Structuralizing Freeform Notes by Implicit Sketch Understanding

Yang Li Zhiwei Guan Hongan Wang Guozhong Dai Xiangshi Ren*

Intelligence Engineering Lab.
Institute of Software, CAS
P.O.Box 8718, Beijing, China
+86-10-6254-0434
{ly, gzw}@iel_mail.iscas.ac.cn
guozhong@admin.iscas.ac.cn

*Department of Information Systems Engineering
Kochi University of Technology
Tosayamada-cho, Kochi 782-8502, Japan
+81-887-57-2209
ren.xiangshi@kochi-tech.ac.jp

Abstract

People are accustomed to capture important events and ideas by sketching notes on papers. Pen and paper offer people great freedom and naturalness to perform these activities. During note sketching, people generally don't pay much attention to the organization of notes, instead they use implicit spatial relationships and concise informal organizing symbols to structuralize notes. We conducted a study to discover how people sketch and organize their notes. The structuralized notes are easy to be maintained, manipulated and reused. Based on our observations, we designed algorithms to structuralize freeform notes into a consistent hierarchical structure. Both implicit spatial parsing and gesture-based structuralizing are enabled. With the supports of these algorithms, we built a system SketchPoint notebook as a tool that allows a user to sketch various notes in a freeform manner. It conducts lazy note structuralizing implicitly. The structuralized notes of SketchPoint enable structure-semantics based ink manipulations that bring much efficiency to users.

Introduction

Sketching notes by pen and paper is a convenient way to perform experience capturing, which is an important period to accumulate information for further knowledge production. People usually sketch notes including words, graphs, special symbols or meaningless doodles. They rarely pay much attention to the organizations of notes because the information in this period used to be premature and unsystematized. The miscellaneous notes are difficult to be maintained, manipulated and reused later. An effective structuralization of these free notes is important for further manipulations.

To acquire the information of how people sketch their notes, we conducted a study which showed that people generally capture the outline of information instead of the complete content, and people use both a set of concise, informal symbols to rapidly organize their notes and the implicit spatial relationships to indicate the structures of notes. The study also showed that miscellaneous notes on the paper often bring troubles to people to efficiently find their wanted information. Based on these observations, we designed algorithms to structuralize notes into a hierarchical structure. With supports of these algorithms, we developed a system named SketchPoint notebook (In fact, SketchPoint notebook is a part of the SketchPoint system that allows user to perform note-taking as well as sketch presentations), as shown in Figure 1, which allows users to sketch notes in a freeform manner and structuralizes the notes implicitly. By structuralizing notes, SketchPoint enables a user to perform structure-semantics based manipulations. For example, people may interactively collapse and expand notes in a structural way. People also can select and delete a piece of related notes efficiently. By utilizing the metaphor of zoomable user interfaces (Bederson and Hollan 1994), SketchPoint enables a user to easily navigate the notebook by zooming.

To achieve a low cognitive load for users, SketchPoint notebook parses the structures of freeform sketches internally and keeps original appearances of ink except giving some instructive feedbacks to users. SketchPoint conducts lazy and incremental structure parsing of ink to decrease cognitive load of users and improve the accuracy of structuralizing.

In this paper, we firstly give a survey of related researches, which influence the design of SketchPoint. There are obvious differences between SketchPoint and previous systems. Then, the analyses of note structures are presented which were based on our informal user study. We

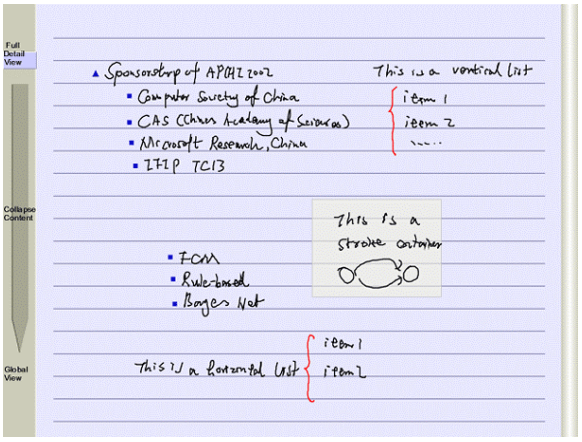


Figure 1: The SketchPoint notebook

summarized and abstracted a hierarchical structure to organize notes. Based on the analyses, we describe the algorithms for note structuralizing in detail. We continue by briefly introducing functionalities and interaction techniques of SketchPoint. We conclude this paper with future work and conclusion.

Related Work

Experience capturing is an important period for people to accumulate information for knowledge production. Most commercial software tools focus on formal knowledge production (Igarashi 2000). For example, MS Word assists people to customize well-formatted document.

However, they are not suitable for informal experience capturing. Several studies have been done under whiteboard metaphor for informal workgroup meetings (Moran, Melle and Chiu 1998) (Pedersen et al. 1993) or experience capturing in office work (Mynatt et al. 1999). By contrast, we designed SketchPoint based on the notebook metaphor to enable users to capture daily experiences by sketching freeform notes.

As many studies of pen interaction, SketchPoint also greatly employs sketch-based interaction techniques, which have been used widely in various areas, e.g., in GUI design (Landay and Myers 2001), web site design (Lin et al. 2000) and other research areas (Hearst et al. 1998). It allows users to sketch notes in a freeform manner, which means that anything can go anywhere without constraint. SketchPoint keeps the sketched ink unrecognized and processes it internally as (Hearst et al. 1998) (Lin et al. 2000) (Pedersen et al. 1993) (Wilcox, Schilit and Sawhney 1997) (Davis et al. 1999).

Dynomite (Wilcox, Schilit and Sawhney 1997) is a portable electronic notebook for the capture and retrieval of handwritten and audio notes, which employs spatial and temporal distance to dynamically group digital ink and audio (Chiu and Wilcox 1998). By contrast, instead of only using the spatial and temporal distance of strokes, SketchPoint employs high-level structure semantics into ink grouping.

In Shipman's work (Shipman, Marshall and Moran 1995), an investigation was conducted to find out some fundamental and typical spatial structures of information. In Igarashi's work (Igarashi, Matsuoka, and Masui 1995), based on Shipman's work, two typical structures, i.e. cluster and list, were supported. They also built a system for parsing card organizations, which used a link model to parse the spatial relationships of cards into cluster and list organizations. In contrast to these previous researches, we focus on the implicit structure of notes which note clusters have dynamically changeable boundary. Both inter-stroke and inter-cluster spatial relationships are necessary to be parsed to perform note structuralizing. As a result, these attributes of notes require quite different structure parsing.

As an important interaction technique, the metaphor of zoomable user interface (Bederson and Hollan 1994) was used by various systems to organize large information into a limited presentation space, e.g. in (Lin et al. 2000). SketchPoint notebook also employs this technique to present a collapsible, hierarchical and structural view of notes to users, and allows a user to smoothly navigate in the notebook.

Analysis of Note Structures

We conducted an investigation to find out how people write and organize their notes when they use pen and paper. Most interviewees write notes to fulfill brainstorm, to capture important events, to make schedules, or to create reminder prompts. Generally, they can't pay much attention to the careful organization of these notes and they have to write quickly and randomly. Instead they use particular spatial layouts of notes to indicate the structure of notes. As shown in Figure 2, the strokes that are close to each other unite as a cluster. The distance and the indention of stroke clusters

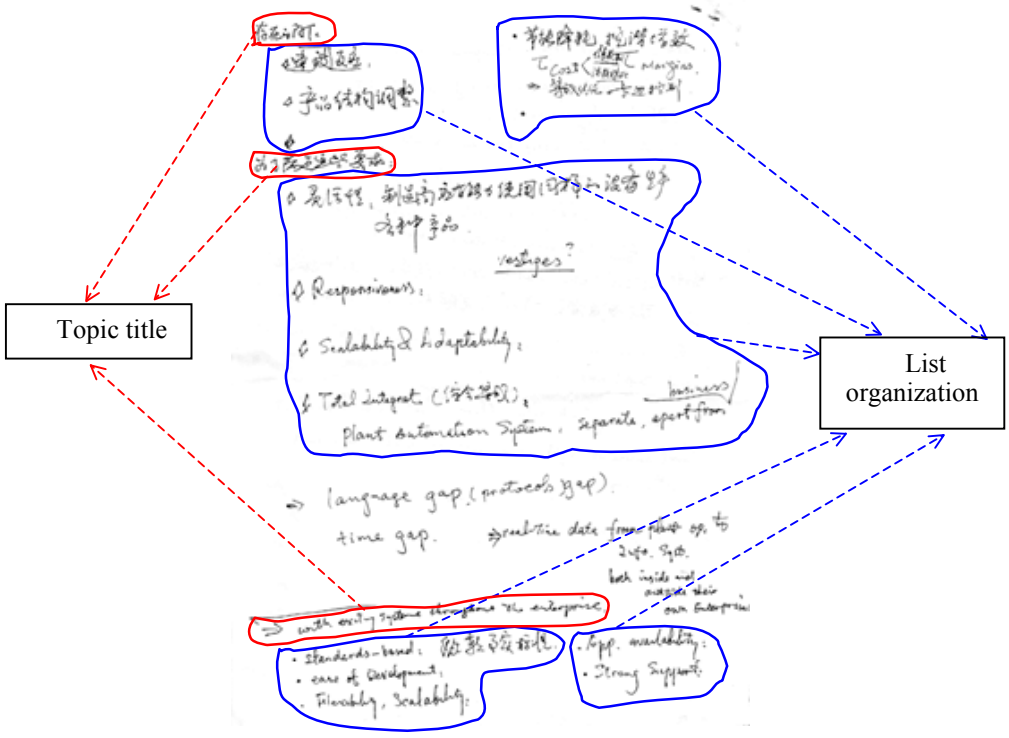


Figure 2: A Sample of using implicit spatial relationships to group notes

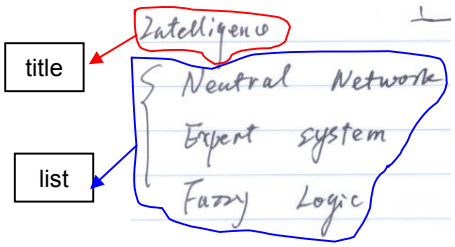


Figure 3: Group a list of notes by a large bracket

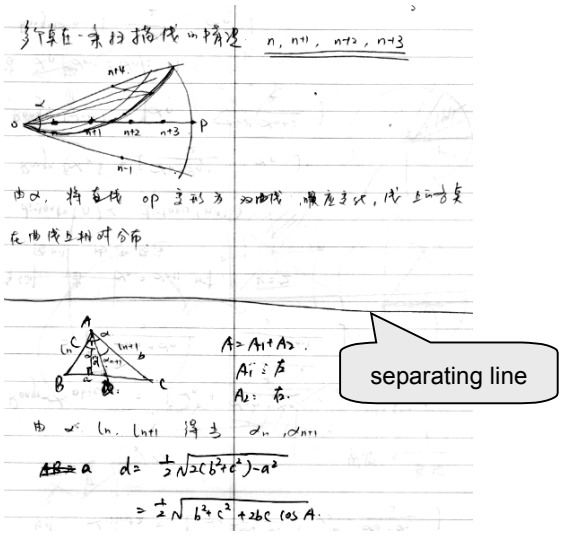


Figure 4: A long straight horizontal line to separate different sections of notes

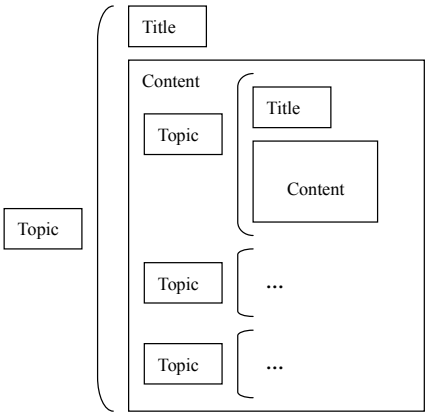


Figure 5: The logic structure of title-content

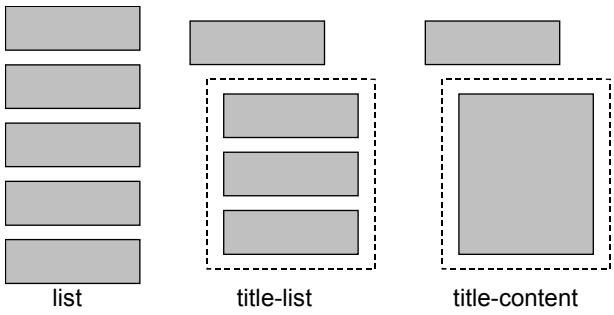


Figure 6: Typical organizations of notes

indicate list organization and title-content or –list organizations.

Besides grouping notes based on implicit spatial relationships, people also employ a set of informal and concise symbols to organize their notes. As shown in Figure 3, a large bracket presents a list. In figure 4, a long straight horizontal line stands for a separation between different sections of notes. And an arrow indicates a relationship between two blocks of notes or further explanations.

Implicit spatial relationships and informal organizing symbols are two major ways of people to organize their notes, which need little attention of users.

The investigation showed that, in note taking, people usually capture the outline rather than the complete content. Logically, all notes are distributed into different sections, which each section is special for a certain task or composed under a special situation. Each section usually consists of several topics. A topic is generally organized as a title-content structure, which the title is usually followed by the content, although different people may use different layouts (vertical or horizontal manners) and/or organizing symbols to denote this structure. The content of a title-content structure could be a list of notes, graphs, tables or any other type of stroke clusters. Particularly, it could be a list of subtopics recursively. In this way, people can organize their information hierarchically. The logic structure is shown in Figure 5.

Based on above observations, we abstracted and formalized the structure of a notebook by CFG (Context-Free Grammar) (Lewis and Papadimitriou 1999) as following specifications:

Notebook⟶Section·Notebook|Section

Section⟶Topic·Section|ε

Topic⟶Title·Content

Title⟶*strokeblock*

Content⟶*strokeblock*|List|ε

List⟶Topic·List|Topic

A *strokeblock* is a set of strokes, which satisfy the distance limit and density requirement. We used the algorithm introduced in (Chiu and Wilcox 1998) to compute the stroke distance in a *strokeblock*. The *strokeblock* could also be created through marking a region as a stroke container by a block-making gesture. And the strokes in this region don't need to meet the distance and density requirements, which is useful for graphs and other unusual notes. The specification unifies various notes of a notebook into an inverted tree structure, which the leaf nodes are strokes and the root node is notebook (This specification for logic structure of notes doesn't discriminate between words

or graphs. It focuses on the logic structures of notes, which are expressed by organizing symbols and particular spatial layouts).

To group notes into above logic framework, three typical layouts, as shown in Figure 6, to indicate the implicit structures of notes, are greatly used in note sketching. "list" structure is characterized as a set of left-aligned stroke clusters with neighboring clusters close to each other (a large bracket symbol is also used to define a list). "title-list" structure and "title-content" structure are denoted by the indention relationships of stroke clusters. In our research, we presently focus on these typical structures.

Algorithms of Note Structuralizing

We designed algorithms for structuralizing notes. The overview of structuralizing process is described in Figure 7. The first step is to cluster strokes into *strokeblock*. A stroke is produced by dragging the pen on the tablet (in our system, we used PL550, i.e. a LCD tablet). Once a stroke is generated, clustering process is invoked immediately to group the stroke into the closest *strokeblock* with a certain requirement or create a new *strokeblock* for this stroke. Fragment clearing up is conducted as a complement to further the stroke clustering. Based on the results of stroke clustering, i.e. *strokeblocks*, high-level structuralizing is performed to analyze the spatial relationships of these *strokeblocks*. Firstly, exceptional analysis is conducted which is special for those structuralizing gestures. These gestures are used to build a list, denote the relationships of *strokeblocks* or make a note section. Finally, implicit spatial structure analysis is performed which organize notes based on spatial relationships of *strokeblocks*, such as indention and distances. For each time of note structuralizing, there might be some notes left un-structuralized and they will be parsed again next time. Consequently, our algorithms structuralize notes in an incremental way.

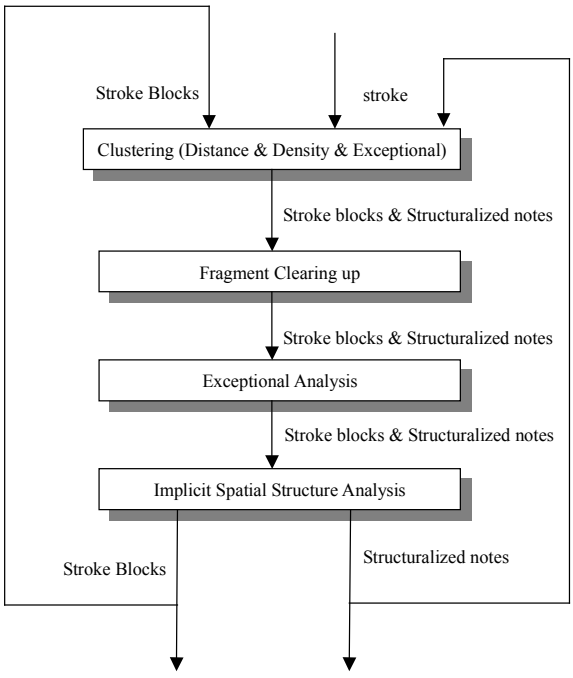


Figure 7: Overview of note structuralizing

Stroke Clustering

Strokes are the primitive elements of notes in our system. However, generally, a single isolated stroke doesn't fully express the information. Instead, a stroke cluster named *strokeblock* in our system, which is composed by a set of strokes, is the fundamental element to keep the information. As a result, a better clustering of strokes is very important to the whole process of note structuralizing.

In our research, the algorithm of stroke clustering can be characterized as followings (stroke ***x*** is the target stroke to be distributed to *strokeblock*):

1. Is the stroke ***x*** is an exceptional stroke (e.g., a structuralizing gesture)? If yes, create an exceptional *strokeblock* for further parsing. Then go to step 6.
2. Is the stroke ***x*** in a stroke container (created by block-making gesture)? If yes, go to step 5.
3. Compute the minimum distance between the stroke ***x*** and all existing *strokeblocks*.
4. If the minimum distance is less than DIS-REQ (a constant of maximum distance of strokes in a *strokeblock*), go to step 5. If not, create a new *strokeblock* containing ***x*** and then go to step 6.
5. Add the stroke to the according *strokeblock*.
6. End the stroke clustering.

The distance between a stroke x and a *strokeblock* Y is defined as followings:

$$\text{Distance}_{(x,Y)} = \text{Min}(f(x, s)), \ \forall s \in Y$$
$$Y = \{s_i \mid 0<i<n,\ s_i \text{ is a stroke of a } \textit{strokeblock } Y\}$$

The function $f(x, s)$ is to compute the distance between two strokes, which is designed based on the algorithm of (Chiu and Wilcox 1998). A brief introduction of this algorithm is shown in Figure 8 (a).

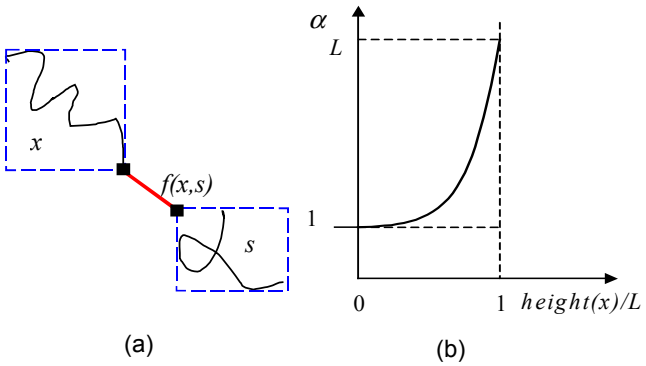


Figure 8: functions for stroke clustering

Presently, we use a simple function as followings to judge whether a stroke $\boldsymbol{x}$ is exceptional.

$$\text{Factor} = \text{width}(x) + \alpha\,(\,\text{height}(x) - M\,) \quad \text{if height}(x) > M$$
$$\text{Factor} = \text{width}(x) \quad \text{otherwise}$$

If the "Factor" is larger than a constant value, we take the stroke as an exceptional stroke. The function width(x) is to compute the width of the stroke bounding box and height(x) is for the height. "M" is a constant. The value of α is computed from the function of Figure 8 (b) where "L" is an adaptive constant.

Sometimes, some fragmentary clusters are generated after stroke clustering. In order to achieve more accurate structure parsing of notes, fragment clearing up is conducted before *strokeblock* structuralizing. It merges those small *strokeblocks* into the neighboring *strokeblocks*. The examples of stroke clustering are shown as Figure 13 in appendix section.

Structure Parsing of Strokeblocks

After stroke clustering, all strokes are categorized into *strokeblocks*. Although stroke clustering is conducted once each stroke is produced, the higher-level structuralizing, *strokeblock* structure parsing, is invoked when it is necessary. For example, when a user intends to select or conduct other manipulations on the un-structuralized notes, this process will be started.

The first step of this process is exceptional analysis. Taking the affordances of those organizing symbols used by people, we generalized and abstracted a set of symbols as structuralizing gestures. For example, a long straight horizontal line is used to separate sections. A large bracket is to group a list of notes. The exceptional analysis doesn't absolutely rely on the results of gesture recognition but employs some contextual factors. For example, if there are more than two *strokeblock*s intersecting with the sensitive region of a large bracket and meeting a certain requirement, as shown in Figure 9, this bracket symbol will be considered as a list-making gesture to group these *strokeblocks* as a list.

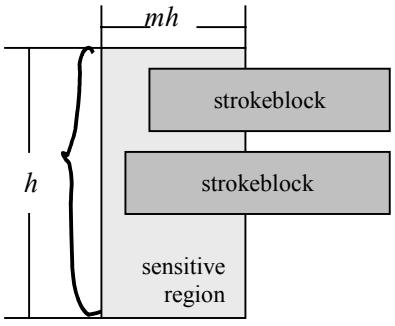


Figure 9: list-making gesture. "m" is an adaptive factor.

After exceptional parsing, all un-structuralized *strokeblocks* will be sorted and parsed in top-down and left-right manner. To each *strokeblock*, all other strokeblocks in a certain region around it will be selected out. The distance computation of *strokeblocks* is similar to the computation of stroke distance. The spatial relationship parsing of neighboring *strokeblocks* is briefly shown in Figure 10. In the neighboring *strokeblocks*, the indention of a *strokeblock* to others is used to tell whether it should be taken as a title or an item of a list.

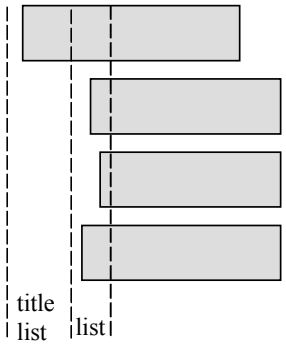


Figure 10: Parsing of list and title-list structures

The results of note structuralizing include the structuralized notes and un-structuralized *strokeblocks*. When the structure parsing is activated next time, these un-structuralized *strokeblocks* will be parsed again under new circumstances where new *strokeblocks* are created, some notes are deleted or structuralizing gestures are used. So our algorithm structuralizes notes in an incremental and interactive way, which continuously complete and refine its parsing with more and more information provided by users. Examples are shown in Figure 15 in appendix section.

SketchPoint Notebook

We applied the algorithms of previous sections in our system SketchPoint to support free note structuralizing. SketchPoint notebook, as shown in Figure 1, is a prototype system built in Java 2, which facilitates people in note taking. It allows a user to sketch notes in freeform manner and structuralizes notes (ink) implicitly and interactively into the hierarchical structure of the previous sections.

Structure-Semantics based Manipulation

SketchPoint organizes notes into a hierarchical tree-like structure. In this way, SketchPoint can bring several advantages to users.

By enhancing ink grouping with the structure semantics of notes, users can more efficiently select and manipulate the desired notes instead of a single stroke. For example, selecting the title of a topic will cause the whole topic including its content and attached sub-level topics to be selected. Similarly, erasing the title of a topic (by scribbling gesture) will remove the whole topic. The title-content structure maintains the correlation of notes, which reduces the effort of user needed to discern the relationships between these miscellaneous notes.

SketchPoint enables a user to select notes by the lightweight interaction techniques, i.e. "continuously taping" the pen on the target notes. The time length of the "hold" or the number of taps in the "continuous tap" method determines the factor of selection expansion. The selection expansion within a *strokeblock* depends on the distance of the stroke clusters. However, outside the *strokeblock*, selection is expanded according to the structure of the notes, i.e. the selection will be expanded bottom-up from the leaf to the root node in the structure tree. In this way, the user can rapidly select notes at any level from a single stroke to the whole notebook. In addition, SketchPoint provides a set of basic functionalities to enable a user to make easy modifications.

Context Sensitive Interaction

There is no explicit mode switch during interaction. Context-aware techniques (e.g., spatial constraint relationship) are used to confine the recognition into a well-defined space to acquire greater reliability.

As mentioned before, "tap" pen on the target is a convenient way to select or confirm etc. However, this interpretation sometimes conflicts with "drawing a point". Here, we used a simple rule to interpret "tap" event as "drawing a point" or "select". We assumed that user generally select a *strokeblock* after this *strokeblock* is finished. If the temporal distance between a strokeblock and the "tap" event is large enough, this "tap" event is considered as a "selecting" action, otherwise as an action of "draw a point".

SketchPoint also allows user to draw scribbling gesture to remove the whole *strokeblock* or draw a horizontal line to cross out part notes of a *strokeblock*. Besides using the results of gesture recognitions, they are analyzed by the relationships with the surrounding notes.

Zoomable and Collapsible Views of Notes

The interviewees complained that, in the later use of notes, it is not easy for them to find the wanted information and recognize the relationship between their notes because there are too many disorderly notes of different topics appearing on the paper. Moreover, after several modifications to the notes, the notes tend to be illegible.

Based on the hierarchical structure of notes, SketchPoint notebook allows a user to collapse and expand notes by zooming. By dragging the elevator of zoom slider (on the leftmost of screen in Figure 1) top-down (i.e. from "full detail view" to "Global view") user can collapse notes gradually while zooming. So by zooming the notes to the appropriate detail level, users can easily find the desired topic. By choosing the zoom center and zooming in or out, a user can efficiently and naturally navigate in the notebook.

There are two major collapsing styles according to different note layouts, as shown in Figure 11 (a list is a kind of content). To achieve the effect that the content is collapsed into the title, the zoom transform for content is accelerated in x-axis (for horizontal layout) or y-axis (for vertical layout) as shown in Figure 12. Screen captures are shown in Figure 14 in appendix section.

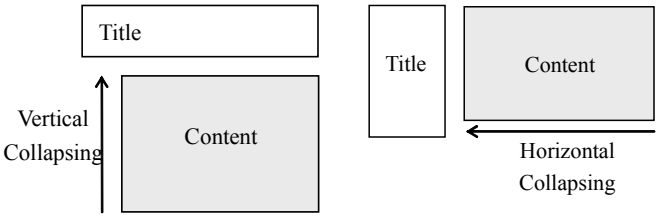


Figure 11: Collapsing styles for different layouts

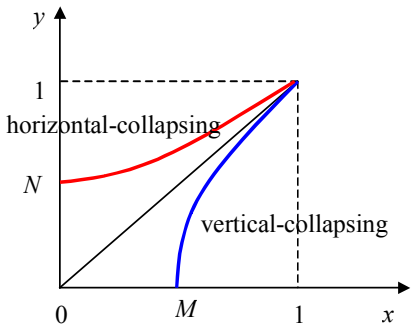


Figure 12: Zoom transform functions for horizontal and vertical layout

Future Work

There are many adjustable parameters involved in our algorithms. Presently, there is no online learning algorithm to training these parameters. They were fixed based on the sample analysis of our user study. To adapt our system to different users, we intend to add learning algorithms into our system to enable online parameter training.

Currently, we use zoomable user interface to enable user to navigate in note space. Although interactively collapsing and expanding notes bring some convenience in finding the desired notes, it is still not easy for people to manually find the target notes when there are lots of notes distributed in a large region. In our future work, we will add automatic note searching for efficient note retrieval, which helps people automatically find target notes. The underlying hierarchical structure of notes will benefit the automatic searching, which structure semantics will give helpful hints to search engine.

Conclusion

We analyzed the intrinsic structures of notes by a study. And based on the results of this study, we designed algorithms to structuralize notes from implicit spatial relationships of strokeblocks. With the supports of these algorithms, we built SketchPoint notebook to facilitate people in experience capturing by note taking. It allows a

user to sketch notes freely and structuralizes notes implicitly and incrementally. SketchPoint notebook groups ink based on structure-semantics of notes and users can benefit from it by freeform note sketching, context-aware interaction, structure-semantics based ink manipulations and the collapsible views of notes.

Acknowledgments. This research was supported by Grant No. 60033020 from National Natural Science Foundation of China and Microsoft Research China. We would like to thank anonymous reviewers for their comments to improve this paper.

Appendices.

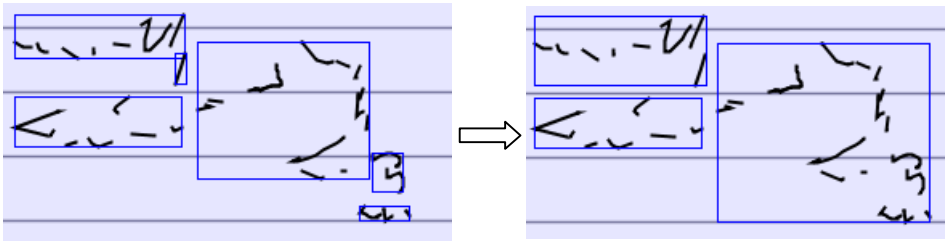
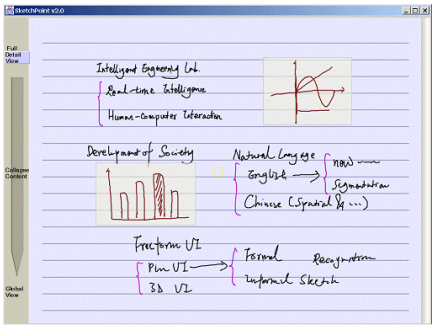
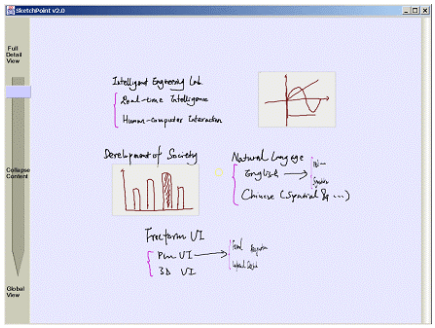


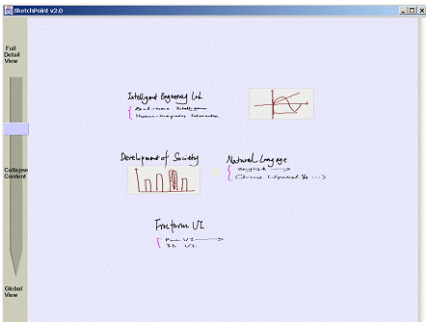
Figure 13: Fragment clearing up for stroke clustering



(a) Full detail view

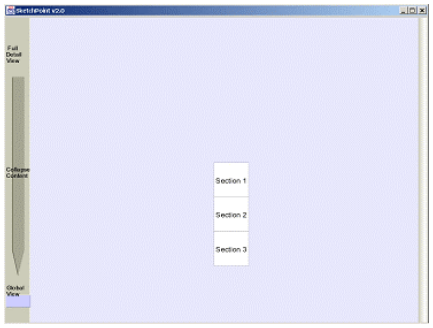


(b) The partially collapsed notes Ⅰ



(c) The partially collapsed notes Ⅱ

(d) The completely collapsed notes



(e) Section view

Figure 14: Zoom from full detail view (a), through (b), (c) and (d), to the section view (e) by dragging the elevator of on the leftmost slider in top-down manner

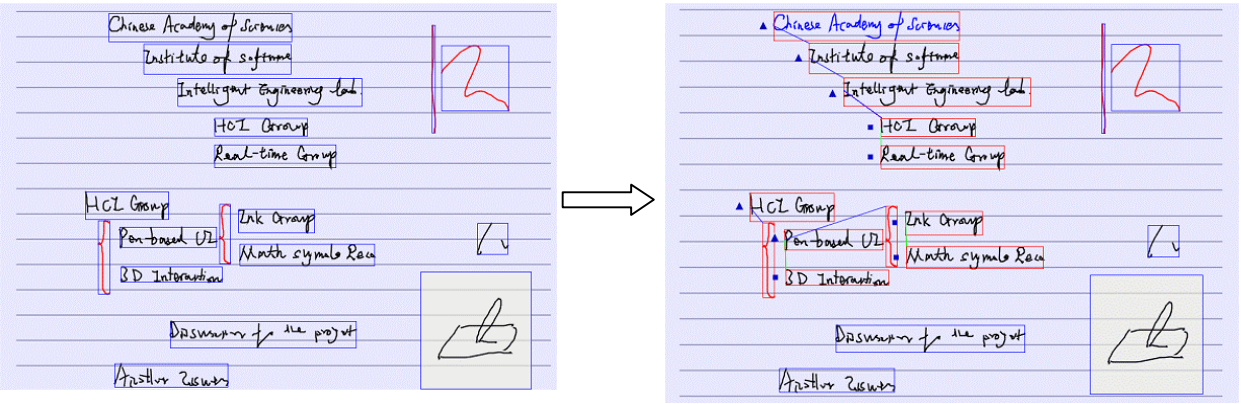


Figure 15: Examples of note structuralizing (in debug mode). In the right graph, the linkage lines and interactive tags are shown to indicate the results of structuralizing.

References.

Bederson, B.B. and Hollan, J.D. 1994. Pad++: A Zooming Graphical Interface for Exploring Alternate Interface Physics. In Proceedings of the ACM Symposium on User Interface Software and Technology: UIST'94, 17-26. Marina del Rey, CA. Nov.2-4.

Chiu, P. and Wilcox, L. 1998. A Dynamic Grouping Technique for Ink and Audio Notes. In Proceedings of the ACM Symposium on User Interface Software and Technology: UIST'98, 195-202. San Francisco, CA. Nov.6-8.

Davis, R.C., Landay, J.A., Chen, V., Huang, J., Lee R.B., Li, F., Lin, J., Morrey, C.B., Schleimer, B, Price, M.N. and Schilit B.N. 1999. NotePals: Lightweight Note Sharing by the Group, for the Group. In Proceedings of the ACM Conference on Human Factors in Computer Systems: CHI'99, 338-345. Pittsburgh, PA, May 15-20.

Hearst, M.A., Gross, M.D., Landay, J.A. and Stahovich, T.F. 1998. Sketching Intelligent Systems., IEEE Intelligent Systems, vol.13, no.3, *IEEE*, May-June, 10-19.

Igarashi T. 2000. Supportive Interfaces for Creative Visual Thinking, Collective Creativity Workshop, Nara Japan, May 7-8.

Igarashi T., Matsuoka, S. and Masui, T. 1995. Adaptive Recognition of Human-Organized Implicit Structures, Proceedings of Visual Languages '95, 258-266. September.

Landay, J.A. and Myers, B.A. 2001. "Sketching Interfaces: Toward More Human Interface Design." *IEEE Computer*, vol. 34, no.3, March, 56-64.

Lewis, H.R. and Papadimitriou, C.H. 1999. *Elements of the Theory of Computation*, Tsinghua University Publisher.

Lin, J., Newman, M.W., Hong, J.I. and Landay, J.A. 2000. DENIM: Finding a Tighter Fit Between Tools and Practice for Web Site Design. In CHI Letters: Human Factors in Computing Systems, CHI 2000, 2(1): 510-517.

Moran, T.P., Melle, W.V. and Chiu, P. 1998. Spatial Interpretation of Domain Objects Integrated into a Freeform Electronic Whiteboard. In Proceedings of the ACM Symposium on User Interface Software and Technology: UIST'98, 175-184. San Francisco, CA. Nov.6-8.

Mynatt, E.D., Edwards, W.K., LaMarca, A. and Igarashi, T. 1999. Flatland: New Dimensions in Office Whiteboards. In Proceedings of the ACM Conference on Human Factors in Computer Systems: CHI'99, 346-353. Pittsburgh, PA, May 15-20.

Nakagawa, M., Machii, K., Kato, N. and Souya, T. 1993. Lazy Recognition as a Principle of Pen Interfaces. In Proceedings of the ACM INTERCHI'93 Conference on Human Factors in Computing Systems -- Adjunct Proceedings, 89-90.

Pedersen, E.R., McCall, K., Moran, T.P. and Halasz, F.G. 1993. Tivoli: An Electronic Whiteboard for Informal Workgroup Meetings. In Proceedings of the ACM INTERCHI'93 Conference on Human Factors in Computing Systems, 391-398.

Shipman, F.M., Marshall, C.C., Moran,T.P. 1995. Finding and Using Implicit Structure in Human Organized Spatial Layouts of Information. In Proceedings of the ACM Conference on Human Factors in Computer Systems: CHI'95, 346-353. Denver, Colorado, May 7-11.

Wilcox, L.D., Schilit, B.N. and Sawhney, N.N. 1997. Dynomite: A Dynamically Organized Ink and Audio Notebook. In Proceedings of the ACM Conference on Human Factors in Computer Systems: CHI'97, 186-193. Atlanta, GA.