Combining Representations for Improved Sketch Recognition

by

Sonya J. Cates

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2009

Author .....
Department of Electrical Engineering and Computer Science
September 4, 2009

Certified by .....
Randall Davis
Professor
Thesis Supervisor

Accepted by .....
Professor Terry P. Orlando
Chairman, Department Committee on Graduate Theses

Combining Representations for Improved Sketch Recognition

by

Sonya J. Cates

Submitted to the Department of Electrical Engineering and Computer Science
on September 4, 2009, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Sketching is a common means of conveying, representing, and preserving information, and it has become a subject of research as a method for human-computer interaction, specifically in the area of computer-aided design. Digitally collected sketches contain both spatial and temporal information; additionally, they may contain a conceptual structure of shapes and subshapes. These multiple aspects suggest several ways of representing sketches, each with advantages and disadvantages for recognition. Most existing sketch recognitions systems are based on a single representation and do not use all available information. We propose combining several representations and systems as a way to improve recognition accuracy. This thesis presents two methods for combining recognition systems. The first improves recognition by improving segmentation, while the second seeks to predict how well systems will recognize a given domain or symbol and combine their outputs accordingly. We show that combining several recognition systems based on different representations can improve the accuracy of existing recognition methods.

Thesis Supervisor: Randall Davis
Title: Professor

Acknowledgments

First, I must thank my advisor Randy Davis, who always asks the right question and provides the right way of looking at a problem, and who may very well be the best advisor at MIT. I didn't know what I was doing when I first arrived at MIT and chose a group, but I certainly chose well.

I would like to thank my committee members Patrick Winston and Pawan Sinha for their valuable suggestions, insights, and questions, and Professor Winston in particular for sharing his renowned presentation skills.

I have been very fortunate not only in my advisors, but in my group mates as well. I have benefited greatly from the advice, feedback, and community the Design Rationale/Multimodal Understanding Group has provided over the years. Thank you in particular to Jacob Eisenstein and Tom Ouyang for being such great officemates and sounding boards, and to Mike Oltmans, Christine Alvarado, and Metin Sezgin for not only being great group mates but also for providing the work upon which this thesis is based. I am particularly indebted to Aaron Adler for being a great officemate and frequent lunch mate, and for his invaluable tech support.

Thank you to T!G, and to Anthony in particular, for solving every problem thrown at them, including but not limited to, excessive processor time demands, mice, leaky ceilings, odd smells and extreme temperatures.

I am grateful to many outside of CSAIL as well. My time spent with the cycling team will always be one of the best experiences of my life, and the women's team in particular has provided me with friendships that will last beyond MIT.

Thank you to my parents for their unwavering love and support (and for not asking too many questions like that pesky "When will you be finished?" one). Thank you to my family, both new and old, for their kindness, support and understanding.

Finally, thank you to my husband Bill, who has been wonderfully supportive and encouraging. I am grateful to have had him with me for the end of this journey, and look forward to the beginnings of many more.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Sketching is a critical first step in many design problems. Architects and engineers, for example, commonly make many rough sketches on the way to a design. Designers often make these initial sketches on paper and must later transfer their work to a computer for further development. Computer recognition of sketches could streamline this common process, but for many applications users require an error rate near zero before adopting a new technology. Until then they prefer to use predictable, if cumbersome methods. While there exists a diverse and advancing body of work on sketch recognition, none of the current systems have both the flexibility and accuracy required in a realistic design setting. The general goal of this work is to improve recognition accuracy and thereby advance sketch recognition applications.

The sketches that this work examines are created with a digital pen, which records both position and timing information, so the sketches have both spatial and temporal aspects. As sketches are also frequently conceived as compositional, i.e. made up of a hierarchical structure of shapes and subshapes (as, for example, a square is made up of lines), they also have a conceptual structure. This multi-faceted quality, i.e., the spatial, temporal, and conceptual aspects, allow a sketch to be thought of and represented in several ways. Later in this chapter we describe three types of representations corresponding to the spatial, temporal and conceptual views of a sketch. Most existing sketch recognition systems are based primarily on one of these representations and do not fully use all of the information contained in a sketch.

This thesis is based on the idea that these different representations may have different strengths and that the choice of representation can affect what recognition is feasible or easy. We show how to combine multiple representations as a way to improve recognition accuracy. We demonstrate improved recognition accuracy by combining existing recognition systems based on different representations and suggest a combined approach for the development of future recognition systems.

1.1 Why Recognize Sketches?

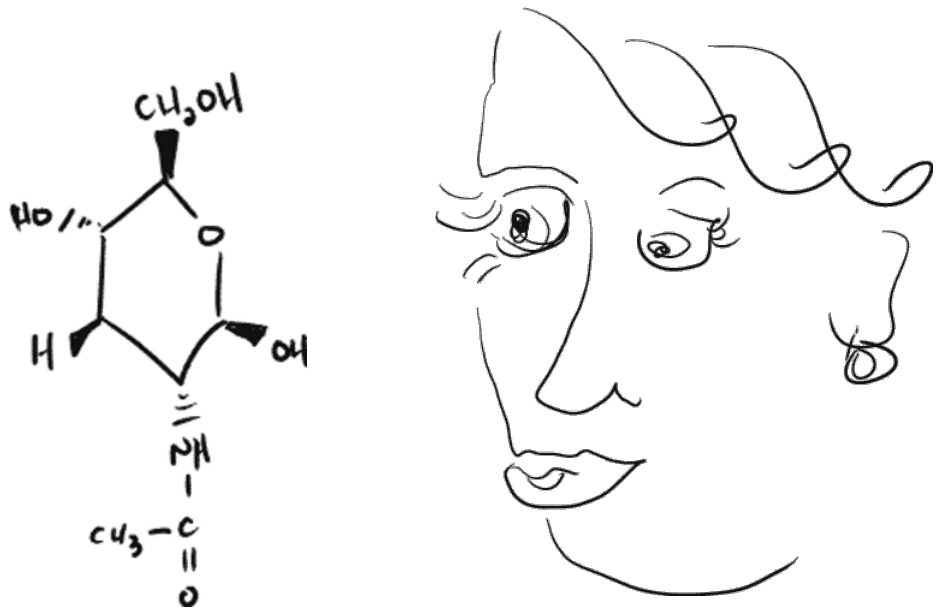
Consider a collaborative meeting of engineers, designers, scientists, or artists. Now take away the whiteboard, pens, and paper. Likely, you will be left with a roomful of unhappy and unproductive people making frantic and indecipherable gestures. Natural communication occurs in different forms, with sketches communicating some information most effectively, as for example, a complex diagram or map. This is true both in collaborative processes [64], and in the design process of an individual, because it extends the designer's memory and cognitive ability [46, 63, 32].

As computers become more powerful and more pervasive, we would like both to communicate with them in the same efficient ways that we communicate with people and to use them to facilitate design and collaboration. This desire for more natural human-computer interaction and for a more effective role for computers in the design process is one compelling motivation for the development of pen based input for computers. Recent work by Adler and Davis [1] and by Zamora and Eyjolfsdottir [70] demonstrates how a computer might aide in the design of mechanical devices and digital logic circuits; however, much remains to be done before a computer can recognize and understand unrestricted sketching.

Tablet computers are currently used commercially for taking notes and recognizing handwriting, but ultimate goals for sketch based interfaces presented by Davis [11] and Forbus et al. [16] include human-like understanding, reasoning, and participation. While these are distant goals, a clear intermediate step is recognizing what has been drawn, i.e. parsing and associating meaningful labels to a collection of pen strokes.

1.2 Nature of Sketches Considered in This Work

There are many different kinds of sketches. Those considered in this work are symbolic, in the sense that they are made up of symbols that have a mental association with a type of object or relationship in the world, but they need not resemble that which they represent [44]. Additionally, the symbols are standardized (we can define a lexicon) and are used to compose sketches by combining them according to rules (we can form a full or partial grammar). Figure 1-1(a) contains an example of a symbolic sketch. Element, compound, and bond symbols in this chemical diagram have defined meanings and relationships with each other. This work does not consider more artistic sketches, such as that in Figure 1-1(b), which do not have a defined lexicon or composition rules.



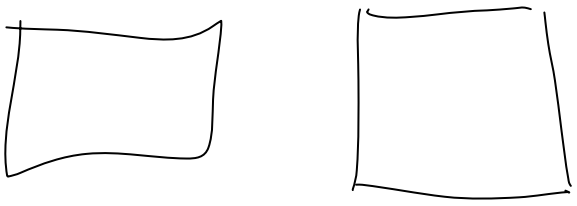
(a) Symbolic sketch.

(b) Nonsymbolic sketch.

Figure 1-1: Examples of symbolic and nonsymbolic sketches.

This work considers sketches drawn with a digital pen (in particular, all of the sketch data described below was captured using digitizing LCD tablets), rather than those drawn on paper then scanned and digitized. This distinction means that the data considered here contains time information: each point has spatial coordinates as well as an associated time stamp. This allows for the use of temporal properties of a sketch including stroke order, drawing speed, and the length of time between strokes, in addition to properties based on spatial distribution of ink.

The sketches are also minimally constrained, or unconstrained. In creating these sketches the user is allowed to draw freely, rather than required to draw symbols with a fixed number of strokes, or to signal the system after each symbol is completed. This means that the same symbol may be drawn with one or many strokes; a stroke may also belong entirely to one symbol or may be split among two or more. Figures 1-2 and 1-3 illustrate these two scenarios. Furthermore, the sketches are frequently messy or imprecise. For example, the sides of the rectangle in Figure 1-2(a) are curved rather than straight lines, while the corners of the resistor in Figure 1-3(b) are rounded rather than sharp.



(a) Rectangle drawn with one stroke.

(b) Rectangle drawn with four strokes.

Figure 1-2: Parts (a) and (b) show the same symbol drawn with one and four strokes.

(a) Resistor and two wires drawn with three stokes.

(b) Resistor and two wires drawn with one stroke.

Figure 1-3: Parts (a) and (b) show the same set of symbols drawn with three strokes and with one stroke, which must be divided among the symbols.

1.3 Meaning of Recognition

While other applications for digital pen input devices have been conceived and developed, this thesis is specifically concerned with sketch recognition. We define recognition in terms of its two necessary components: segmentation and classification. We define these processes separately for clarity; however, they may be performed either sequentially or simultaneously by a recognition system.

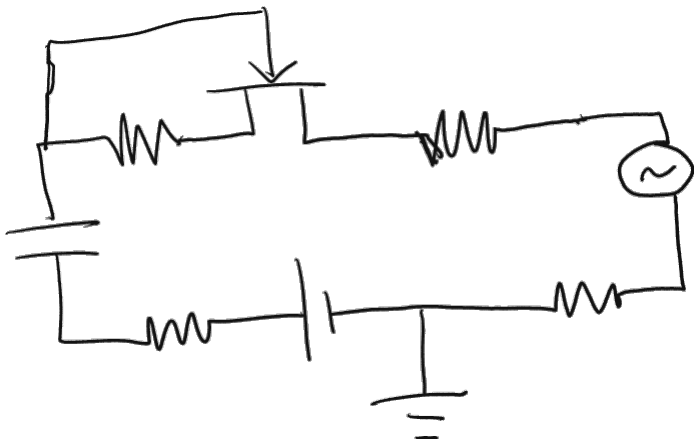


Figure 1-4: A sketch of a circuit diagram.

Segmentation of a sketch refers to the grouping of strokes, subparts of strokes, or pixels according to the domain symbols they comprise. Figure 1-5 illustrates the correct segmentation of the circuit diagram sketch in Figure 1-4 into groups corresponding to symbols for wires, resistors, grounds, capacitors, batteries, AC sources, and JFETs.

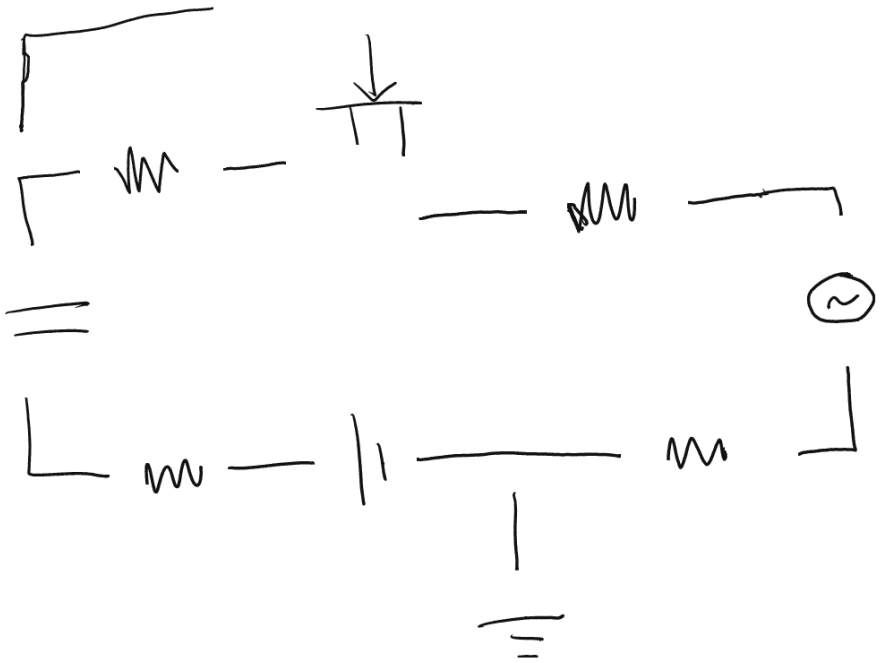


Figure 1-5: A correct segmentation of the sketch in 1-4. Symbols are spatially separated to indicate groupings of stroke segments.

We define classification as assigning labels to the resulting groupings. Figure 1-6 illustrates the assignment of labels to the segmented sketch in Figure 1-5.

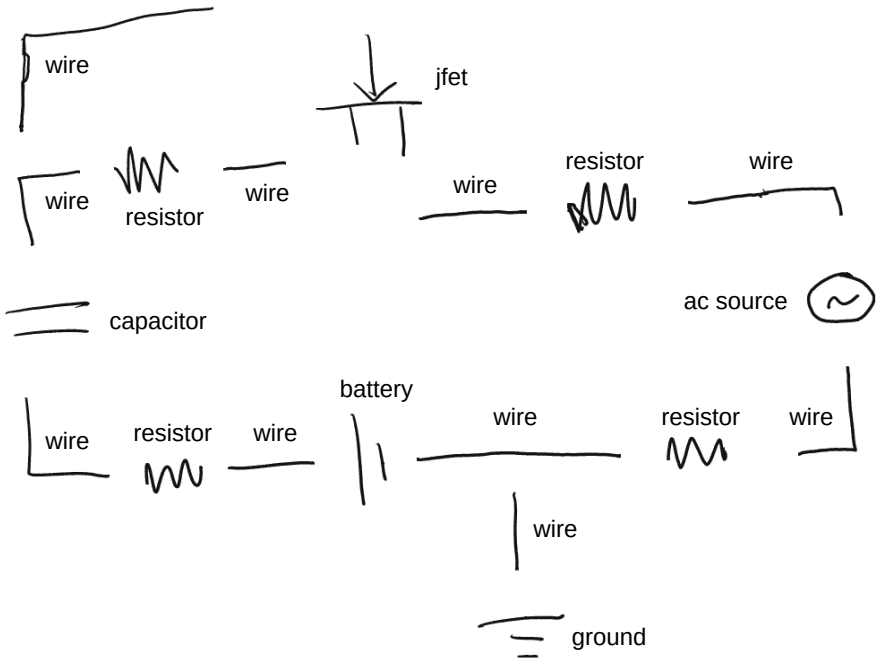


Figure 1-6: An example of classification of a segmented sketch. Symbols are spatially separated to indicate groupings of stroke segments.

1.4 Sketch Representations

We identify three primary aspects of sketches: the spatial, the temporal, and the conceptual. Each aspect provides a way of thinking about a sketch and a means to represent it, and each has advantages and drawbacks for recognition. Throughout this work we consider three representations of sketches corresponding to the three aspects of sketches.

1.4.1 Spatial

By the spatial aspect of a sketch, we mean literally what the sketch looks like: the areas of ink and absence of ink that we see when looking at the sketch on a screen or piece of paper. Obvious spatial representations are simply an array of pixels or a list of coordinates. This type of representation is appealing both because of its simplicity and because of the large existing body of work in the field of computer vision that uses similar representations.

1.4.2 Temporal

The temporal aspect of a sketch is based on the way the sketch was drawn, including drawing order, pauses, etc. We create the most basic temporal representation with a sequence of time-stamped pen positions. From that higher level abstractions may be created such as time-stamped Boolean observations corresponding to whether or not a drawing action occurred, velocity vectors, or other relevant features. The temporal aspect of sketching is an appealing basis for representation because it is a unique quality of online pen-based interaction; without the timing information, we have only a static image, as might be obtained by drawing on paper and scanning the result.

1.4.3 Conceptual

We define the conceptual view of a sketch as its geometric or symbolic contents and the configuration of the contents within the sketch. A conceptual representation indicates the sketch's geometric primitives, for example line segments and curves, and their spatial relationships, for example locations or whether or not two segments meet. We might alternatively list more complex geometric or symbolic objects as the sketch's contents, such as triangles or resistors. A conceptual representation is attractive because it reflects how the sketch may have been conceived by its author and facilitates high level inferences about what has been drawn.

1.4.4 Relationship to Existing Recognition Systems

It is unclear how to evaluate a representation directly, independent of an implemented system. Furthermore, a representation is interesting only in so far as it facilitates recognition (or other task). Therefore, we evaluate, compare, and combine recognition systems, rather than representations.

Given that we want to discuss systems rather than representations, it is useful to note that the three sketch representations that we have presented can be used to categorize sketch recognition systems as well, according to which representations are used. While most systems do not take purely one approach, many do strongly focus on one of the representations that we have described. Our work builds directly on three existing recognition systems, which are described below. We discuss other related systems in Chapter 5.

Spatial

The recognition method developed by Oltmans [40] performs recognition based on visual parts, also referred to as image patches. Shapes are classified by comparing the parts of a candidate symbol to a standard set of parts, resulting in a vector of measurements that represent the degree to which each standard part appears in the candidate symbol. Classifying this vector then results in a classification of a shape. Segmentation is performed by locating many candidate symbols and keeping only those with a high classification score.

Temporal

Sezgin and Davis [55] developed a recognition method that represents a sketch as a sequence of observations (strokes or substrokes). A Hidden Markov Model is trained for each symbol in a domain. A sketch may then be recognized by using dynamic programming to find a sequence of symbols that accounts for all of the strokes in a sketch and that maximizes the overall probability of the observations.

Conceptual

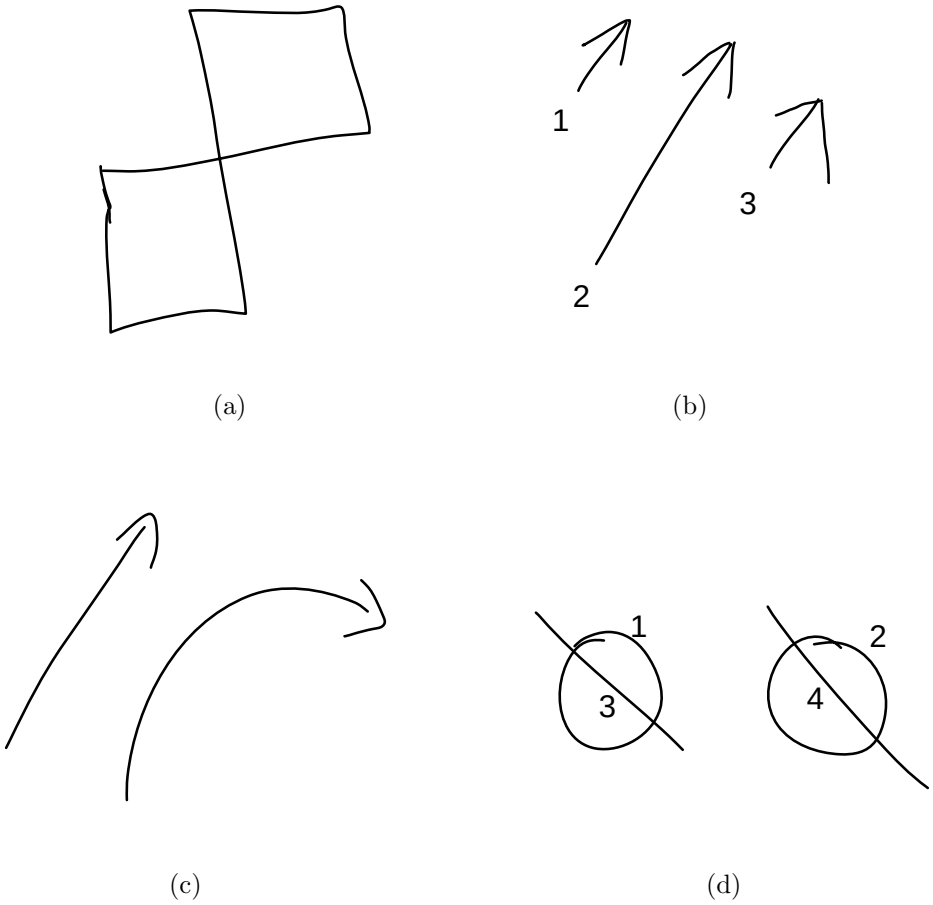
The SketchREAD recognition system developed by Alvarado [5] uses the shape description language LADDER [23] to describe shapes in a domain in terms of components and constraints between those components. Figure 1-7 illustrates an example of a LADDER description for a rectangle. A sketch is recognized by parsing strokes into possible interpretations. The system generates likely interpretations for groups of strokes, then corrects mistakes in the low level interpretations of components and constraints based on high level domain knowledge.

```
(define shape Rectangle
  (components
    (Line line1)
    (Line line2)
    (Line line3)
    (Line line4)
  )
  (constraints
    (touching line1line2)
    (touching line2 line3)
    (touching line3 line4)
    (touching line4 line1)
    (perpendicular line1 line2)
    (parallel line2 line4)
    (equal line1 line3)
      ...
  )
)
```

Figure 1-7: A LADDER shape description for a rectangle.

1.5 Why Use Multiple Representations?

In this section we provide some motivation for the use of multiple representations for sketch recognition. Figure 1-8 presents several simple shapes, the recognition of which is made easier or harder depending on the representation selected.



(a) (b)

(c) (d)

Figure 1-8: Simple shapes whose recognition may be helped or hindered by the choice of representations

Figure 1-8(a) contains two squares. These squares might be easily recognized with computer vision pattern matching techniques; however, recognizing these shapes with a conceptual representation could pose a problem. If the entire figure is drawn with one stroke, that stroke must be broken and divided between the two squares. Testing all such possible divisions may not be feasible (the time complexity of segmenting and grouping strokes grows exponentially with the number of strokes in a sketch), and it is difficult to define heuristics that are sufficiently thorough and accurate.

Figure 1-8(b) demonstrates the opposite case; here a spatial representation may make recognition difficult while a conceptual approach is clear. We identify both shapes 1 and 2 as arrows, though they are related by a non-affine transformation. However, another non-affine change made to shape 1 produces shape 3, which is something that we do not identify as an arrow. Specifying all possible transforms to shape 1 that will result in an arrow would be cumbersome with a purely spatial representation. However, recognizing shapes 1 and 2 as arrows without including shape 3 may be done simply with a conceptual definition that specifies an arrow as a line forming the shaft and two lines of roughly equal length forming the head.

The arrows in Figure 1-8(c) present another case where a spatial representation might be cumbersome. In a sketch these groups of strokes are likely to have the same interpretation, though they appear different, again as a consequence of a non-affine transformation. A temporal representation, however could be useful in determining this similarity, as the arrows would likely be drawn with the same temporal pattern. For example, one might consistently draw an arrow's shaft before drawing its head.

In Figure 1-8(d) the numbers correspond to drawing order, so the two circles were drawn first, followed by the two lines. This interspersing of the parts of different symbols can be problematic for a temporal approach since parts that are relevant to each other are not adjacent temporally. Thus segmenting the sketch on purely temporal grounds would be unsuccessful. However, this interspersing poses no problem visually because the parts that are relevant to each other are adjacent spatially.

These simple scenarios are representative of common phenomena in hand-drawn sketches. Employing an appropriate representation can greatly simplify the problem and improve the accuracy of recognition. However, the wide variety of phenomenon in hand-drawn sketches means that a domain or a sketch is unlikely to contain only elements that are ideally recognized with a single approach.

1.6 Method Overview

In this chapter we have introduced three ways of representing sketches and a recognition system based on each of them. The remainder of this thesis explores ways of combing these representations and recognition systems in order to improve recognition accuracy. Two combination methods are discussed.

The first method improves recognition by improving segmentation. While existing systems classify isolated symbols with a high degree of accuracy [43], localizing symbols within a sketch remains a difficult problem. Improving segmentation is a critical step towards improving recognition of whole sketches.

We present three methods for approximately segmenting sketches, one corresponding to each of the three representations described above. That is, we develop segmenters based on spatial, temporal, and conceptual representations. These segmenters are based on characteristics of human behavior and perception as they pertain to each representation, rather than domain knowledge. As a consequence they do not rely on knowing what symbols may be drawn. As one example, a long pause between two strokes is a good indication that the two strokes are part of different symbols.

We demonstrate that the resulting segmentations can be used improve recognition by processing sketches with our approximate segmentation methods, then recognizing the segmented pieces of the sketches with systems based on different representations than those used for the presegmentation. This two stage process enforces break points that are very likely to be symbol boundaries, thus preventing recognition errors made in one part of a sketch from propagating.

Our second combination method seeks to determine which representation/system is most likely to be correct for a given domain or symbol, and combine outputs accordingly. The method does not require running an actual recognition system and may be applied prospectively to decide what type of system or what approach to take for a new recognition problem.

We first propose simple complexity measures for each representation that predict recognizer performance on a domain. We then expand the idea of complexity to

the symbol level by defining symbol complexity in terms of confusability with other symbols in a domain. This symbol complexity, which may be computed without empirical data, is used to judge the credibility of each recognition system for each symbol it recognizes in a sketch. The three recognition systems may then be combined by comparing the likelihoods of their outputs or by training the systems only for those symbols for which they are judged to be highly credible.

1.7 Contributions

This thesis makes four primary contributions, listed here in the order of presentation.

- First, the thesis presents segmentation methods corresponding to each of the representations. We demonstrate that combining the segmentation methods of multiple representations yields a good approximate sketch segmentation without a need for domain knowledge.

- Second, we demonstrate that the segmentations described above can be used to improve recognition accuracy, by performing approximate segmentation as a preprocessing step.

- Third, this thesis proposes symbol confusability as a means of judging the credibility of a system. We demonstrate that combining recognition systems based on how likely they are to confuse symbols can improve recognition accuracy.

- Finally, we present representation specific measures to approximate confusability without the need to collect empirical data or build systems. We demonstrate that these measures can be used to effectively combine recognition systems.

1.8 Outline

This thesis is organized as follows. Chapter 2 describes the sketch data considered by this thesis, including the particular domains and data sets used in later chapters.

Chapter 3 presents methods for segmenting sketches corresponding to each of the three representations described in Section 1.4. We then describe how these segmentations may be combined and how the combination may be used to improve recognition. Chapter 4 proposes simple domain complexity measures, and then expands the idea of complexity to the symbol level by defining symbol confusability metrics. We then use these metrics to combine recognition systems in parallel. We end with chapters discussing related and future work (Chapters 5 and 6) and our contributions (Chapter 7).

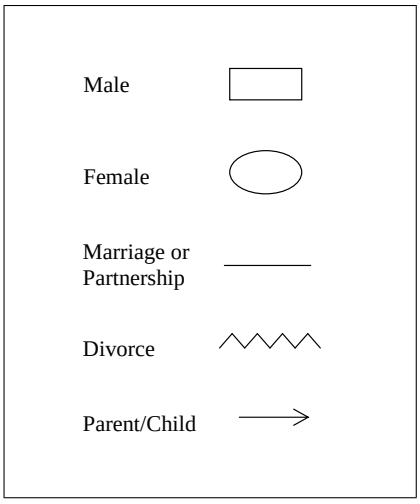
Chapter 2

Sketch Data

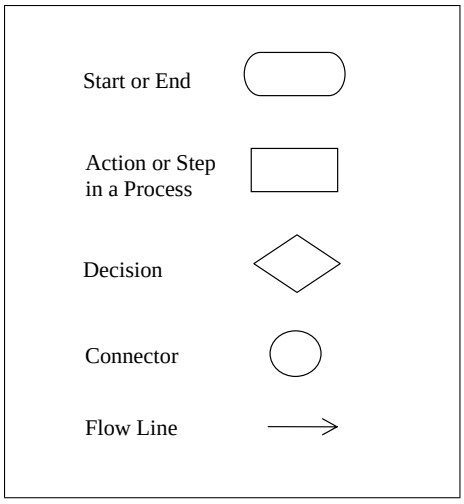
This chapter describes the sketch data considered by this thesis: the particular domains and data sets for which results are presented in later chapters.

2.1 Domains

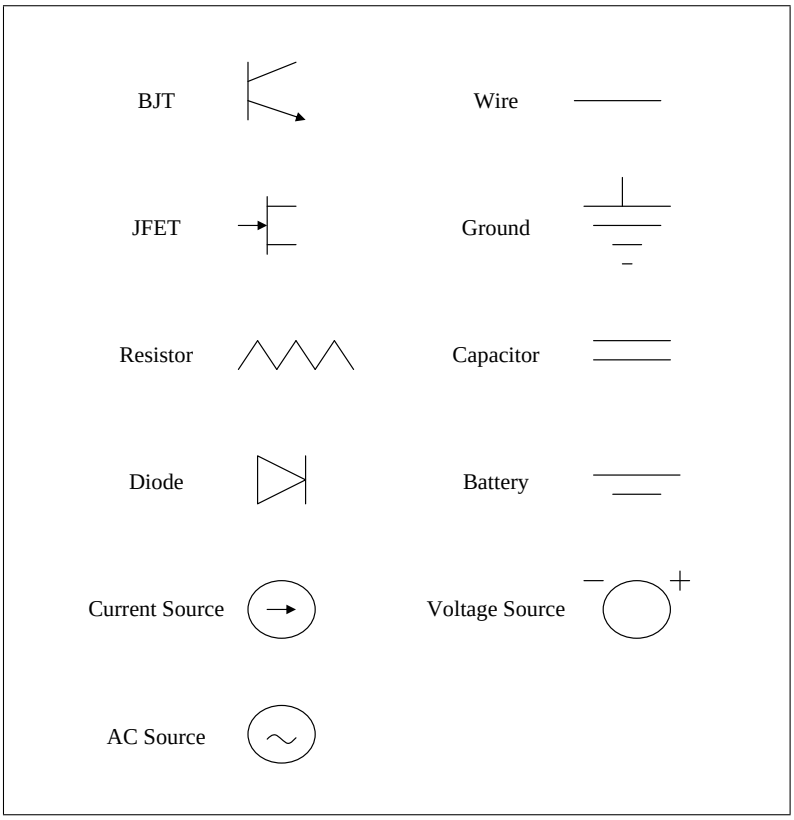
We consider three domains: family trees, flow charts, and circuit diagrams. Lexicons for each are given in Figure 2-1. The family tree domain contains five symbols: male, female, marriage or partnership link, divorce link, and parent-child link. The parent-child link is defined as linking one male or female to another male or female, thus each child is linked by an arrow to each parent (rather than joining the marriage or partnership to the child with a single arrow). The flow chart domain, with five symbols, and circuit domain, with eleven symbols, are each made up of a subset of standard symbols for such diagrams.



(a) Family tree symbols.



(b) Flow chart symbols.



(c) Circuit diagram symbols

Figure 2-1: Lexicons for each domain.

2.2 Data Sets

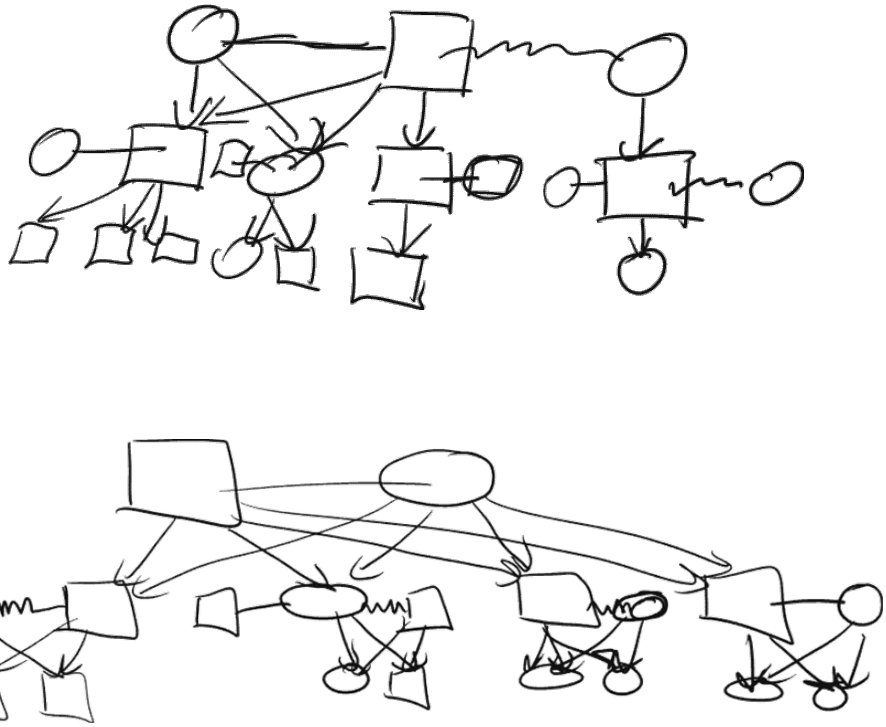
Unlike some areas of research, like speech recognition or handwriting recognition, the field of sketch recognition does not yet have large standardized data sets. Though work towards such data sets has begun, including work by Oltmans et al. [41], available data sets remain relatively small, on the order of tens or hundreds of sketches, rather than the thousands of handwritten addresses or phone interactions used in many recognition applications [28], [20].

	Number of sketches	Number of sketch authors	Average number of symbols per sketch
Family Tree	36	18	42.3
Flow Chart	36	18	23.6
Circuit	110	10	27.6

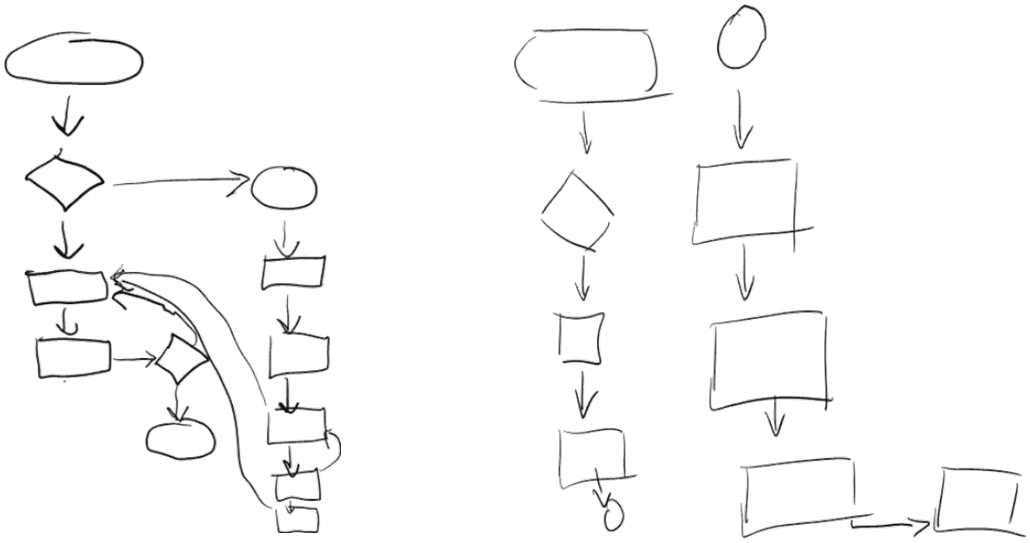
Table 2.1: Statistics for the data sets used throughout this work.

Table 2.1 gives statistics for the three data sets used in this research: family trees, flow charts, and circuit diagrams. Family tree and flow chart data was collected from the same set of subjects [9]. After a warm-up period to become familiar with the drawing environment, sketch authors were asked to draw their own family trees and then were given a textual description of an extended family and asked to draw the corresponding tree. The same subjects were also asked to draw flow charts representing their morning routines and were given a textual description of a common process and asked to draw the corresponding flow chart. Figure 2-2 contains examples of family tree and flow chart sketches.

Circuit diagram data collection was conducted by Alvarado [4]. Sketch authors familiar with circuit design were asked to draw several circuits, each with a specified numbers and types of components, but no constraints on function or layout. Figure 2-3 contains examples of circuit diagram sketches.



(a) Family tree sketches.



(b) Flow chart sketches.

Figure 2-2: Family tree and flow chart sketches.

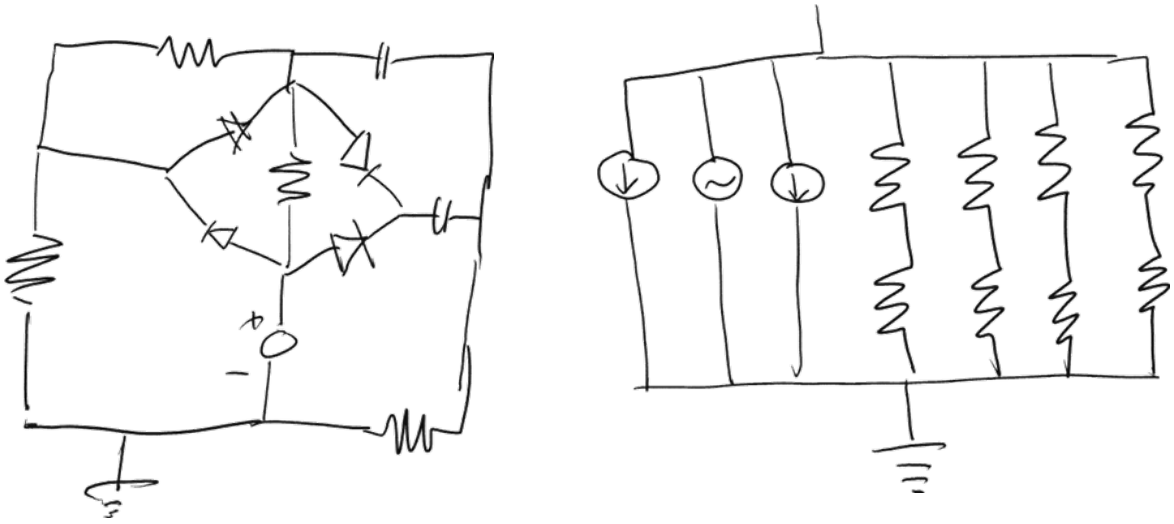


Figure 2-3: Circuit diagram sketches.

2.3 Single Representation Recognition Results

Table 2.2 contains recognition results for the three recognition systems described in Section 1.4.4 on each of the three data sets described above. Numbers in the table are the percentage of symbols recognized correctly, meaning the percentage of symbols across the entire domain that were both correctly localized and correctly classified.

	Family Tree	Flow Chart	Circuit
Spatial	21.3	**37.3**	**49.9**
Temporal	36.7	25.8	6.0
Conceptual	**47.3**	25.1	13.9

Table 2.2: Recognition results (% correct symbols) for the spatial, temporal, and conceptual recognition systems.

The goal of this thesis is to examine the improvement that combining representations can offer. Throughout this work, we use the results in Table 2.2 as a baseline for comparison. We seek to improve upon the single system recognition rates in Ta-

ble 2.2 and to produce a combined approach that outperforms the best single approach (highlighted in bold) across all three domains.

Chapter 3

Segmentation of Sketch Data

This chapter discusses sketch segmentation, one subproblem of recognition. Section 3.2 details a segmentation strategy corresponding to each of the three representations discussed in Section 1.4: spatial, temporal, and conceptual. The methods described in this chapter are based on human behavior and perception rather than domain knowledge, meaning that they do not rely on knowing what symbols may be drawn. We show that these segmentation methods may be combined to produce a good approximate segmentation. Section 3.3 demonstrates the resulting segmentations improve recognition by combining knowledge from different representations.

3.1 Basis for Segmentation without Domain Knowledge

Many methods of in-context recognition perform segmentation jointly with classification (the assigning of labels to stroke or pixel groups). Examples of this approach include [59], [42], and [69]. In these cases, segmentation relies on domain knowledge, which can be highly useful in the segmentation of a messy sketch with overlapping symbols. Consider the family tree sketch in Figure 3-1. In this sketch, knowledge of the concept of an arrow can be very useful for grouping strokes, because arrowheads tend to be closer to each other or closer to the shapes they point to than to the

corresponding arrow shaft strokes.

While in many cases a perfect segmentation requires this type of domain knowledge in order to group strokes or pixels, in this chapter we describe methods for approximate segmentation that do not rely on knowledge of what is being drawn. These methods are based on how humans tend to express ideas and perceive the world. Later in this chapter we show that these approaches provide useful segmentation information that complements domain specific approaches to recognition.

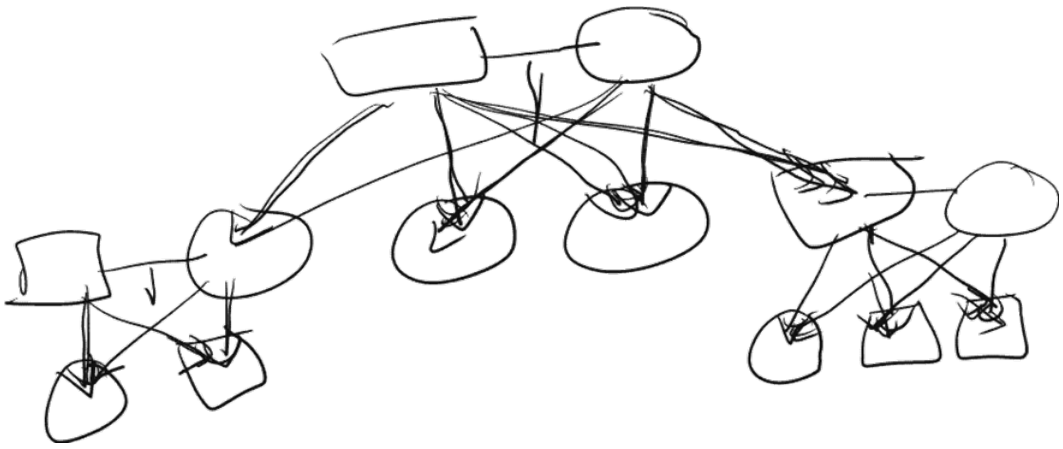


Figure 3-1: A family tree sketch. Knowledge of the concept of an arrow is useful for segmenting this sketch.

3.1.1 Human Behavior

Timing is one characteristic of human behavior that may implicitly convey meaning. In particular, pause placement and duration can aide in sketch segmentation. Authors are more likely to pause between drawing two symbols than in the middle of drawing one symbol. Across the data sets described in Chapter 2, the average length of time between temporally adjacent strokes within a symbol and between adjacent strokes in different symbols is significantly different, which suggests that useful segmentation clues are contained in this time stamp information. Mean pause lengths in milliseconds and statistics for t-tests for differences in the means are shown in Table 3.1.

The average length of time between temporally adjacent strokes within a symbol and between adjacent strokes in different symbols are significantly different for each of the three domains. Pause behavior has also been used for analyzing organization of spoken discourse. Grosz and Hirshberg [21] note the usefulness of pauses for locating topic boundaries.

	Within Symbol Mean Pause in ms	Across Symbol Mean Pause in ms	T-statistic
Family Tree	692	2683	9.09 ($p<.05$)
Flow Chart	1155	5707	13.14($p<.05$)
Circuit	982	1929	8.59($p<.05$)

Table 3.1: The average length of time between temporally adjacent strokes within a symbol and between adjacent strokes in different symbols are significantly different.

Continuity in expressing ideas is a second characteristic of human behavior that is useful for analyzing sketches. Sketch authors are likely to finish drawing one symbol before beginning the next. Therefore, temporal adjacency is also one indicator of symbol groupings, i.e., nonconsecutive strokes are less likely to belong to the same group than are consecutive strokes. It is not a perfect indicator, however, since interspersing of symbols does occur. For example, in Figure 3-2 the arrow bodies were all drawn before the arrow heads. In the sketches examined in this work between 3% and 22% of symbols are drawn with interspersed strokes.

Like temporal continuity, spatial continuity is useful as an indicator of segmentation: similar areas of ink that are near each other are more likely to be related than those that are not or those that are separated by blank areas of the page. However, this indicator is also imperfect when taken by itself, as Figure 3-1 illustrates.

These basic properties of human behavior underlie the spatial and temporal segmentation methods described in Section 3.2.

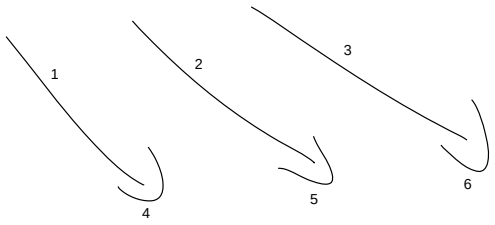
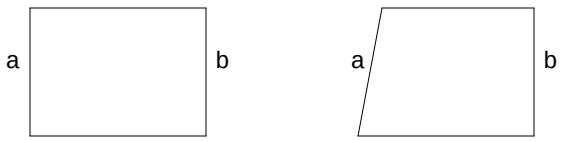


Figure 3-2: The arrow bodies were all drawn before the arrow heads. Numbers indicate drawing order.

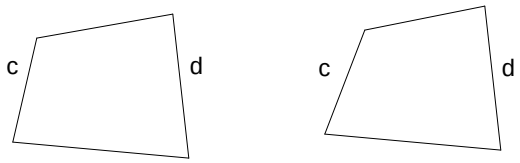
3.1.2 Human Perception

We also base segmentation on ideas about human perception, in particular that some relationships among shapes are more important than others. In tests of human perception, Goldmeier [19] noted that subjects preferentially notice some qualities of shapes, referred to as singularities. Salient singularities such as parallelism or verticality are special cases of geometry and are distinct because small variations away from the singularity affect perception of a symbol. Figure 3-3 illustrates a visual singularity. The small variation away from the parallelism of lines a and b affects the interpretation of the shape in 3-3(a) more than the equal variation in lines c and d in 3-3(b).

Such observations about human perception form the basis for the recognition model for learning from a single example proposed by Veselova [66]. In the segmentation model described in Section 3.2.3, we consider a subset of the geometric properties proposed by Veselova, including parallelism, perpendicularity, and equality of size (Table 3.2 contains the full list of properties). In the next section we describe how grouping these important shape properties can yield a sketch segmentation.



(a)



(b)

Figure 3-3: Parallelism as a singularity. The same change in the angle of line a and line c affects the perception of the shapes differently.

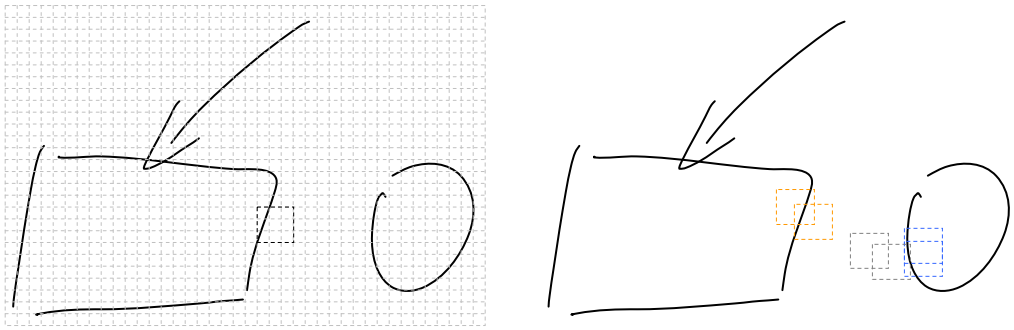
3.2 Representation Based Segmentation Methods

Each of the three sketch representations described in Section 1.4 attends to different aspects of a sketch and highlights different information contained within it. This section describes how to use these differing sets of information to produce different segmentations, which may then be combined, yielding a better approximate segmentation. These approximate segmentations are then used in Section 3.3 to improve on previous recognition results.

3.2.1 Spatial

A spatial representation is concerned with the distribution of ink on a page or colored pixels on a screen. In any drawing, the distribution of ink varies, with areas of higher ink density separated by areas of lower ink density. A spatial segmentation may be constructed by dividing relatively dense areas from areas that are relatively empty.

To produce a spatial segmentation, first the ink density is calculated for many overlapping windows. Figure 3-4(a) illustrates these windows. Ink density is defined as the number of drawn points divided by the area of the window. Next, windows are grouped with bottom-up clustering. Each window begins as its own group and overlapping groups with similar ink densities are merged. Figures 3-4(b) contains six windows that have been grouped in three clusters, indicated by different colors. Windows of the same color are overlapping and have similar ink density. Finally strokes are segmented at corners according to the method described by Sezgin et. al [57] and stroke segments are assigned to only one group, meaning strokes may not be divided between two groups at arbitrary points. This is a simplifying assumption that departs somewhat from a purely spatial method.



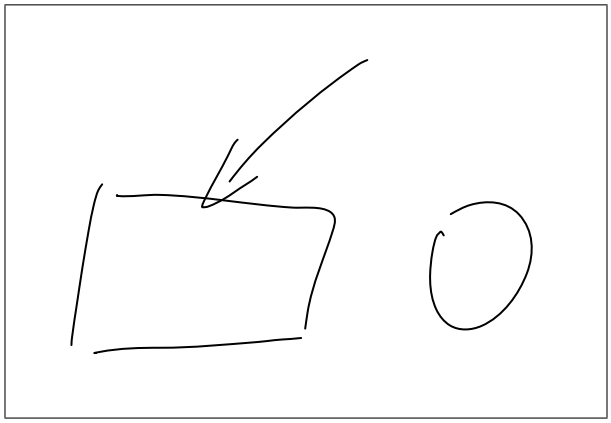
(a) (b)

Figure 3-4: To produce a spatial segmentation, the sketch is first scanned and ink density is found for many overlapping windows. Overlapping windows with similar density are then clustered.

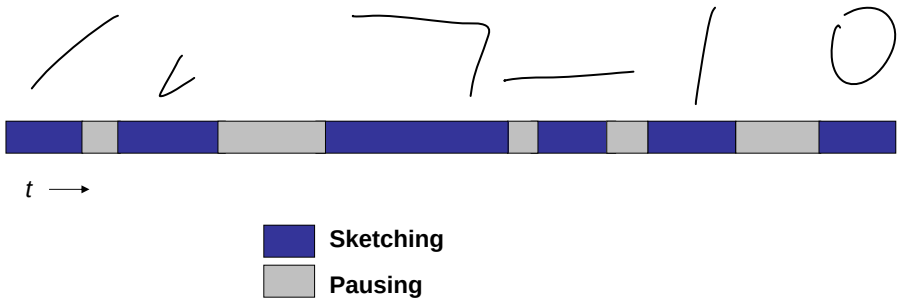
3.2.2 Temporal

A temporal representation views the sketch as a sequence of events. For the purpose of this segmentation method, we consider only two types of events: sketching and pausing. Sketching refers to any uninterrupted time period during which the pen is in contact with the screen (or page). Pausing refers to any block of time when no ink is

being laid on the screen (or page). Figure 3-5(b) illustrates the timeline corresponding to the sequence of events in the creation of the sketch shown in Figure 3-5(a).



(a)



(b)

Figure 3-5: An excerpt of a larger family tree sketch and the timeline illustrating how it was drawn.

We segment a sketch temporally by using a bottom-up clustering approach, similar to the spatial segmentation method described in the last section. Each stroke segment begins as its own group, and groups are continually merged as long as two groups are close temporally. Temporal distance between two stroke groups is defined as the ratio of the length of the pause between stroke groups to length of sketching time of the adjacent groups. Thus the temporal distance is relative: short strokes separated by a long pause would not be grouped, but longer strokes separated by the same

pause length could be. This results in groups of temporally adjacent strokes that were drawn in relatively quick succession, with groups separated by longer pauses. Figure 3-6 demonstrates this successive grouping of strokes for the sketch in Figure 3-5(a). The bottom row of strokes contains each stroke as its own group, and each higher row shows two stroke groups being merged.

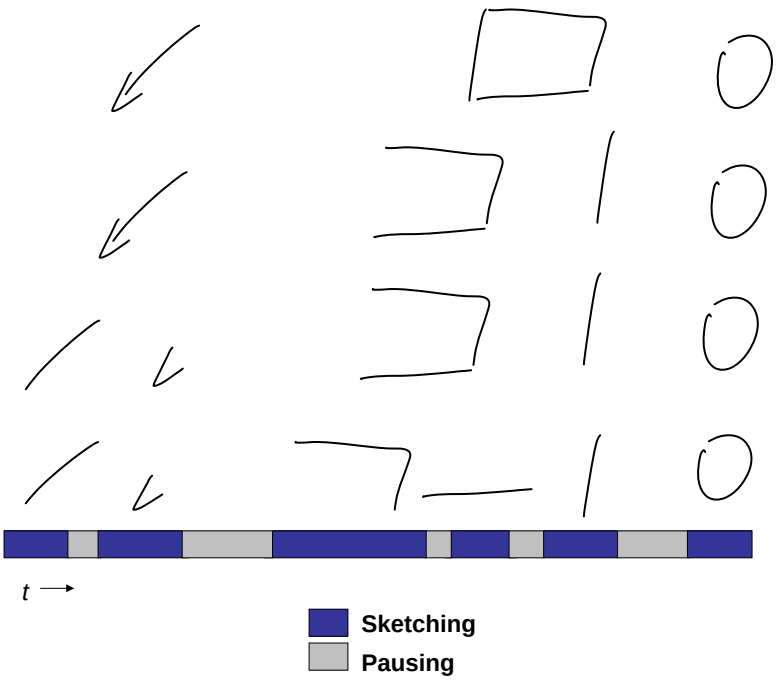


Figure 3-6: Temporal clustering resulting in the segmentation of the sketch in Figure 3-5(a).

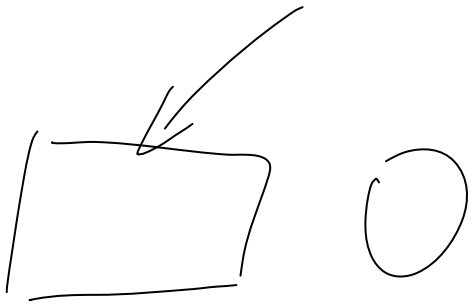
3.2.3 Conceptual

We produce a conceptual segmentation of a sketch, by representing the conceptual elements as a graph. Segmenting the graph yields a corresponding segmentation of the sketch. We define two types of conceptual elements: components and the constraints between components. This representation is based on the LADDER shape description language [23].

Components	Constraints	
Line	Intersect	Parallel
Ellipse	Perpendicular	Contains
Arc	Coincident	Touching
	Equal	

Table 3.2: Geometric components and constraints used in sketch descriptions.

First, a low level processor identifies components, including lines, arcs, and ellipses [57]. Single strokes may be broken at corners to produce more than one component. Next, pairs of components are tested exhaustively for the binary constraints listed in Table 3.2. These two steps result in a list of components and constraints. Figure 3-7(b) contains an excerpt of the description of the conceptual representation, formatted as a LADDER shape description, corresponding to the same sketch in Figure 3-5(a) reproduced in Figure 3-7(a).



```
(define shape sketch
(components
   (Line line1)
   (Line line2)
   (Line line3)
   (Line line4)
   (Line line5)
   (Line line6)
   (Line line7)
   (Ellipse ellipse1)
    ...
 )
 (constraints
   (touching line1 line2)
   (touching line2 line3)
   (touching line3 line4)
   (touching line4 line1)
   (perpendicular line1 line2)
   (perpendicular line3 line4)
   (parallel line2 line4)
   (parallel line1 line3)
   (equal line1 line3)
   (touching line6 line2)
   (touching line7 line2)
   (equal line6 line7)
    ...
 )
)
```

(a) (b)

Figure 3-7: An excerpt from a larger family tree sketch and a partial description of its components and constraints.

Next a graph is constructed that represents components as nodes and constraints between components as edges. Figure 3-8 contains the graph corresponding to the partial text description in Figures 3-7.

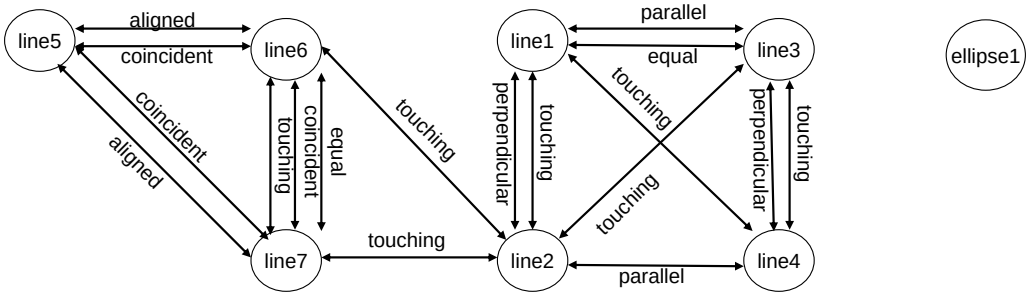
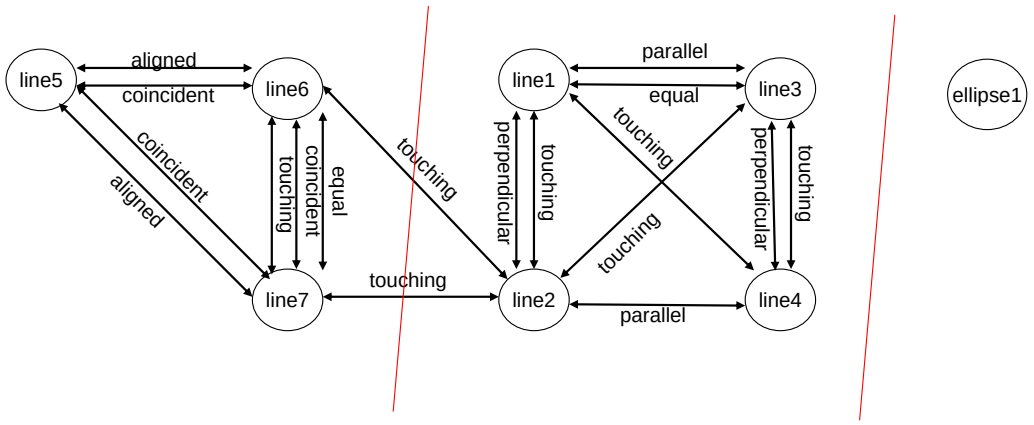
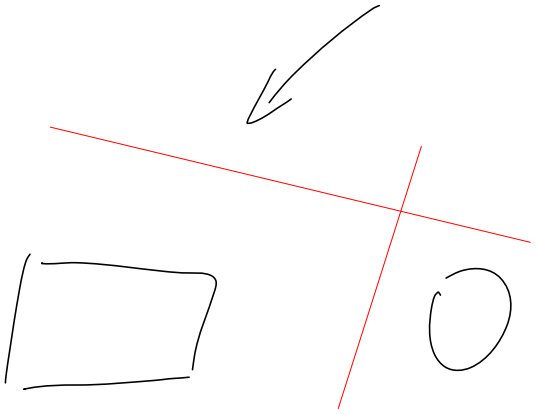


Figure 3-8: The constraint graph corresponding to the sketch and description in Figure 3-7.

The constraint graph is divided by computing the minimum cut, defined as a partition of the graph into two disjoint sets such that the number of edges with end points in different sets is minimized. The graph is further divided by computing the single minimum cut among the two resulting subgraphs. This process continues while the minimum cut is less than an empirically set threshold. Figure 3-9 illustrates the minimum cuts of the graph in Figure 3-8 and the resulting segmentation. By dividing the graph through areas of minimal connectivity thus preserving densely connected groups of nodes, groupings of constraints and their components in the sketch are grouped as well.



(a)



(b)

Figure 3-9: The minimum cuts of the graph in 3-8 and the resulting segmentation.

3.2.4 Segmentation Results

Figure 3-10 contains a sketch from the flow chart data set, and Figure 3-11 contains a segmentation of that sketch with each of the three methods described in this section: spatial(3-11(a)), temporal(3-11(b)), and conceptual(3-11(c)). The three resulting segmentations are all different. They all also undersegment, meaning that many groups contain more than one symbol. The segmentation methods described in this section are not intended to be used alone, since by using single representations they each attend to valuable but incomplete information. Rather they are to be taken together,

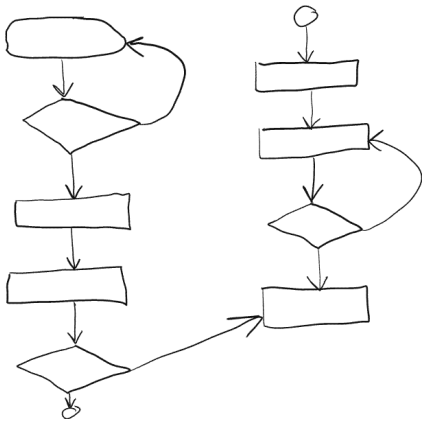


Figure 3-10: A sketch of a flow chart.

with each contributing some segmentation indicators. Thresholds that determine how finely each single representation method segments a sketch are set empirically.

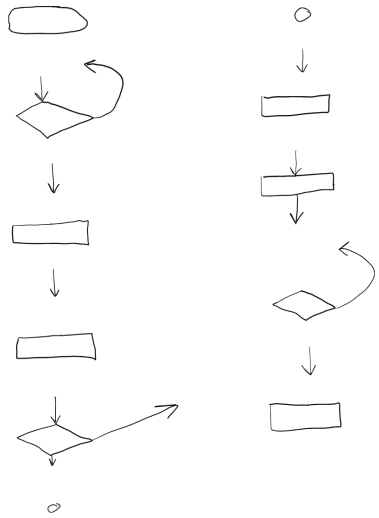
Figure 3-12 shows a combined segmentation, produced by considering all of the break points found by the spatial, temporal, and conceptual segmentations: if a break point is found in any of the three representation based segmentations it is included in the combined segmentation.

The segmentation in Figure 3-12 contains only one mistake, highlighted in Figure 3-13: the arrow and rectangle should not be grouped together. While this sketch appears quite neatly drawn, this near perfect segmentation was produced with no knowledge of the domain, that is no knowledge of what an arrow, rectangle, or diamond is, and allows for symbols consisting of any number of strokes or stroke segments.

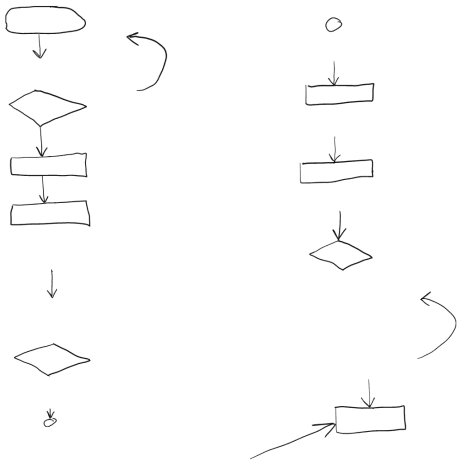
Table 3.3 contains the segmentation accuracy for the combined segmentation for each of the data sets described in Chapter 2. Accuracy is defined as the percentage of symbols, across all sketches in the domain, that were correctly isolated.

	% correct segmentation
Family Tree	48.3
Flow Chart	59.5
Circuit	29.7

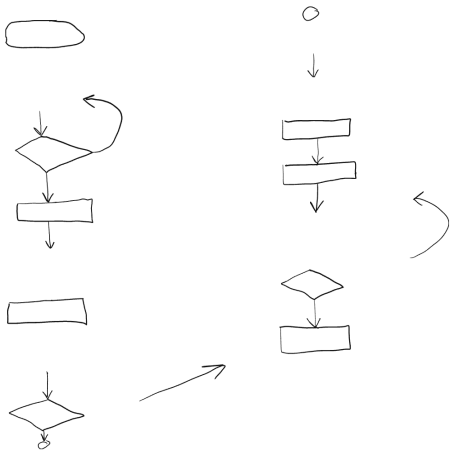
Table 3.3: Segmentation accuracy.



(a) Spatial segmentation.



(b) Temporal segmentation.



(c) Conceptual segmentation.

Figure 3-11: Segmentations of a flow chart sketch.

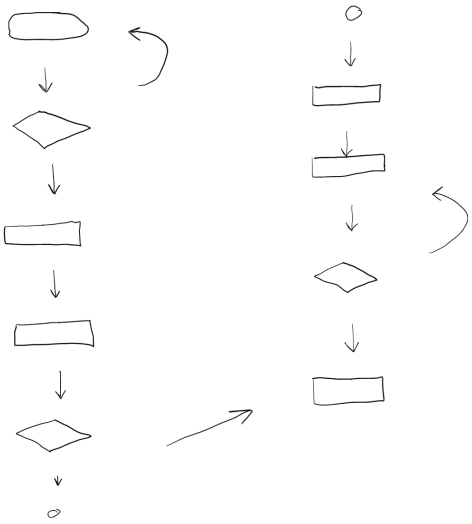


Figure 3-12: Segmentation resulting from the combination of the representation specific segmentations in Figure 3-11.

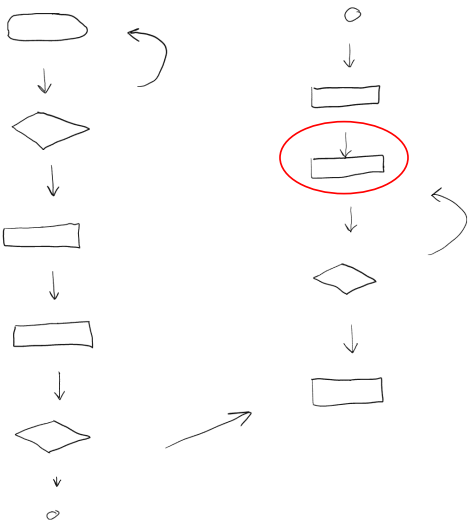
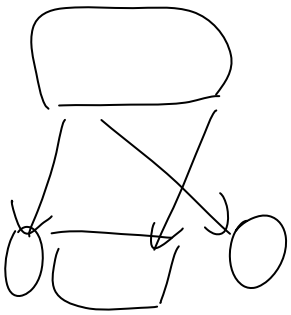


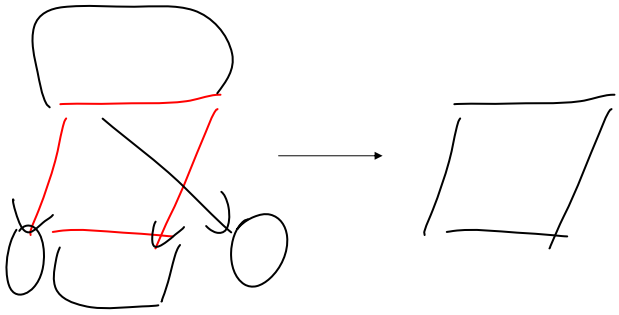
Figure 3-13: Segmentation error in Figure 3-12 indicated by the circle.

3.3 Segmentation as a Pre-processing Stage of Recognition

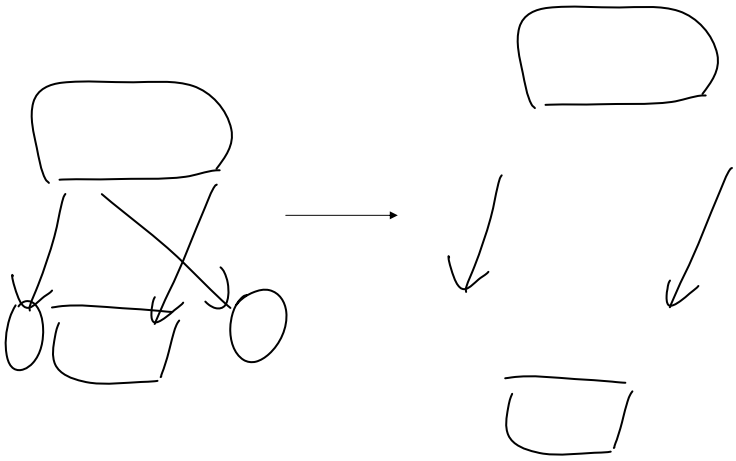
The method described above for approximate segmentation may have applications on its own, for example, neatening a messy sketch or applications such as those described by [29] and [53] where recognition is not required or desirable. However, segmentation as a pre-processing step can also boost performance of existing recognition systems, which themselves also perform segmentation.



(a) Excerpt from a family tree sketch.



(b) Incorrect segmentation.



(c) Correct segmentation.

Figure 3-14: Incorrect segmentation of one symbol can result in other segmentation and classification mistakes.

Recognition systems that perform both segmentation and classification, such as those described in Section 1.4.4, may be subject to error propagation, particularly if stroke segments may not be assigned to more than one symbol. A segmentation mistake (or classification mistake, if context is being used to aide recognition as in [5]) in one part of a sketch may cause segmentation mistakes elsewhere in the sketch. Those segmentation mistakes are then likely to cause classification errors.

Consider the example in Figure 3-14. The red square in Figure 3-14(b) is formed from strokes that belong in four different symbols. This error precludes the correct segmentation of the four symbols (two arrows and two rectangles), shown in Figure 3-14(c). This section describes how small, cross-representational segmentation clues can prevent this sort of error propagation.

3.3.1 Method

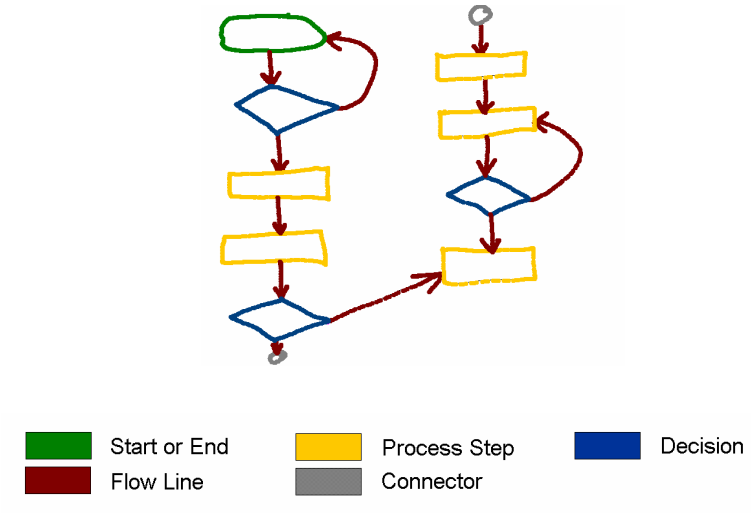
In order to boost the performance of an existing recognition system based on a single representation, sketch data is first pre-processed with complementary representations. Sketches are roughly segmented according to the methods described in Section 3.2, taking strong indicators of symbols boundaries and erring on the side of undersegmentation. Segmentation is based on complementary representations than that of the recognition step. The resulting segments are then recognized independently from one another, meaning that a symbol may not span both sides of a segmentation boundary.

3.3.2 Recognition Results

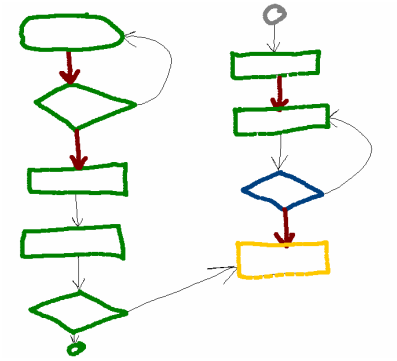
Figure 3-15(a) shows the correctly labeled flow chart sketch from Figure 3-10. Colors indicate symbol labels: diamonds(decision symbols), for example, are blue and ovals(start/end symbols) are green.

Figure 3-15(b) shows results of processing the sketch using the conceptual representation based system described in Section 1.4.4. If, however, we first presegment the sketch according to strong temporal and spatial clues, then process it with the same conceptual representation based system, we get the results in Figure 3-15(c).

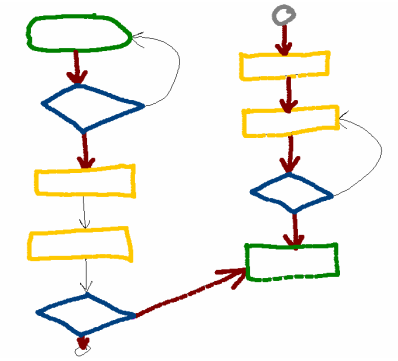
Recognition is improved significantly by considering strong break points in other representations to be enforced symbol boundaries.



(a) Correctly labeled flow chart sketch.



(b) Conceptual recognition results of the sketch in part (a).



(c) Conceptual recognition results after presegmenting.

Figure 3-15: Presegmenting with spatial and temporal methods improves later conceptual recognition.

Table 3.4 contains quantitative results for this method of presegmenting, then recognizing sketches. In all cases sketches are presegmented with strong break points from two representations and then recognized with the remaining representation, noted in the table.

The results in Table 3.4 indicate that by applying the segmentation methods described in this chapter, the single system recognition described in Chapter 2 (results repeated in Table 3.5) is improved in all cases, and significantly so in some cases. This suggests that domain independent indicators of segmentation can compliment segmentation and recognition methods that rely on domain knowledge.

Presegmented

	Family Tree	Flow Chart	Circuit
Spatial	62.3	51.3	63.1
Temporal	37.5	45.6	13.4
Conceptual	60.1	54.9	25.7

Table 3.4: Recognition accuracy for sketches presegmented with cross-representational information.

Single Representation

	Family Tree	Flow Chart	Circuit
Spatial	21.3	37.3	49.9
Temporal	36.7	25.8	6.0
Conceptual	47.3	25.1	13.9

Table 3.5: Recognition results (% correct symbols) for the individual spatial, temporal, and conceptual recognition systems.

Chapter 4

Judging Recognition Credibility with Representation Specific Complexity Measures

This chapter explores the idea that different representations are good at different things, and specifically that one representation may be better or worse at recognizing symbols in a particular domain than another representation. When combining representations we would like to take advantage of these variations. This chapter suggests ways of predicting such differences in performance prospectively, i.e., without having to build recognition systems. In Section 4.1 we propose ways of predicting performance for a domain, which can be used to select the best recognition system for that domain. Section 4.2.2 extends this idea to the symbol level and introduces symbol confusability metrics, which judge how well a system can be expected to recognize a particular symbol within a domain. Section 4.3 applies the metrics of Section 4.2.2 to combine recognition methods.

4.1 Domain Level Complexity

The results in Table 2.2, repeated in Table 4.1, show that the accuracies of the recognition systems described in Section 1.4.4 vary, in both absolute value and relative

to one another. For example the spatial system has the best performance on the circuit domain, but it has the worst performance on the family tree domain. There is not a single best method for recognition across all domains.

	Family Tree	Flow Chart	Circuit
Spatial	21.3	37.3	49.9
Temporal	36.7	25.8	6.0
Conceptual	47.3	25.1	13.9

Table 4.1: Recognition results (% correct symbols) for the spatial, temporal, and conceptual recognition systems.

Table 4.1 suggests that the difficulty of the recognition problems for each system can vary. One could determine how difficult a recognition problem is for the three systems by simply running each on available data. However, we would like to prospectively decide what type of system, or what approach to take for a new recognition problem. This leads to the questions: what makes a problem hard for a spatial representation, for a temporal representation, and for a conceptual representation? As partial answers to those questions, we propose the domain complexity measures listed in the left column of Table 4.2. A high degree of spatial overlap among symbols complicates recognition for a system based on a spatial representation, which may not be able to distinguish ink from one symbol if it is drawn on top of another symbol. Similarly, a high degree of temporal overlap, or the interspersing of symbol strokes while drawing, complicates recognition for a system that distinguishes symbols based on their temporal patterns. For a conceptually based system, the structural complexity of a domain's symbols, as measured here by the length of the LADDER shape descriptions for the symbols [23], increases the difficulty of recognition for a domain. A symbol with a longer description has a greater number of subcomponents that may be incorrectly located or classified.

	Family Tree	Flow Chart	Circuit
Mean Symbol Overlap in Pixels (a spatial metric)	565.2	297.9	43.8
Percentage Interspersed Symbols (a temporal metric)	3.8	5.5	22.0
Mean Symbol Description Length (a conceptual metric)	3.8	7.2	7.9

Table 4.2: Domain complexity measures.

Table 4.2 contains values for each of the properties described above for each domain. Rows in Table 4.2 correspond to the rows of Table 4.1, i.e., mean symbol overlap is a spatial complexity measure. Low numbers in Table 4.2 indicate less complex recognition problems (for example, the description length value of 3.8 in the third row is the smallest in that row, suggesting that the family tree domain is the simplest domain when viewed conceptually). Those cells with lowest values in Table 4.2 (the least complex recognition problems) correspond to the highest recognition rates in Table 4.1. This relationship is shown graphically in Figures 4-1, 4-2 and 4-3. These figures illustrate that recognition is more accurate in less complex domains. This mirrors our intuition that performance on an easy problem is likely to be better than performance on a harder problem.

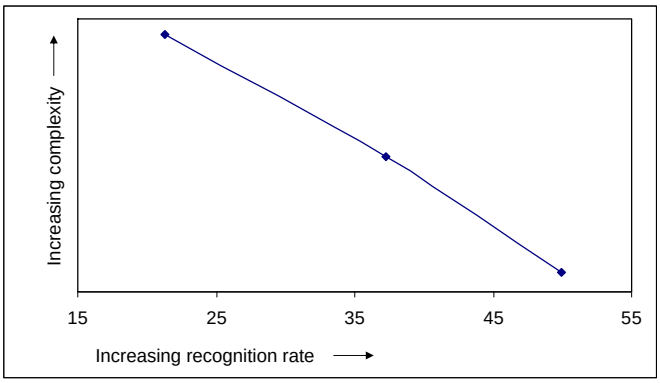


Figure 4-1: The relationship between our spatial domain complexity measure and recognition rates.

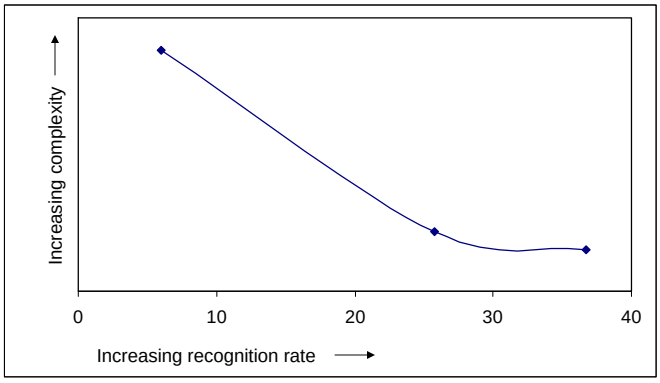


Figure 4-2: The relationship between our temporal domain complexity measure and recognition rates.

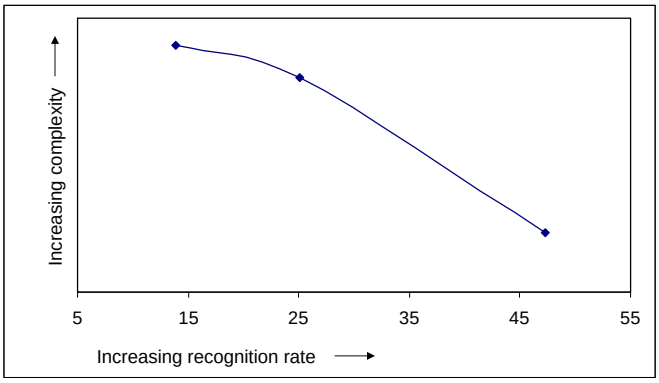


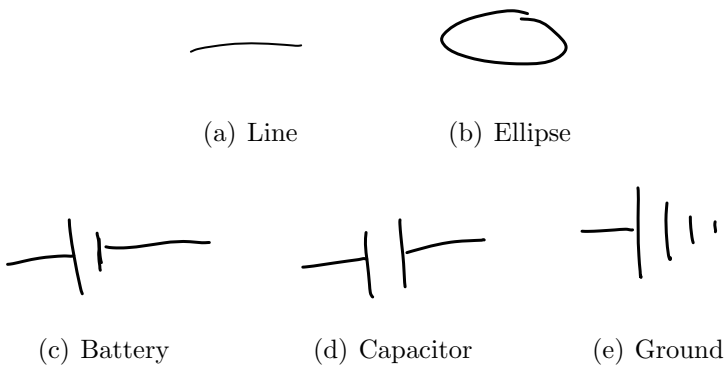
Figure 4-3: The relationship between our conceptual domain complexity measure and recognition rates.

4.2 Symbol Level Complexity

The domain level guidance for selecting the best representation for recognition can be useful: as shown above, a domain may not be equally hard for all recognition systems or methods. However, domains themselves are often not uniform either; even within a domain one representation may be better for recognizing a particular symbol than another. Extending the idea of complexity as a predictor of performance to the symbol level can further improve recognition.

4.2.1 Confusion Matrices as Indicators of Complexity

The difficulty of recognizing a symbol depends on what other symbols are in its domain (i.e., what else it might be confused with). Consider the shapes in Figure 4-4. Distinguishing between a line and an ellipse is generally not difficult, so recognition in a domain with only those two symbols would likely not pose a problem for any representation or recognition system. However our task is rarely so easy. More commonly a domain will contain symbols that are easily confused. The circuit domain contains symbols (c), (d), and (e) in Figure 4-4. Distinguishing these three symbols is much harder. A slight variation in relative lengths of the two parallel lines in the battery and capacitor can change their interpretations, and a small third line, which could be missed entirely, distinguishes the battery from the ground.



(a) Line (b) Ellipse

(c) Battery (d) Capacitor (e) Ground

Figure 4-4: Symbols a and b are from the flow chart domain. Symbols c, d, and e are from the circuit domain. The ellipse is more likely to be confused with the circle than the battery, and likewise the battery is more likely to be confused with the capacitor or ground than with symbols from the flow chart domain.

As the above example illustrates, complexity should take into account a symbol's confusability with other symbols. A confusion matrix can provide this information empirically. Figure 4-5 contains an example of a confusion matrix generated by testing the conceptual recognition system on a subset of family tree data. Rows correspond to correct symbol labels, and columns correspond to the labels assigned by a recognition system: the value 45 in cell (1,2) is a count of the number of times the recognition

system found a female symbol where there was actually a male symbol. High off-diagonal numbers indicate that two symbols are easily confused by a given system, while low off-diagonal number indicate they are not often confused.

	Male	Female	Parent/child	Marriage	Divorce
Male	175	45	3	18	0
Female	7	213	0	0	0
Parent/child	0	48	82	75	0
Marriage	0	0	12	37	0
Divorce	1	0	4	32	0

Figure 4-5: Confusion matrix for the conceptual recognition system on a subset of family tree data.

4.2.2 Approximating Confusion Matrices with Representation Specific Distance Metrics

A confusion matrix can provide the desired measure of symbol confusability. However, generating a confusion matrix in this way requires building recognition systems first, and one of the goals of this work is to provide guidance for building future sketch systems. So, as in the previous section, we would like to be able to obtain this guidance without having to build and test a system, and possibly even before empirical data has been collected for a new domain.

Our approach to this problem is based on a very simple but powerful observations: a system confuses two symbols because they are similar according to some representation. In response we approximate a confusion matrix by defining and measuring the representation specific similarity of pairs of symbols in a domain. This offers two important advantages. First, while the similarity measures are representation specific, they are domain independent and can easily be applied to the symbols in a new

domain. Second this gives us our desired prospective metric, allowing us to compute a confusion matrix without building and running a system. The remainder of this section defines a distance metric for each of the three representations.

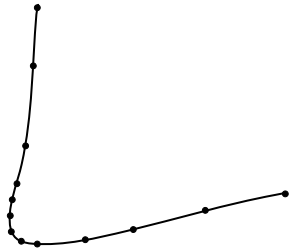
Spatial

To calculate a spatial distance between two symbols, each symbol is represented as an edge direction histogram. Symbols are first rescaled to eliminate differences due only to size, then resampled to yield points at a constant spatial frequency (rather than the constant temporal frequency of points of the sketches collected with a digital pen). These preprocessing steps are common for spatially based recognition methods [41], [26]. Figures 4-6(a) and 4-6(b) illustrate resampling. In part(a) points are distributed evenly in time; there are more points around the corner because the pen was moving more slowly there. In part (b) points have been redistributed evenly along the length of the stroke.

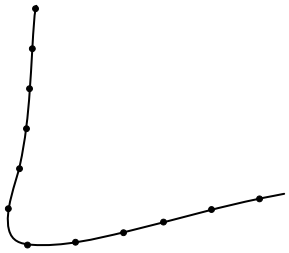
Next, edge directions are calculated as the angles of the lines connecting neighboring points, and these directions are binned, forming histograms. Figures 4-6(c) and 4-6(d) illustrate these steps. The angles between the points in the stroke in part (c) fall mostly into the near horizontal bins (around zero radians) and near vertical bins (around $\pi/2$ radians), shown in part (d) of the figure.

The distance between the two histograms is then calculated as the Euclidean distance between the vectors (the sum of the squared difference in bin counts). This distance is expressed mathematically in Equation 4.1, where S_1 and S_2 are two symbols to be compared and $H(S_1)$ and $H(S_2)$ are corresponding n-dimensional histograms.

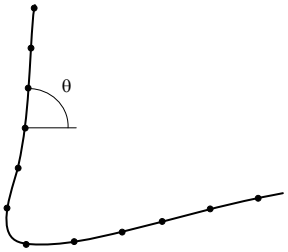
$$d(S_1, S_2) = \sqrt{\sum_{i=1}^{n} (H_i(S_1) - H_i(S_2))^2} \tag{4.1}$$



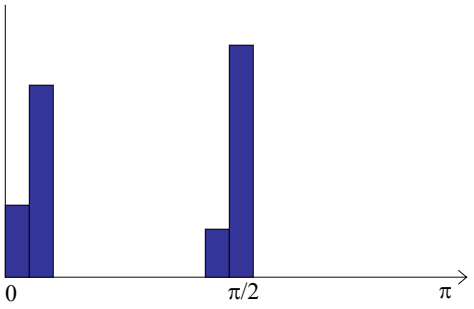
(a) Originally points are distributed with a constant temporal frequency.



(b) Points are resampled to have constant spatial frequency.



(c) Edge directions are computed as the angle between adjacent points.



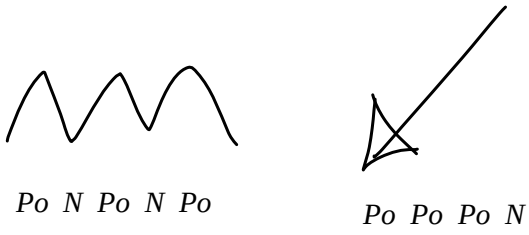
(d) Angles are binned to form histograms.

Figure 4-6: Steps in the generation of an edge direction histogram.

Temporal

We represent symbols temporally as a sequence of different kinds of strokes, for example differently sloped lines. Temporal similarity is measured with a string edit distance, which is a weighted sum of the number of insertion, deletion, and substitution operations required to transform one string into another. Figure 4-7 contains two symbols, along with their temporal representations. The symbols Po and N refer to positively and negatively sloped lines. For example, the resistor in Figure 4-7(a) was drawn with a positively sloped line, followed by a negatively sloped line, followed by another positively sloped line, and so forth. An efficient sequence of operations

that transforms the resistor into the arrow is $Po\ N\ Po\ N\ Po \rightarrow Po\ Po\ Po\ N\ Po \rightarrow Po\ Po\ Po\ N$, which requires one substitution and one deletion.



(a) Resistor (b) Arrow

Figure 4-7: Temporal representation of a resistor and an arrow.

Equation 4.2 contains the mathematical representation of the temporal distance, where S_1 and S_2 are two strings to be compared and $\mathbf{w}$ is the set of weights and $\mathbf{c}$ is the set of counts for insertions, deletions, and substitutions.

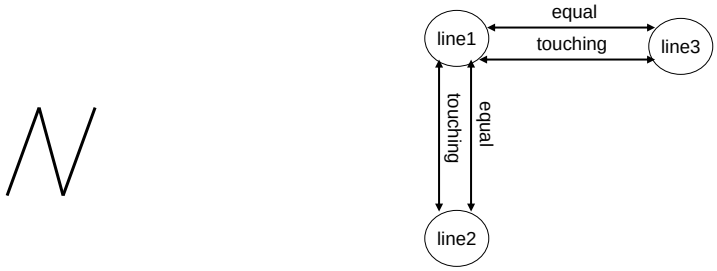
$$d(S_1, S_2) = \mathbf{w} \cdot \mathbf{c} \tag{4.2}$$
$$\mathbf{w} = [w_i, w_d, w_s]$$
$$\mathbf{c} = [c_i, c_d, c_s]$$

Conceptual

The conceptual distance is an edit distance as well, though it is the distance between constraint graphs (described in Section 3.2.3), rather than strings. The graph edit distance is a weighted sum of the number of insertions, deletions and substitutions required to transform one graph into another, but operations are counted for both nodes and edges, yielding six transforming operations rather than three, as is the case for the temporal string edit distance.

Figure 4-8 contains two simple symbols along with their constraint graphs. The graph in part (a) may be transformed into the graph in part (b) through the sequence of one edge deletion, one node insertion, and two edge insertions shown in Figure 4-9.

(a)



(b)

Figure 4-8: Conceptual graph representations of two simple symbols.

Equation 4.3 contains the mathematical representation of the conceptual distance, where G_1 and G_2 are two constraint graphs to be compared, $\mathbf{c}$ is a vector of counts of node insertions, node deletions, node substitutions, edge insertions, edge deletions, and edge substitutions, and $\mathbf{w}$ is a corresponding vector of weights.

$$
\begin{aligned}
d(G_1, G_2) &= \min_{V(G_1) \to V(G_2)} (\mathbf{w} \cdot \mathbf{c}) \\
\mathbf{w} &= [w_{ni}, w_{nd}, w_{ns}, w_{ei}, w_{ed}, w_{es}] \\
\mathbf{c} &= [c_{ni}, c_{nd}, c_{ns}, c_{ei}, c_{ed}, c_{es}]
\end{aligned}
\tag{4.3}
$$

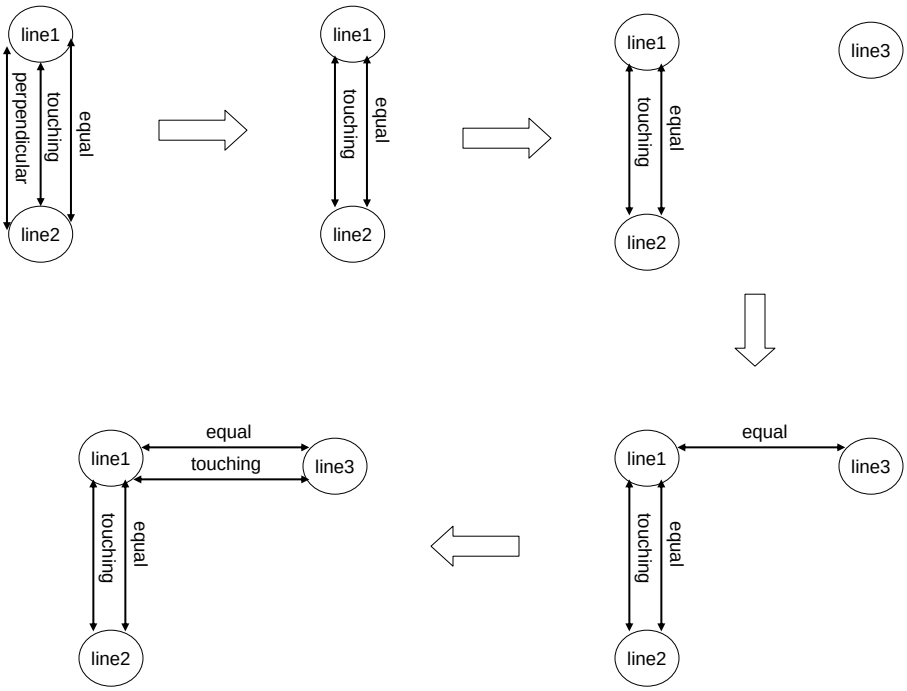


Figure 4-9: Sequence of edits required to convert one graph into another.

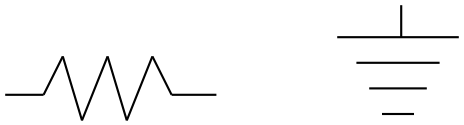
4.2.3 Symbol Versions for Computing Distances

The distances above may be applied to symbols in two ways. They may be used to measure the distance between ideal instances of symbols or between empirical instances of symbols (i.e., symbols as drawn by users). In both cases, the similarity of two symbols in a domain is computed from the average pair-wise distance across all pairs of instances of the symbols being considered.

Ideal

Ideal instances of symbols are constructed from their definitions in the domain, which may come from LADDER shape descriptions (as in Figure 1-7), text-book symbol definitions (as in Figure 2-1), or canonical ways of drawing a symbol. The result is generally a single instance of a symbol, but several ideal versions are possible if a symbol's definition includes more than one valid way of drawing it. Figure 4-

10 illustrates ideal spatial versions of a resistor symbol and a ground symbol. The advantage of this method is that it may be computed before data is collected; however, it may not take into account many possible variations or the imprecision and messiness in hand drawn sketches.



(a) (b)

Figure 4-10: Ideal versions of a resistor and ground symbol.

Empirical

Alternatively, if empirical data is available, symbol instances may be gathered from actual sketches, by hand segmenting and labeling. Figures 4-11 and 4-12 contains several instances of resistors and grounds gathered from sketches. Confusability between these two symbols is calculated from the average distance between a resistor from set (a) and a ground from set(b). Although this method does require user generated data, it may still be applied without building a system. Results in Section 4.3 rely on this method of distance calculation.

Figure 4-11: Several sketched versions of resistor symbols.

Figure 4-12: Several sketched versions of ground symbols.

4.3 Combining Recognition Systems Based on Symbol Level Complexity

This section presents two methods for applying symbol complexity to improve recognition. The first combines recognition systems in parallel and uses symbol confusability to determine how to combine their outputs. The second combines recognition systems serially by selectively training systems according to their strengths.

4.3.1 Weighting Recognition Results Based on Credibility

Method

To create a combined recognition result, each recognition system is run on a sketch, generating a collection of symbol labels and their locations within the sketch. In many cases the systems do not agree, requiring that we have some way of determining how to combine their opinions. We do this by estimating the likelihood of a symbol being present given the recognition results of each system and combining those likelihoods to generate an overall likelihood.

We can generate conditional probabilities for each label (e.g. the probability that a symbol is a capacitor given that a temporal system has labeled it a ground) from a confusion matrix. In the confusion matrix in equation 4.4, $n_{i,j}, (i \neq j)$ is a count of

the number of times the k^{th} recognition system misclassified a symbol as having the j^{th} label when it actually belonged to the class i.

$$CM_k = \left\{ \begin{array}{ccccc} n_{0,0} & \cdots & & & \\ \vdots & \ddots & & & \\ & & & & \\ & & & n_{i,j} & \\ & & & & \ddots \end{array} \right\} \tag{4.4}$$

The chance that a symbol x should belong to the i^{th} class, given that the k^{th} recognition system labeled it as belonging to the j^{th} class is:

$$P(x \in c_i | s_k(x) \in c_j) = \frac{n_{ij}}{\sum_{i=0}^{n} n_{ij}} \tag{4.5}$$

Confusion matrices for this calculation may either be generated with held out training data or approximated with the distance metrics described in Section 4.2.2. When actual data is used, additional rows may be added to a confusion matrix to account for missed symbols and over-recognized symbols, and those conditional probabilities included as well.

For simplicity we assume that the recognition results are independent. Empirically we have found that prior probabilities for each type of symbol may also be assumed to be equal. Given recognition results and estimated conditional probabilities, the likelihood of each label at a given location is estimated by the product of the probability in equation 4.5 for each system:

$$\prod_{k=1}^{K} P(x \in c_i | s_k(x)) \tag{4.6}$$

The most likely label is that with the highest likelihood, found by maximizing Equation 4.6 over i.

Results

Table 4.3 contains recognition results for the method described above using both actual confusion matrices and approximated confusion matrices based on distance metrics. These results are compared with the baseline results, presented in Section 2.3.

	Family Tree	Flow Chart	Circuit
Combination with Confusion Matrices	56.0	52.3	55.4
Combination with Distance Metrics	54.7	51.0	52.9
Baseline	47.3	37.3	49.9

Table 4.3: Recognition results (% correct symbols) for the combination of recognition systems.

In both cases the combined recognition performance exceeds our base line, (defined in Chapter 2 as selecting the single best system for each domain). These results indicate that symbol confusability, as defined by a confusion matrix, can be used to arbitrate between recognition systems at the symbol level and is a useful measure of recognition difficulty for a given recognition system. Furthermore, distance metrics proposed in this chapter, which do not require running an actual recognition system, may be used to approximate confusion matrices and also provide a useful measure of recognition difficulty.

By first applying the segmentation methods from Chapter 3, then using the combined recognition approach above, we find some improvement over the results presented Chapter 3. It is worth noting however, that presegmentation followed by spatial recognition performs almost as well as presegmentation followed by the combined recognition method above. This comparison is shown in Table 4.4. Spatial methods, including the one used in this work by Oltmans, have previously been shown to be highly reliable for classifying isolated symbols [40, 43, 26]. Our results suggest that while conceptual and temporal methods do correctly classify some symbols that the spatial system cannot, their greatest value may come from improved segmentation and shape localization.

	Family Tree	Flow Chart	Circuit
Presegmenation + Combined Recognition	63.0	55.4	62.7
Presegmentation + Spatial Recognition	62.3	51.3	63.1
Baseline	47.3	37.3	49.9

Table 4.4: Recognition results (% correct symbols) for presegmentation followed by spatial recognition and presegmentation followed by the combined recognition.

4.3.2 Training Recognition Systems Based on Predicted Credibility

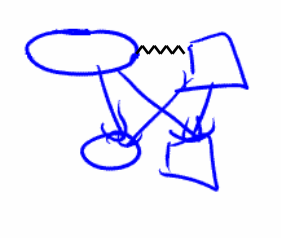
Method

Rather than run each system on a full sketch and combine the results as above, this method of combination narrows the focus of each component recognition system and considers each system to be an expert on its new, narrowed focus. Confusion matrices, either actual or estimated with distance metrics, determine which symbols each system should concentrate on.

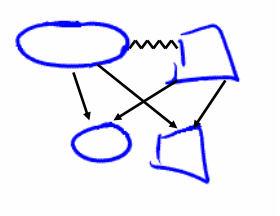
First each symbol is assigned to a representation and corresponding recognition system based its confusability with other symbols in the domain. This assignment is done in a greedy way, with the least confusable symbol among all of the representations assigned to the representation under which it is most distinct, and so on, until all symbols have been assigned.

The component systems are adapted by selecting training data for the spatial and temporal systems to include only those symbols assigned to the corresponding representation, and by adjusting the input domain descriptions of the conceptual system to include only those symbols assigned to the conceptual representation. In this way, symbols not assigned to a given recognition system are regarded as noise by that system. Once the systems have been trained, they are then applied to a sketch serially, and successive systems consider only unrecognized parts of a sketch, i.e. parts previous systems labeled noise. Figure 4-13 illustrates this process.

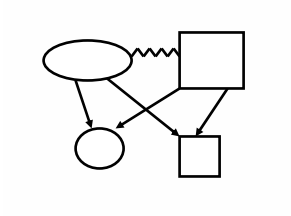
(a) Original sketch of a family tree. Temporal recognition is performed first.



(b) The temporal system recognizes the divorce-link. The remaining blue areas are shown to the conceptual recognizer.



(c) The conceptual system recognizes arrows. The remaining blue area are shown to the spatial recognizer.



(d) The spatial system recognizes ellipses and rectangles.

Figure 4-13: An example of serial recognition. At each stage, black shapes have been recognized and blue areas of the sketch are unrecognized.

Results

Table 4.5 contains recognition results for the method described above. These results are compared with the baseline results, presented in Section 2.3.

	Family Tree	Flow Chart	Circuit
Confusion Matrices	51.9	48.7	51.2
Distance Metrics	50.4	46.1	49.2
Baseline	47.3	37.3	49.9

Table 4.5: Recognition results (% correct symbols) for serial combination of recognition systems.

This method of combination does provide some improvement over our baseline (the best single representation system for each domain); however, these results suggest that

this combination method is not as strong as the previous parallel method. However, we have adapted existing recognition systems that were not originally designed to function in this way (i.e. to take partially recognized sketches as their inputs). A better test of this method would be to combine purposefully designed component systems.

Chapter 5

Related Work

We discuss work related to this thesis in three groups. We begin by positioning this work among other research on pen-based applications and sketch recognition. We then discuss related work on representations, specifically the use of multiple representations for pen-based data. Finally, we describe work related to our methods of combining recognition systems based on different representations.

5.1 Sketch Recognition

Early pen input devices for computers took the form of light pens, such as the Sketch-Pad system developed by Sutherland in 1963 [62]. This form of interaction actually predates the computer mouse [14], yet pen input devices did not see the same commercial success as the mouse, which has since become ubiquitous. However, increasingly powerful processors and ever shrinking hardware have recently lead to several commercial pen-input products, including the Palm Pilot and the Tablet PC, and have renewed interest in pen-based interaction.

Current research in pen-based computer interaction ranges from that concentrated primarily on usability and user interfaces, such as [34, 68], to work focused on sketch recognition, such as this work, which does not consider how the end user may ultimately interact with the computer. Research on sketch recognition can be placed along a spectrum as well, determined by the degree of segmentation performed and

the degree of drawing flexibility permitted. A sketch application may be based on single stroke gestures [50, 39], may classify isolated symbols with any number of strokes [26, 69], or may perform segmentation with varying assumptions about how symbols may be drawn, whether they must be drawn with a set number strokes, or whether they may be interspersed [55, 4, 18, 59]. In this discussion, we refer to any system that assigns labels to isolated symbols as a classifier and one that locates symbols with in a sketch and assigns labels as a recognizer. While this work focuses on the later, performing segmentation on minimally constrained drawings, it is related to both existing classification and recognition techniques.

5.2 Representations

5.2.1 Knowledge Representation

This thesis is based on the idea that sketch data may be represented in several ways and that the different lenses of these multiple representations supply a variety of perspectives on the problem of sketch recognition. This in turn is important because changing perspectives can change a problem's difficulty. This premise is very similar to ideas proposed in the study of knowledge representations. Levesque and Brachman note a trade off between expressiveness and tractability, but also contend that one formalism cannot be "better" than another, but rather each comes with a tradeoff [38]. Considering diagrammatic representation specifically, several works have suggested that solving a problem is largely a matter of representing it in the best way [22, 35]. Davis et al. note that different representations focus attention in different areas and lend themselves to different inferences, and the authors warn against attempting to force a problem into representation that does not suit it [12]. Similarly, this thesis suggests that one sketch representation alone is not sufficient for all problems or domains.

5.2.2 Multiple Representations in Sketch Recognition

This thesis considers three representations for sketches, which we have called spatial, temporal, and conceptual. While this nomenclature is original, various forms of these three classes of representations have been used for both sketch recognition and symbol classification.

Spatial

Many of the spatial approaches to sketch processing have been developed for isolated symbol classification, rather than the recognition of full sketches. These include work by Kara and Stahovich, which uses pixel-based, image similarity measures to match incoming sketched symbols to templates [31] and work by Apte et al.[7] and Fonseca et al.[15], which each classify shapes based on global spatial features such as the shape of the bounding box and length of the perimeter of a shape.

In addition to the work by Oltmans [40], Gennari, Kara, Stahovich, and Shimada take a spatial approach to the problem of sketch recognition [18]. Their approach combines the isolated symbol classifier of Kara and Stahovich with a spatial segmentation, which relies on changes in ink density, to recognize entire sketches.

Temporal

Many symbol classification schemes and most on-line sketch recognition systems use at least some temporally based information. For example, timing information is required in order to orient symbols with pen direction before applying a spatial approach [40, 18] or to use pen speed to locate corners and segment strokes as the base elements of a conceptual approach [4]. While only Sezgin and Davis have used a primarily temporal representation for sketch recognition [55, 56], there do exist temporally based methods for other pen-based interaction tasks including curve refinement [60] and isolated symbol recognition [6].

Conceptual

Yin and Sun [69] take a conceptual, parts-based approach to isolated symbol classification, by defining a sketched symbol in terms of line and arc primitives and topological relationships among those primitives, including intersection and tangency. However, they do not define higher level conceptual objects.

In addition to the recognition system of Alvarado, which this work builds upon [4], Shilman et al. [58] and Deufemia and Risi [13] also each describe a conceptual recognition approach. Both define languages for sketch parsing and recognition, which are similar in concept to the LADDER language developed by Hammond [24].

5.2.3 Multiple Representations in Handwriting Recognition

Handwriting recognition is perhaps the most well studied and commercially successful area of pen-input research. Though more constrained in some ways, handwriting recognition systems share much with sketch recognition approaches, including similarities in representations. We find handwriting recognition particularly relevant for this thesis because some prior work has been done on combining representations of handwriting.

Plamondon and Srihari review some of the large body of work on handwriting recognition, and divide the field into online and offline approaches [45]. Offline approaches have only an image available, and thus tend to represent input spatially. Senior and Robinson [54] describe a representation that includes a histogram of ink density and is similar to the approach taken by Oltmans [40]. Online handwriting recognition methods operate on input created with digital pens. Hu et al. represent handwriting as a time-ordered sequence of stroke segments and use a Hidden Markov Model-based method for recognition[27]; this is similar to the sketch representation and methods used by Sezgin and Davis [56, 55].

In work most related to this thesis, Alimoglu and Alpaydin combine standard online and offline approaches for the problem of handwritten digit recognition [2]. Their work is similar to ours in that they define two representations , which they

call "static" and "dynamic." They find that some of the errors made using each representation alone are uncorrelated and report an improvement in accuracy when the two are combined.

5.3 Combining Representations

5.3.1 Combining Classifiers and Recognizers

Multiple classifier systems are the subject of much research, and it is widely recognized that a combination of classifiers or experts is generally preferred to a single opinion[48, 51, 33]. In spite of a similar premise, much of the existing work in multiple classifier systems differs from our problem in several important ways. First, the output of most component classifiers used in combination is relatively simple, for example a class label for a given instance of data. The output of the recognition systems that we are examining is a labeled sketch, which may contain many labeled subcomponents as well as their locations. Second, much of the research in this field is conducted on large, standard data sets, and our sketch data sets are quite small by comparison, preventing the use of many standard methods that require extensive training. Despite these differences, several questions in the field of multiple classifier systems are relevant for us as well.

Combination Topology

One relevant area of research in classifier combination concerns the topology of the combination, which may be parallel, serial, or a mix of the two. In this thesis we have presented both parallel and serial methods of combination. Rahman and Fairhurst [47] prefer a hybrid method; however Rahman et al. [49] acknowledge that an ideal combination may require access to internal structures in the base classifiers, which may not always be possible. Alimoglu and Alpaydin [2] compare combination strategies using multiple classifiers based on different representations of handwritten digits, as described in the previous section. They find that serial, multi-stage cascading is the

best balance between accuracy and efficiency. In their approach, a simple scheme handles most cases and difficult cases are passed to a more complex classifier. Stolfo et al. use a parallel approach to combining the outputs of multiple speech recognizers [61]. This work is related to ours in that the constituent recognizers act on full utterances, rather than isolated words. They note that one of the key problems is determining the granularity of the combination scheme (in their case the choice is between phonemes and words).

Confidence Measures

Chapter 4 presented domain and symbol complexity measures and used the measures to judge the reliability of a recognition system's output. This question of how much to trust the output of each constituent recognizer or classifier has also been examined previously. Bengio et al. discuss how to measure confidence in several different identity verification methods and create a multimodal identity verification scheme by weighting individual methods based on this degree of confidence [8]. Hao et al. test several methods of computing confidence scores for diverse classifiers on the problem of handwritten digit recognition [25].

Improving Black Box Systems

In this work, we have built upon existing, well developed recognition systems, modifying them relatively little and treating them as near "black boxes", which limits how they may be combined, as internal steps are not accessible. Rahman et al. consider a similar problem: how to improve a general purpose, commercial speech recognizer [49]. They post-process the results of a single commercial speech recognition system to achieve a better final ranking of possible utterance labelings for a specific user or environment.

5.3.2 Distance Measures

A key component of this thesis is judging the credibility of a given representation/system on a particular domain or symbol in order to combine systems most effectively. Section 4.2.2 introduced three distance metrics (spatial, temporal, and conceptual) to calculate the similarity of two sketched symbols and gauge their confusability. These methods are related to several existing applications of distance measures for the comparison of simple images and handwriting. Calculating the similarity of an image or a handwritten character to templates or to members of a training data set is a common method of classifying such data. The key difference between the application of distance measures in this thesis and exiting applications is that in this work, distance measures are not used directly for classification or recognition. Instead, we use distance measures to provide guidance for combining separate recognition methods.

Spatial

We define the spatial distance between symbols in terms of distances between edge direction histograms. Similar approaches have been well studied as a means of image comparison. Jain and Vailaya [30] apply edge direction histograms, similar to those used in this work, to the problem of identifying similar trademark images, while Veltkamp and Latecki [65] compare the properties and performance of a number of spatial similarity measures, including edge direction histograms, for image retrieval.

Temporal

We represent a symbol temporally as a sequence of events, which is then encoded as a string. Temporal distances are calculated as string edit distances. Cortelazzo et al. also represent images as strings and apply a string edit distance to determine similarity and identify trademark images [10]. Their strings, however, represent spatial elements of images, rather than temporal elements as in this work, and although their distance measure is very similar to ours, their representation is fundamentally different.

Conceptual

We calculate a conceptual distance between symbols by computing the distance between corresponding graph representations. This application of a distance measure is the most closely related to previous work. Similar graph distance measures have been applied to pen-based data, while the distance measures discussed above were applied to static, pixel-based images.

Sanfeliu and Fu apply a graph distance measure to the problem of classifying hand written English characters [52]. Their graph distance measure is similar to that used in this work; however, nodes and edges represent individual strokes and connections between strokes, while in our work a stroke may correspond to more than one node and edges encode additional relationships between nodes. WeeSan Lee et al. [37] and Seong-Whan Lee[36] each perform sketched symbol classification by matching graphs that contain geometric information and define alternatives to the graph distance measure used in this work.

Chapter 6

Future Work

This chapter discusses three areas of possible future work: developing more integrated combination schemes, using recognitions systems other than those employed in this thesis, and applying a combination of representations and systems to other domains and tasks.

Alternate Models for Combination

The combination methods we have developed employ representations separately, i.e. the constituent recognition systems do not consider all available information and they do not exchange intermediate information, only final labels or segments. Combination schemes that allow more information exchange between representations would be an interesting avenue for future work. Such schemes are particularly relevant for sketch recognition because in many cases, subdecisions affect one another, and improving one subdecision may impact several neighboring decisions and result in a cascading improvement in the final result. For example, changing the interpretation of a stroke from a curve to a line may result in a collection of strokes being reinterpreted as a square, which may then affect the interpretation of a higher level symbol, and if domain knowledge is incorporated, the reinterpretation of a symbol may affect the meaning of a nearby symbol.

One way of allowing this information exchange between representations would be a combination scheme that uses iterative refinement. Each constituent system, based on

a single representation, would evaluate a sketch, share intermediate information with other systems, and then reevaluate based on new information. In this work we have treated existing systems as near black boxes, making relatively few changes to internal structures; however, such an iterative method would require different constituent systems (or significant changes to current systems) since most stand alone systems are not capable of sharing and accepting intermediate information.

A second way of combining information from different representations would be to combine evidence from multiple representations in a single probabilistic model. Several recognition systems, in particular the conceptual system used in this work [4] and a related temporal system [56], use such models; however, these systems ignore some possible sources of information from other representations, which could be includes as observations.

Alternate Implementations of Representations

We have considered only a single recognition system based on each of the three representations (spatial, temporal and conceptual). However, it would be useful to examine different recognition implementations in order to better understand the relationship between a representation and a particular implementation of that representation. In particular, there is a question of which recognition results are due to the implementation and which are due to the underlying representation. A challenge in answering this question is that creating implementations requires significant development time (the three implementations used in this thesis are theses themselves). Furthermore, there are currently few systems which act on the kinds of unconstrained sketches examined in this work.

Other Symbolic Domains and Sketching Tasks

This work examines three symbolic domains (family trees, flow charts and circuit diagrams); however, there are many others to which this work could be applied. For example, military planning diagrams, chemistry molecules, and mechanical or physical systems have been used in other sketch recognition work [17, 42, 3]. Expanding our

work to new sketching domains would provide more reference points to better gauge the strengths and weaknesses of each representation/system. Including more domains with different combinations of characteristics, like the degree of temporal interspersing or pixel overlap between symbols, could help to refine the complexity metrics for domains and symbols in Chapter 4. One challenge to incorporating new domains in this work is data collection. Locating knowledgeable subjects, who are willing to provide sufficient training data, can be difficult for many specialized domains (e.g. military planning diagrams).

In addition, we consider only recognition of symbolic sketches, but there are many other pen-based tasks for which a combination of multiple representation or systems could be useful, for example the beautification of artistic sketches. In particular, the segmentation approaches presented in Chapter 3, which do not use domain knowledge, could be useful as part of a user interface application that groups strokes to facilitate editing or indexing, but does not perform recognition (for example the digital notebook created by Wang et al. [67]).

Chapter 7

Conclusion

This thesis is based on the idea that looking at a problem from different perspectives may make it appear easier or harder. It explores ways of improving recognition accuracy by combing different sketch representations, and recognition systems that are based on those representations. There are four main contributions of this work.

- We have developed spatial, temporal and conceptual approximate segmentation methods, which do not rely on domain knowledge.
- We have applied the segmentation methods to improve recognition accuracy.
- We have introduced confusability as a means of judging the credibility of a recognition system and as a means of combining systems.
- We have developed representation specific measures, which approximate confusability without the need to build systems.

We have presented two methods for combining recognition systems. The first improves recognition by improving segmentation, while the second seeks to predict how well a particular system will recognize a given domain or symbol. We have shown that combining several recognition systems based on different representations can improve the accuracy of existing recognition methods. This work brings us closer to the level of recognition accuracy necessary to apply sketch recognition in a realistic design setting.

Bibliography

[1] Aaron Adler and Randall Davis. Symmetric multimodal interaction in a dynamic dialogue. In *2009 Intelligent User Interfaces Workshop on Sketch Recognition*. ACM Press, February 2009.

[2] Fevzi Alimoglu and Ethem Alpaydin. Combining multiple representations for pen-based handwritten digit recognition. In *Proceedings of the Fourth International Conference on Document Analysis and Recognition*, 1997.

[3] Christine Alvarado. A natural sketching environment: Bringing the computer into early stages of mechanical design. Master's thesis, MIT, 2000.

[4] Christine Alvarado. *Multi-Domain Sketch Understanding*. PhD thesis, Massachusetts Institute of Technology, August 2004.

[5] Christine Alvarado and Randall Davis. Sketchread: A multi-domain sketch recognition engine. In *Proceedings of UIST 2004*, pages 23–32, New York, New York, October 24-27 2004. ACM Press.

[6] Derek Anderson, Craig Bailey, and Marjorie Skubic. Hidden markcov model symbol recognition for sketch-based interfaces. In *AAAI Fall Symposium Series, Making Pen-Based Interaction Intelligent and Natural*, 2004.

[7] Ajay Apte, Van Vo, and Takayuki Dan Kimura. Recognizing multistroke geometric shapes: an experimental evaluation. In *UIST '93: Proceedings of the 6th annual ACM symposium on User interface software and technology*, pages 121–128, New York, NY, USA, 1993. ACM.

[8] Samy Bengio, Christine Marcel, Sébastien Marcel, and Johnny Mariéthoz. Confidence measures for multimodal identity verification. *Information Fusion*, 3:267–276, 2002.

[9] Sonya Cates. Using context to resolve ambiguity in sketch understanding. Master's thesis, MIT, 2006.

[10] G. Cortelazzo, G.A. Mian, G. Vezzi, and P. Zamperoni. Trademark shapes description by string-matching techniques. *Pattern Recognition*, 27(8):1005–1018, August 1994.

[11] Randall Davis. Sketch understanding in design: Overview of work at the MIT AI lab. *Sketch Understanding, Papers from the 2002 AAAI Spring Symposium*, pages 24–31, March 25-27 2002.

[12] Randall Davis, Howard Shrobe, and Peter Szolovits. What is a knowledge representation? *AI Magazine*, 14(1):17–34, 1993.

[13] Vincenzo Deufemia and Michele Risi. A dynamic stroke segmentation technique for sketched symbol recognition. pages 328–335. 2005.

[14] D. C. Engelbart. X-y position indicator for a display system. U.S. Patent 3,541,541, 1970.

[15] Manuel J. Fonseca, Csar Pimentel, and Joaquim A. Jorge. Cali: An online scribble recognizer for calligraphic interfaces. In *Sketch Understanding, Papers from the 2002 AAAI Spring Symposium*, pages 51–58, 2002.

[16] Kenneth Forbus, R. Ferguson, and J. Usher. Towards a computational model of sketching. In *Proceedings of QR2000*, 2000.

[17] Kenneth D. Forbus, Jeffrey Usher, and Vernell Chapman. Sketching for military courses of action diagrams. In *IUI '03: Proceedings of the 8th international conference on Intelligent user interfaces*, pages 61–68, New York, NY, USA, 2003. ACM.

[18] Leslie Gennari, Levent Burak Kara, Thomas F. Stahovich, and Kenji Shimada. Combining geometry and domain knowledge to interpret hand-drawn diagrams. *Computers and Graphics*, 29(4):547–562, 2005.

[19] E. Goldmeier. Similarity in visually perceived forms. In *Psychological Issues*, volume 8:1, 1972.

[20] A. L. Gorin, B. A. Parker, R. M. Sachs, and J. G. Wilpon. How may I help you? *Speech Communication*, 23:113–127, 1997.

[21] Barbara Grosz and Julia Hirschberg. Some intonational characteristics of discourse structure. In *Proceedings of the International Conference on Spoken Language Processing*, pages 429–432, 1992.

[22] Jungpil Hahn and Jinwoo Kim. Why are some diagrams easier to work with? effects of diagrammatic representation on the cognitive integration process of systems analysis and design. *ACM Transactions on Computer-Human Interaction*, 6(3):181–213, 1999.

[23] Tracy Hammond and Randall Davis. Ladder, a sketching language for user interface developers. *Computers and Graphics*, 28:518–532, 2005.

[24] Tracy Anne Hammond. *LADDER: A Perceptually-based Language to Simplify Sketch Recognition User Interface Development*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, January 2007.

[25] Hongwei Hao, Cheng-Lin Liu, and H. Sako. Confidence evaluation for combining diverse classifiers. In *Seventh International Conference on Document Analysis and Recognition*, pages 760–765, 2003.

[26] Heloise Hse and A. Richard Newton. Sketched symbol recognition using zernike moments. In *ICPR '04: Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 1*, pages 367–370, Washington, DC, USA, 2004. IEEE Computer Society.

[27] Jianying Hu, Michael K. Brown, and William Turin. HMM based on-line handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:1039–1045, October 1996.

[28] J.J. Hull. A database for handwritten text recognition research. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(5):550–554, May 1994.

[29] Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. Teddy: A sketching interface for 3d freeform design. *SIGGRAPH 99*, pages 409–416, August 1999.

[30] Anil K. Jain and Aditya Vailaya. Image retrieval using color and shape. *Pattern Recognition*, 29(8):1233–1244, 1996.

[31] Levent Burak Kara and Thomas F. Stahovich. An image-based, trainable symbol recognizer for hand-drawn sketches. *Computers and Graphics*, 29(4):501–517, 2005.

[32] Manolya Kavakli and John S. Gero. Sketching as mental imagery processing. *Design Studies*, 22(4):101–122, 2001.

[33] Josef Kittler, Mohamad Hatef, Robert P.W. Duin, and Jiri Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998.

[34] James A. Landay and Brad A. Myers. Sketching interfaces: Toward more human interface design. *Computer*, 34(3):56–64, 2001.

[35] Jill H. Larkin and Herbert A. Simon. Why a diagram is (sometimes) worth 10, 000 word. *Cognitive Science*, pages 65–99, 1987.

[36] Seong-Whan Lee. Recognizing hand-drawn electrical circuit symbols with attributed graph matching. In *Structured Document Image Analysis*, pages 340–358. Springer-Verlag, 1992.

[37] Weesan Lee, Levent Burak Kara, and Thomas F. Stahovich. An efficient graph-based symbol recognizer. In *EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling*, pages 11–18, 2006.

[38] Hector J. Levesque and Ronald J. Brachman. A fundamental tradeoff in knowledge representation and reasoning (revised version). In Ronald J. Brachman and Hector J. Levesque, editors, *Readings in Knowledge Representation*, pages 41–70. 1985.

[39] A. Chris Long, Jr., James A. Landay, Lawrence A. Rowe, and Joseph Michiels. Visual similarity of pen gestures. In *SIGCHI conference on Human factors in computing systems*, pages 360–367, 2000.

[40] Michael Oltmans. *Envisioning Sketch Recognition: A Local Feature Based Approach to Recognizing Informal Sketches*. PhD thesis, Massachusetts Institute of Technology, May 2007.

[41] Michael Oltmans, Christine Alvarado, and Randall Davis. Etcha sketches: Lessons learned from collecting sketch data. In *Making Pen-Based Interaction Intelligent and Natural*, pages 134–140, Menlo Park, California, October 21-24 2004. AAAI Fall Symposium.

[42] Tom Y. Ouyang and Randall Davis. Recognition of hand drawn chemical diagrams. In *Proceedings of AAAI*, pages 846–851, 2007.

[43] Tom Y. Ouyang and Randall Davis. A visual approach to sketched symbol recognition. In *Proceedings of the 2009 International Joint Conference on Artificial Intelligence (IJCAI)*, 2009.

[44] Charles Peirce. *What is a sign?*, volume 2 of *The Essential Peirce*, chapter 2. Indiana University Press, 1998.

[45] Rejean Plamondon and Sargur N. Srihari. On-line and off-line handwriting recognition: A comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:63–84, January 2000.

[46] A. T. Purcell and J. S. Gero. Drawing and the design process. *Design Studies*, 19:389–430, 1998.

[47] A.F.R. Rahman and M.C. Fairhurst. A study of some multi-expert recognition strategies for industrial applications: Issues of speed and reliability. In *International Conference on Vision Interfaces*, pages 569–574, 1999.

[48] Ahmad Fuad Rezaur Rahman and Michael C. Fairhurst. Multiple classifier decision combination strategies for character recognition: A review. *IJDAR*, 5(4):166–194, 2003.

[49] Fuad Rahman, Yuliya Tarnikova, Aman Kumar, and Hassan Alam. Second guessing a commercial 'black box' classifier by an 'in house' classifier: Serial classifier combination in a speech recognition application. In *Multiple Classifier Systems*, pages 374–383, 2004.

[50] Dean Rubine. Specifying gestures by example. In *Computer Graphics*, volume 25(4), pages 329–337, 1991.

[51] Dymitr Ruta and Bogdan Gabrys. An overview of classifier fusion methods. In *Computing and Information Systems*, volume 7, pages 1–10, 2000.

[52] Alberto Sanfeliu and King-Sun Fu. A distance measure between attributed relational graphs for pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 13(3):353–362, 1983.

[53] Eric Saund, David Fleet, Daniel Larner, and James Mahoney. Perceptually supported image editing of text and graphics. In *Proceedings of UIST '03*, 2003.

[54] Andrew W. Senior and Anthony J. Robinson. An off-line cursive handwriting recognition system. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:309–321, March 1998.

[55] Tevfik Metin Sezgin and Randall Davis. Hmm-based efficient sketch recognition. In *Proceedings of the International Conference on Intelligent User Interfaces (IUI'05)*, pages 281–283, New York, New York, January 9-12 2005. ACM Press.

[56] Tevfik Metin Sezgin and Randall Davis. Sketch interpretation using multi-scale models of temporal patterns. *IEEE Computer Graphics and Applications*, 27(1):28–37, January 2007.

[57] Tevfik Metin Sezgin, Thomas Stahovich, and Randall Davis. Sketch based interfaces: Early processing for sketch understanding. In *The Proceedings of 2001 Perceptive User Interfaces Workshop (PUI'01)*, Orlando, FL, November 2001.

[58] M. Shilman, H. Pasula, and S. Russel R. Newton. Statistical visual language models for ink parsing. In *Proceedings of the AAAI Spring Symposium on Sketch Understanding*, pages 126–32, 2002.

[59] M. Shilman, P. Viola, and K. Chellapilla. Recognition and grouping of handwritten text in diagrams and equations. In *Ninth International Workshop on Frontiers in Handwriting Recognition*, pages 569–574, Oct. 2004.

[60] Saul Simhon and Gregory Dudek. Sketch interpretation and refinement using statistical models. pages 23–32. Eurographics Symposium on Rendering, 2004.

[61] Salvatore J. Stolfo, Zvi Galil, Kathleen McKeown, and Russell Mills. Speech recognition in parallel. In *Human Language and Technology Conference Workshop on Speech and Natural Language*, pages 353–373, 1989.

[62] Ivan B. Sutherland. Sketchpad, a man-machine graphical communication system. *Proceedings of the Spring Joint Computer Conference*, pages 329–346, 1963.

[63] David G. Ullman, Stephen Wood, and David Craig. The importance of drawing in the mechanical design process. *Computer and Graphics*, 14(2):263–274, 1990.

[64] Remko van der Lugt. How sketching can affect the idea generation process in design group meetings. *Design Studies*, 26(2):101–122, March 2005.

[65] Remco C. Veltkamp and Longin Jan Latecki. Properties and performances of shape similarity measures. In *Content-Based Retrieval*, volume 06171, 2006.

[66] Olya Veselova and Randall Davis. Perceptually based learning of shape descriptions. *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI-04)*, pages 482–487, 2004.

[67] Xin Wang, Michael Shilman, and Sashi Raghupathy. Parsing ink annotations on heterogeneous documents. In *Eurographics Workshop on Sketch-Based Interfaces and Modeling*, pages 43–50, 2006.

[68] Xiaogang Xu, Wenyin Liu, Xiangyu Jinand, and Zhengxing Sun. Sketch-based user interface for creative tasks. In *Asia Pacific Conference on Computer Human Interaction*, 2002.

[69] Jianfeng Yin and Zhengxing Sun. *An Online Multi-stroke Sketch Recognition Method Integrated with Stroke Segmentation*, pages 803–810. Lecture Notes for Computer Science. Springer-Verlag Berlin Heidelberg, 2005.

[70] Shane W. Zamora and Eyrun A. Eyjolfsdottir. Circuitboard: Sketch-based circuit design and analysis. In *IUI Workshop on Sketch Recognition*, February 2009.