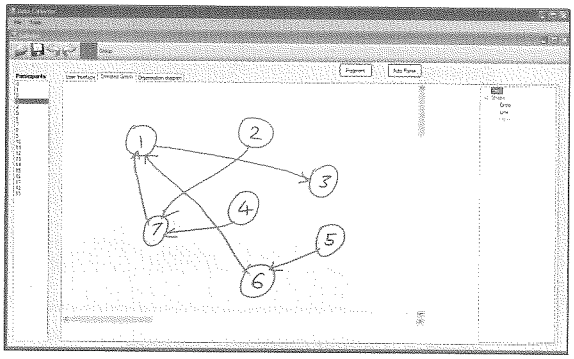


Figure 2. Labelling interface

3. Innovative Applications

Our toolkit is currently being used by our own research group. We have compiled a labeled dataset of sketched diagrams. Data was collected from 20 participants, with 3 diagrams each, totaling 7261 strokes. The data has been labeled with single stroke and group labels. Using our stroke feature library we have measured 100 features for each stroke in our dataset to use for analysis; the feature data file was constructed using our automatic dataset generator function.

We are using this feature data file and our toolkit to explore possible recognition algorithms to use for the divider problem. Currently we are investigating the use of data mining tools such as Weka (Witten and Frank 2005) to assist with this. Our plan is to automate the process by bridging between our toolkit and Weka. By automating many of the processes required for the development of recognition algorithms we seek to significantly minimize the time and effort required by researchers to improve on existing recognition techniques.

This is an open source project and we encourage other researchers to use and contribute to this project. It is available for download from *www.cs.auckland.ac.nz/research/hci.*

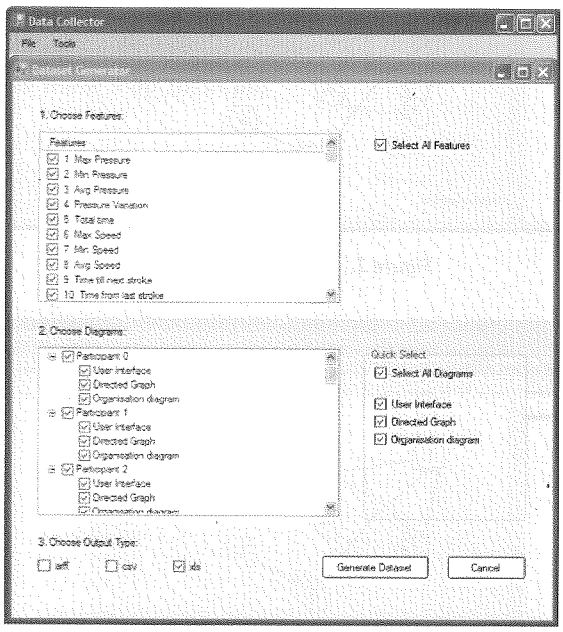


Figure 3. Dataset Generator interface

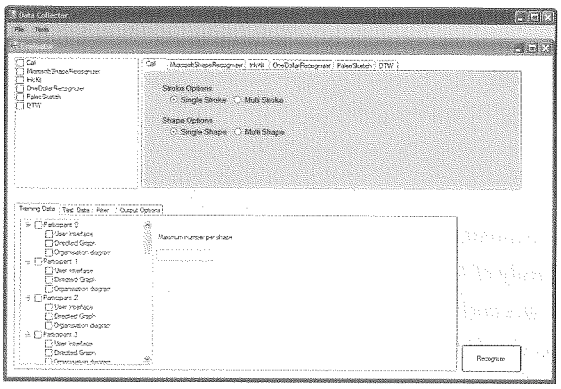


Figure 4. Evaluation interface

4. Publications

[1] Rachel Blagojevic, Beryl Plimmer, John Grundy, and Yong Wang, "A Data Collection Tool for Sketched Diagrams", *Proceedings of Sketch Based Interfaces and Modeling (SBIM'08), 2008, p 73-80*

[2] Rachel Blagojevic, Beryl Plimmer, John Grundy, and Yong Wang, "Development of Techniques for Sketched Diagram Recognition", *Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing Graduate Consortium, 2008, p 258-259*

[3] Rachel Blagojevic,Paul Schiemder,Beryl Plimmer, "Towards a Toolkit for the Development and Evaluation of Sketch Recognition Techniques", *Proceedings of Intelligent User Interfaces (IUI'09) Sketch Recognition Workshop, 2009, accepted in press,(With acknowledgement to the grant of Microsoft Research Asia Mobile Computing in Education Theme)*

[4] Rachel Blagojevic, "Development of Techniques for Sketched Diagram Recognition", *Australasian Artificial Intelligence (AI'08) Graduate Consortium, 2008.*