ANALYSIS OF CHILDREN'S SKETCHES TO IMPROVE RECOGNITION ACCURACY IN SKETCH-BASED APPLICATIONS

A Thesis

by

HONG-HOE KIM

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Tracy Hammond
Committee Members,	Eun Jung Kim
	Erin McTigue
Head of Department,	Hank Walker

December 2012

Major Subject: Computer Science

ABSTRACT

The current education systems in elementary schools are usually using traditional teaching methods such as paper and pencil or drawing on the board. The benefit of paper and pencil is their ease of use. Researchers have tried to bring this ease of use to computer-based educational systems through the use of sketch-recognition. Sketch-recognition allows students to draw naturally while at the same time receiving automated assistance and feedback from the computer.

There are many sketch-based educational systems for children. However, current sketch-based educational systems use the same sketch recognizer for both adults and children. The problem of this approach is that the recognizers are trained by using sample data drawn by adults, even though the drawing patterns of children and adults are markedly different. We propose that if we make a separate recognizer for children, we can increase the recognition accuracy of shapes drawn by children.

By creating a separate recognizer for children, we improved the recognition accuracy of children's drawings from 81.25% (using the adults' threshold) to 83.75% (using adjusted threshold for children).

Additionally, we were able to automatically distinguish children's drawings from adults' drawings. We correctly identified the drawer's age (age 3, 4, 7, or adult) with 78.3%. When distinguishing toddlers (age 3 and 4) from matures (age 7 and adult), we got a precision of 95.2% using 10-fold cross validation. When we removed adults and distinguished between toddlers and 7 year olds, we got a precision of 90.2%. Distinguishing between 3, 4, and 7 year olds, we got a precision of 86.8%.

Furthermore, we revealed that there is a potential gender difference since our recognizer was more accurately able to recognize the drawings of female children (91.4%) than the male children (85.4%).

Finally, this paper introduces a sketch-based teaching assistant tool for children, EasySketch, which teaches children how to draw digits and characters. Children can learn how to draw digits and characters by instructions and feedback.

DEDICATION

I would like to thank for my family: my father Jongkyu Kim, my mother Jeonghee Yoon, my brother Sangmin Kim, and my loving wife Jihee Park and Isabel Hyunji Kim. Also thank you for Jesus.

ACKNOWLEDGEMENTS

Thank you for my advisor Dr. Hammond, as well as to my committee member Dr. Kim and Dr. McTigue. Also I would like to thank for the sketch recognition lab members: StephanieValentine for helping me in the development of the recognizer, Francisco Vides for assistance with the user interface and the agent system, and Wenzhe Li for his advice on feature selection.

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

People have been using paper and pencil for many years for various purposes. Using these tools we have recorded our history, communicated with each other through letters, and passed down our knowledge to subsequent generations. Writing, and more generally sketching, has been one of the most important methods of communicating in the history human society.

With modern technology, sketching and handwriting have found new purposes as a form of human-computer interaction. We sign our names on tablets when we buy something using our credit cards at grocery stores. We jot down memos using our smartphones. We have even begun to use intelligent sketch-recognition based systems in computer interfaces for a broad range of purposes, from education, to design, to engineering . The benefit of these systems that support intelligent recognition of hand drawn sketches is that they allow people to draw and interact freely and naturally. Without sketching technology, we are limited to mouse and palette shapes that limit our creativity, and they also limit our domain because systems cannot provide all the domains we want to use. Sketching alleviates these limitations and also contributes to our creativity.

We are particularly interested in the use of sketch systems in education. There are a many sketch-based educational systems [29,58] that recognize children's sketches and teach them how to draw shapes. However, the limitation of these systems is that the recognizers are trained by adults' drawings. It is clear that the drawing patterns of children are markedly different from adults', and therefore we should make a separate sketch recognizers specifically built for use by children. Through the collection of sketches of ten children aged 3-7 and the analysis of the differences between adults and children, we were able to 1) create a separate recognizer for children, and 2) automatically determine a sketcher's age by their drawing features.

To analyze the children's sketches, we collected 475 sketches of simple shapes, digits, and characters drawn by ten children (seven 7 year-olds, two 4 year-olds, and one 3 year-olds) and four adults. To increase the accuracy of children's data, we analyzed the optimal resampling points for children and we found that different resampling point (32 points) increased recognition accuracy from 81.25% (a threshold for adults) to 83.75%. Using subset selection, we found that different feature sets dominate recognition accuracy for children and adults. We analyzed the optimal threshold for children and adults using forty-four features introduced by Paulson et al. [31], and we found that the shape recognition accuracy for children was dominated by "Percentage of strokes that passed the line test". However, "The perimeter (of bounding box) to stroke length ratio" gave the best recognition result for adults. Additionally, we found that different gender seems to have different sketch-recognition accuracy. During our user study, the female children gave more accuracy (88.5%) than the male children (77.8%).

Analysis of this data showed that there are several differences between adults' and children's sketches. Using these distinguishing features, we could automatically recognize their ages by their drawings. We were able to determine the drawer's age (age 3, 4, 7, and adult) with a precision of .783, recall of .768, and an f-measure of .773. When we compared between toddlers (age 3 and 4) and matures (age 7 and adult), we got a precision of .952, recall of .957, and an f-measure of .952 using 10-fold cross validation. When we just distinguished children's ages (age 3, 4, and 7) removing adults, we got a precision of .868, a recall of .874, and an f-measure of .862. Distinguishing between toddlers and 7-year olds, we got a precision of .902, a recall of .902, and an f-measure of .902.

This paper also introduces a sketch-based educational system, EasySketch, which teaches children how to draw digits and characters. To help children learn more easily, EasySketch has an agent system that acts as a virtual instructor for children. The agent system gives feedback whenever children draw their sketches and gives different feedback based on their correctness history. To keep the interest of children, we also used the sound system that gives different feedback sounds by their correctness history.

2. RELATED WORK

In this chapter, we examine sketch-recognition approaches and sketch-recognition based applications are examined.

2.1 Sketch recognition

In this section, we describe sketch recognition approaches and useful features for sketch recognition. Over many years of research, different approaches were proposed for sketch recognition: (1) the statistical feature-based approaches, (2) the gesture based approaches, and (3) the geometric approaches.

The first approach is the statistical feature-based approach. This approach analyzes the feature values from strokes. After retrieving the features, the recognizer determines the shapes by classifiers such as linear classifier or naïve Bayes. The examples are Rubine [33] and Long [25].

The second approach is the gesture-based approach. The gesture-based approach compares the input data with data sets, which we already have. This approach tries to recognize the best matched shapes for the input data. To compare each shape, the approach calculates the distances between input shape and the shapes in the data sets.
Many researchers developed their algorithms with this approach. The examples are (1) the elastic structure matching, which uses dynamic programming [12], (2) $1 and $N recognizer by Wobbrock et al. [9, 46], and (3) the vision-based recognizer by Kara et al. [19]. Our system is based on the vision-based recognizer. The benefit of the gesture- based approach is

that the approach does not need to represent rules of primitives in shapes. However, the template-matching approach needs many data sets, which consumes many times when the recognizers compare the input data with data sets.

The third approach is the geometric approach. This approach describes each shape by representing rules of primitives. There have been various studies that implement the geometric approaches. One of the examples is the LADDER system by Hammond et al. [18]. The benefit of the geometric approach is that it can enlarge recognizable shapes by representing more rules of primitives in each shape. The approaches need to have accurate rules for representing primitives in shapes.

The following sections describe in more detail about these three approaches.

2.1.1 *Review of Statistical Feature-Based Approaches*

In this section, we explain various statistical feature-based approaches and useful primitives to describe shapes.

2.1.1.1 Rubine and Long

Rubine introduced GRANDMA, a toolkit for rapidly adding gestures to direct manipulation interfaces [33]. Rubine defined a set of features that could be extracted from a stroke. The features could be used in a linear classifier to perform shape recognition.

Long et al. improved on Rubine's feature set by removing correlated features and adding several new features [25].

2.1.1.2 Writing Angle

Writing angle is widely used for sharp point detection. The sharp point is a point where angle difference is larger than some threshold. The writing angle can detect a point that should divide a shape into two shapes.

The idea of writing angle is comparing angles between three points with other three points. If the difference of angles is larger than some threshold, the middle point in the angle is a sharp point.

The equation to calculate the angle between three points is as follows:

$$a = (p1.x - p2.x, p1.y - p2.y) \tag{2.1}$$

$$b = (p1.x - p3.x, p1.y - p3.y) \tag{2.2}$$

$$\theta = \arccos\left(\frac{a \bullet b}{|a||b|}\right) \tag{2.3}$$

p1 is the middle point between *p2* and *p3*, and θ represents an angle between *p1* and *p3*.

Figure 2.1 shows an example of writing angle.

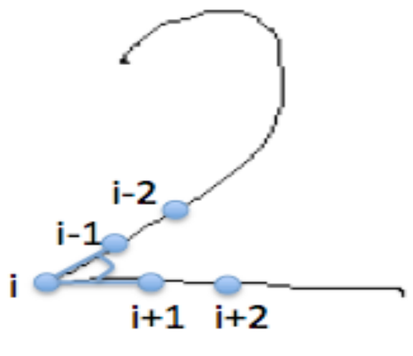


Figure 2.1. Writing angle *i* is the sharp point.

2.1.1.3 Number of Strokes

Number of strokes can be used as a basic primitive for recognizers. For example, a triangle should have three lines (strokes). The benefit of number of strokes is that it can be easily recognized. However, if the shapes have many variances, the number of strokes can contribute to variance differences. Because of each person's writing style, the same character can have a different number of strokes.

Figure 2.2 shows different forms to represent digits. Ward and Kuklinski [45] explained that the symbol displayed on the left in Figure 2.2 is common for North Americans, while the symbol on its right is widely accepted in Europe.

To solve the problem, Tapia proposed an algorithm that makes additional rules for different number of strokes, or the algorithm reduces the stroke variances by enforcing a fixed number of strokes per character, e.g. if the number of strokes M exceeds the given number N, the N-1st to M*th* strokes are concatenated [37].

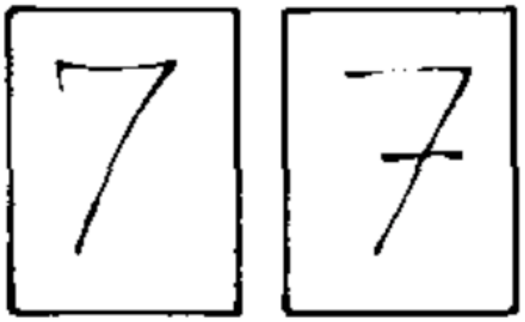


Figure 2.2. Different forms of strokes (from [45]).

2.1.1.4 Point Density

Point density recognizes shapes by analyzing distributions of points in particular areas. We will use the character “o”, “p”, and “b” as examples.

- The “o” is evenly distributed in the whole areas.
- The “p” has more distribution on the higher part than the lower part.
- The “b” has more distribution on the lower part than the higher part.

To compute the distribution, a bounding box is divided vertically into three parts: the upper 40%, the middle 20%, and the lower box of remaining 40%. The reason why the bounding box is divided into three rather than two is due to variances in hand-writing.

Figure 2.3 shows an example of the distribution of the character “p”. Oltmans’s Bulls-eye uses this feature to recognize shapes [28]. However, the limitation is that this feature can represent limited numbers of shapes and it cannot represent detailed shapes.

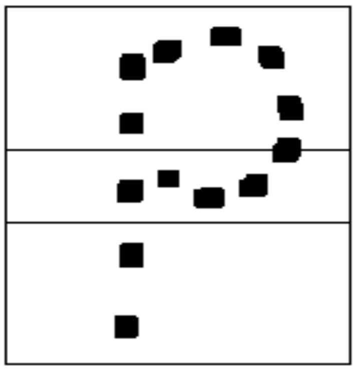


Figure 2.3. Letter “p” has more density in higher part then other parts.

2.1.2 Review of Gesture-Based Approaches

This section explains various gesture-based recognition approaches.

2.1.2.1 Elastic Matching

Chan proposed the elastic matching algorithm that recognizes shapes by finding minimum distances between two shapes [12]. The algorithm calculates each distance by comparing two continuous points in shapes. The mapping between the two sequences of points allows for both one-to-one and many-to-one correspondences between points. By using dynamic programming, there is no need for each shape to have the same point size.

The total distance between the *ith* point of the unknown character and the *jth* point of the model, D(*i*,*j*), is calculated as:

$$D(i,j) = \sigma(i,j) + \left\{ \begin{array}{ll} \sum_{k=0}^{j-1} \delta(0,k) & if\ i = 0 \\ \sum_{k=0}^{i-1} \delta\,(k,0) & if\ j = 0 \\ min \left\{ \begin{array}{c} D(i-1,j) \\ D(i-1,j-1) \end{array} \right\} & if\ i > 0, j = 1 \\ min \left\{ \begin{array}{c} D(i-1),j) \\ D(i-1,j-1) \\ D(i-1,j-2) \end{array} \right\} & if\ i > 0, j > 1 \end{array} \right\} \tag{2.4}$$

$$\delta(i,j) = (x_i - x_j)^2 + (y_i - y_j)^2 + C|\phi_i - \phi_j| \tag{2.5}$$

The benefit of this approach is that it can be expended to other data set easily and the high accuracy (98.60% for digits). Figure 2.4 shows an example of elastic matching algorithm.

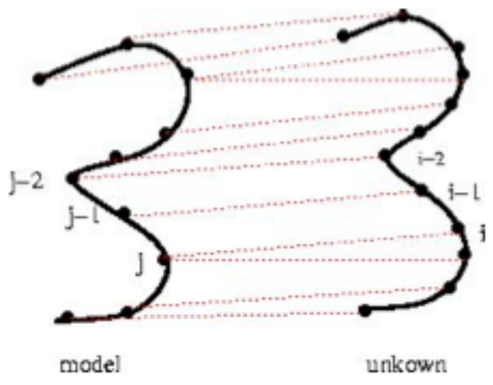


Figure 2.4. Elastic matching (from [44]).

2.1.2.2 $1and $N recognizer

Wobbrock introduced the $1 recognizer (for one-stroke shapes) [46] and the $N recognizer (for multi-stroke shapes) [9].

The $1 recognizer includes preprocessing steps for the raw data. The raw data has several issues as follows:

- The number of points can be different by writing speed ratio and devices used.
- The users can have different experiences with the devices.

To alleviate these problems, the preprocessing steps in the $1 and the $N recognizer include four steps: (1) resampling points, (2) rotating once based on the "indicative angle", (3) scaling and translate, and (4) finding the optimal angle for the best score [9].

The difficulty of the multi-stroke is that it can have various kinds of stroke orders and start, end points. Figure 2.5 shows an example of multi-strokes. The “X” has two strokes and each stroke has two end points. To compare the shapes, the $N recognizer translates the multi-stroke shape into the one-stroke shape (Figure 2.6).

The accuracy of the $N recognizer is about 80-90% [9]. However, the limitation of the $N recognizer is that there are 2^N combination for N strokes. The limitation is that the recognizer takes exponential time to recognize shapes, and it can limit the symbols for the recognizer. For example, the recognizer cannot be used to determine Japanese and Chinese characters, which contain multi-strokes.

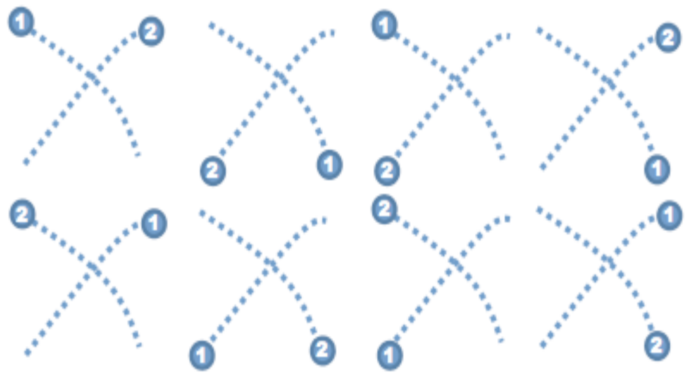


Figure 2.5. Letter “X” have 8 different samples (from [9]).

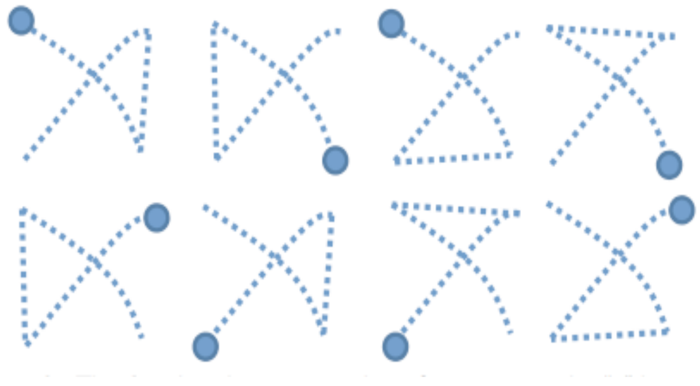


Figure 2.6. Translated “X” character (from [9]).

2.1.2.3 Image-based (vision-based) recognition

Kara introduced an algorithm that uses the vision (image)-based recognition [19]. The problem of prior gesture-based algorithms [9, 12] is that the algorithms have difficulty if the shapes have different stroke orders and different start, end points.

The vision-based algorithm converts the shapes into fixed pixel and compares the unknown shapes with data sets (Figure 2.7). The procedure includes three steps: (1) preprocessing, (2) polar analysis, and (3) template-matching with four measurements (the Hausdorff distance, the Modified Hausdorff distance, the Tanimoto coefficient, the Yule coefficient). We implemented the algorithm for our application. However, the limitation of this algorithm is that the time complexity from several template-matching equations is high.

To solve the problem, we compared the four measurements and found that the single measurement (the Tanimoto coefficient) gives the best accuracy. We will explain our recognition accuracy in Section 4 and 5.

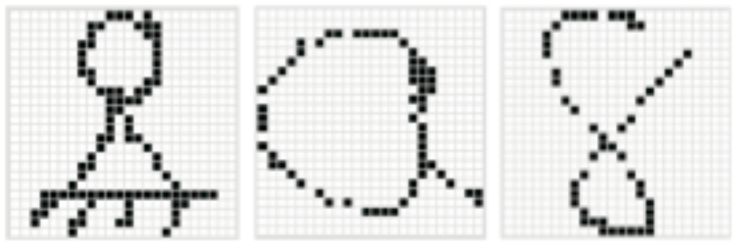


Figure 2.7. Examples of symbol templates (from [19]).

2.1.3 Review of Geometric Approaches

This chapter introduces several geometric approaches.

2.1.3.1 Pictorial pattern using a formal language

A. C. Shaw first explained pictorial patterns using a formal language: picture description language (PDL) [35]. PDL describes how primitives are connected. In PDL, each primitive has two connection points, i.e., a head and a tail. PDL uses four binary operators, denoted by +, -, ×, and *. Additionally, the unary operator ~ is used to express the reverse of a head and a tail of a primitive or a PDL expression. Figure 2.8 shows how primitives are connected using these four binary operators.

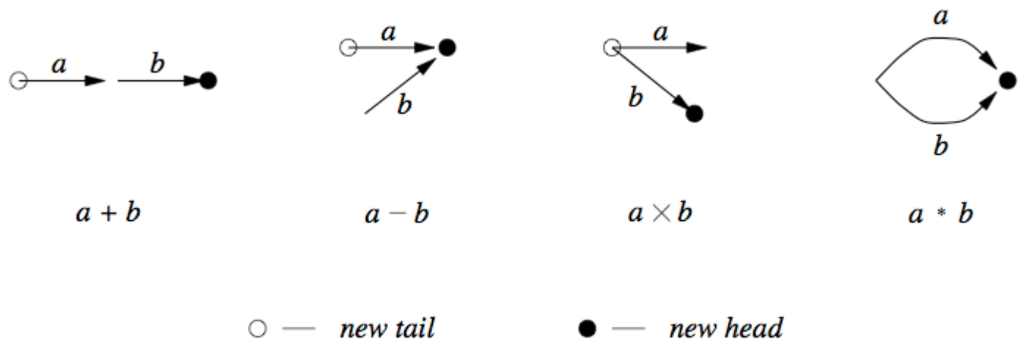


Figure 2.8. The figures show how to express four binary operators in PDL.

Figure 2.9 shows two examples, which describe the structure of the two characters, ‘F’ and ‘P’.

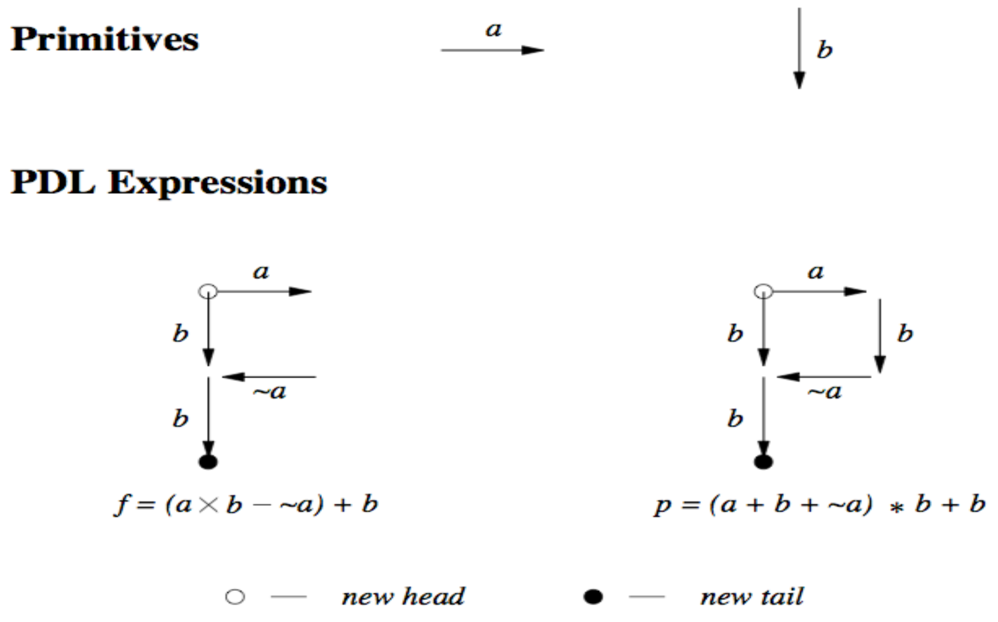


Figure 2.9. Two examples of the character “F” and “P”.

The advantage of PDL is that the pictorial patterns can be described by strings. However, the disadvantage of PDL is that the pictorial patterns can be expressed in different ways.

2.1.3.2 Plex grammar

To solve this ambiguity, J. Feder introduced the plex grammar [15]. To overcome the limitation of PDL, the plex grammar allows more than two connection points. A plex structure has three components:

- A list of n-attaching point entities (simply, napes),
- A list of internal connections between napes,
- A list of attaching points, which can be used for joining the plex structure with other napes or the plex structures.

Figure 2.10 exemplifies how the plex structure expresses the character “F” and “P”.

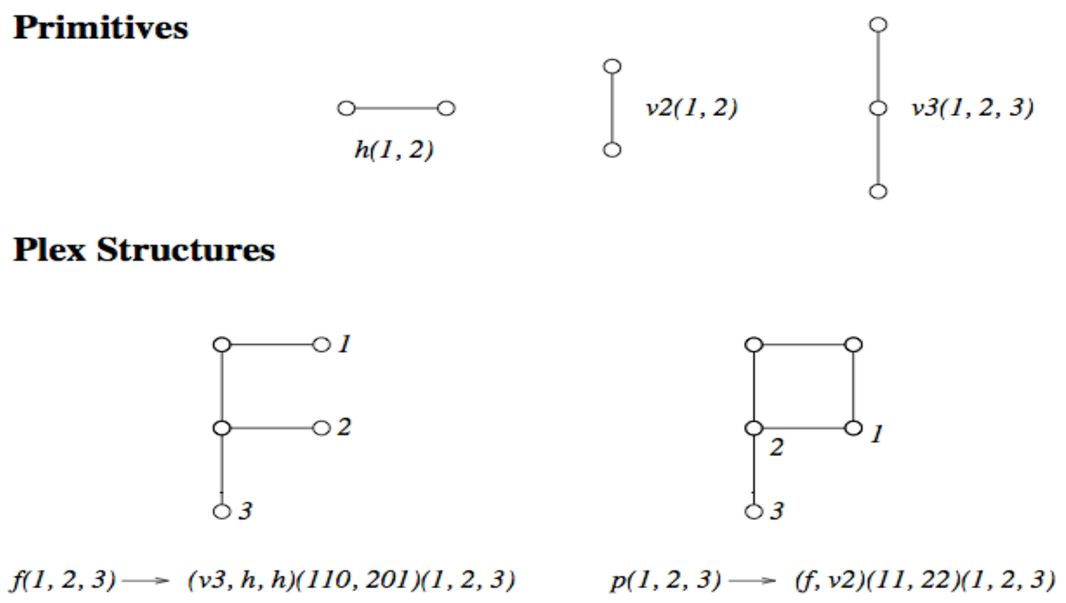

Primitives
h(1, 2)
v2(1, 2)
v3(1, 2, 3)
Plex Structures
1
2
3
2
1
3
f(1, 2, 3) → (v3, h, h)(110, 201)(1, 2, 3)
p(1, 2, 3) → (f, v2)(11, 22)(1, 2, 3)

Figure 2.10. Two examples of the character “F” and “P” using the plex structure.

In primitives, the “h” means a horizontal line and respectively the “v” means a vertical line. The digits show how many end points are in a line. However, the plex grammar also has the same problem with PDL [35]. The expressions can also be expressed several ways.

2.1.3.3 Freeman’s chain code

Freeman’s chain code is a widely used method for representing shapes [17]. The chain code uses the direction of each line. Chan et al. used the chain code for their elastic structural matching algorithm [12]. The chain code has eight values from 0 to 7, which indicates the direction from the current point to the next point. Figure 2.11 shows how the direction values are expressed in Freeman's chain code. The following formula describes the calculation of direction.

$$Direction = \lfloor ((((angle \times 16)/(2 \times \pi)) + 1)\%16)/2 \rfloor \qquad (2.6)$$

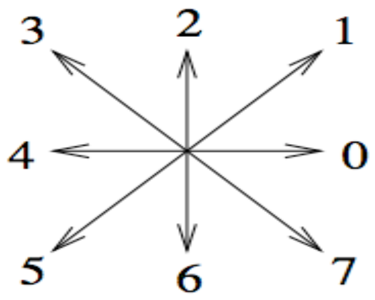


Figure 2.11. Directions in Freeman's chain code.

Using the directions, Chan et al. [12] described five types of primitives:

- line
- up (curve going counter clockwise)
- down (curve going clockwise)
- loop (curve joining itself at some point)
- dot (a very short segment)

For example, Figure 2.12 shows an example of the digit "3". The letter has two down primitives.

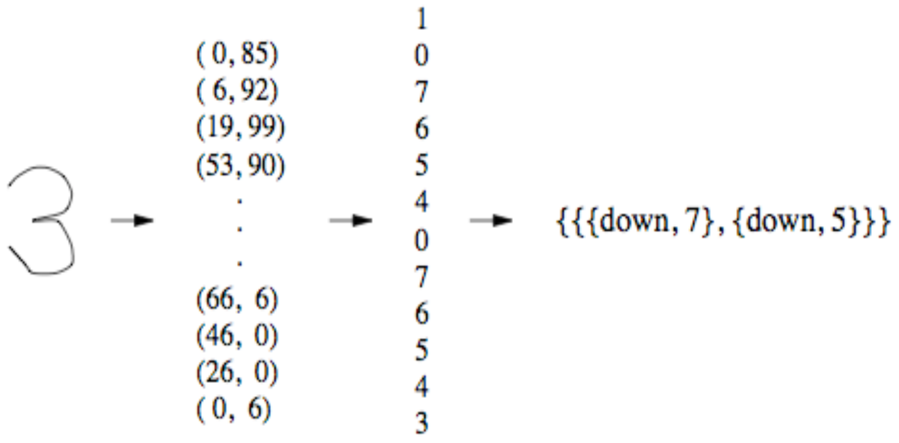


Figure 2.12. Representation of the digit “3” (from [12]).

The limitation of this approach is that it has difficulty describing all the variances of each shape. For example, Figure 2.13 shows an example of variance of the digit “3”. The digit “3” in Figure 2.13 has two down primitives and one loop primitive, which are different primitives from Figure 2.12. Figure 2.14 shows the other variances of the digit “3”. To recognize the shapes more accurately, we need to collect all the characters’ variation and specify primitives in shapes. However, people have many different kinds of drawing styles. Describing all the variance can be time-consuming.

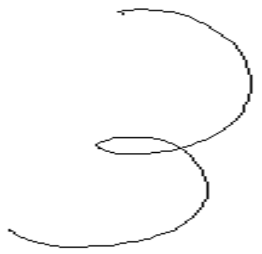


Figure 2.13. An example of variation of digit “3”.

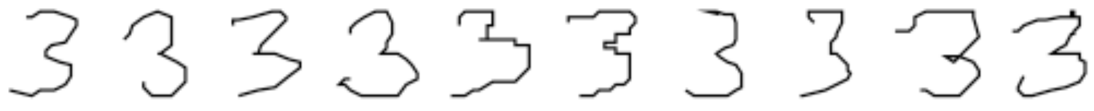


Figure 2.14. Number “3” has 10 different models (from [43]).

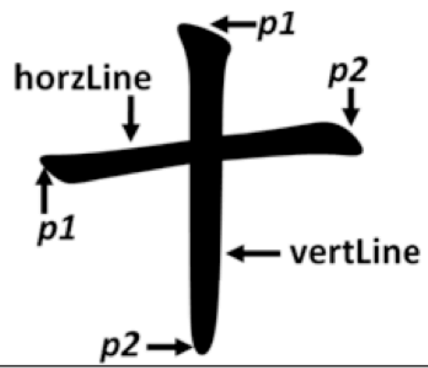
2.1.3.4 LADDER Sketching Language

Hammond et al. introduced LADDER sketching language, which describes “A **L**anguage to **D**escribe, **D**isplay, and **E**diting in Sketch **R**ecognition (LADDER) [18].

A general purpose of LADDER is to describe how sketch diagrams for various domains are drawn, displayed, and edited. To recognize users’ input data, LADDER recognizes the primitives of input data such as a line or an ellipse. After recognizing the primitives, LADDER represents geometric rules for shapes.

There are applications that use LADDER for their recognition system. For example, Paul et al. implemented an educational system for teaching how to draw East Asian character sets [36]. To teach how to draw the characters, the system represents geometrical constrains per each character.

Figure 2.15 represents the Chinese character ten. The components field shows that the character ten should have two lines: a horizontal line and a vertical line. The constraints and aliases fields represent the geometric rules for the two lines.



Name:

TenKanji

Components:

Line	horzLine
Line	vertLine

Constraints:

Horizontal	horzLine	
Vertical	vertLine	
LeftOf	horzLine.p1	horzLine.p2
Above	vertLine.p1	vertLine.p2
EqualLength	horzLine	vertLine
SameX	horzLine.center	vertLine.center
SameY	horzLine.center	vertLine.center

Aliases:

Point	horzLine.p1	leftPoint
Point	horzLine.p2	rightPoint
Point	vertLine.p2	bottomPoint
…		

Figure 2.15. A shape description for the Chinese character ten (from [36]).

2.2 Related applications

In this section, we describe several sketch-based applications that teach mathematics and engineering course. Additionally, we will also introduce educational systems for children and describe the limitation of these systems.

2.2.1 Applications for recognizing mathematics equations

LaViola et al. [23] introduced a sketch-based application for mathematical expression called MathPad2. Users can draw mathematic equations and diagrams on sketch panels. The application also provides editing gestures such as a lasso, an eraser, and so on. The essence of mathematical sketches comes from making association between mathematical expressions and diagrams by using coordinate positions. Association between mathematical expression and diagrams can be made two ways: an implicit way and an explicit way:

The implicit association means that system can associate variables and the variables' nearest diagram. The implicit association is based on the familiar variables and constant variables found in mathematics and physics text illustrations.

By the way, the users can make explicit expressions by drawing a line through a set of related mathematical expressions. However, the problem of MathPad2 is structure recognition. Users need to explicitly select a set of strokes comprising a single mathematical expression by drawing a lasso.

MathBrush [22] also recognizes handwritten mathematical expression. The system transforms hand drawn mathematics equations into MathML [11], which is a mark-up language that represents the underlying mathematical expression and that can be used to transmit the expressions to the backend Computer Algebra Systems (CAS) such as Maple [6] or Mathmatica [7].

2.2.2 Application for engineering students

Mechanix [41] is an educational system for engineering students, which have been using at Texas A&M University and LeTourneau University. Students and instructors can draw their trusses and free-body diagrams, and other shapes on sketch pads. The benefit of Mechanix is that the system gives beneficial feedback to students when the students want to check if their sketches are correct. Another benefit of Mechanix is that the system automatically scores students' assignments.

The recognizer of Mechanix uses a geometrical approach such as LADDER [18]. Most of shapes in engineering class include many numbers of lines. For example, an arrow has one horizontal line and two small lines. By using this approach, Mechanix can recognize many kinds of shapes for engineering course.

2.2.3 Applications for children

We have many applications to educate children. Generally, applications for children use games because children can be boring when they learn. For example, IXL [3] and Sheppard Software [4] have divided sections per each age from pre-kindergarten to eighth grade. The applications teach basic mathematics by showing animation, and children can select the correct answers by clicking buttons. When children choose wrong answers, they can see the explanations for the questions.

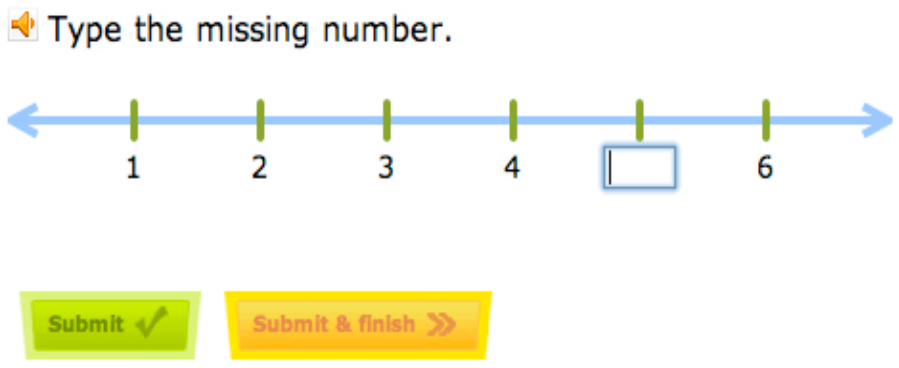


Figure 2.16. An example of IXL.

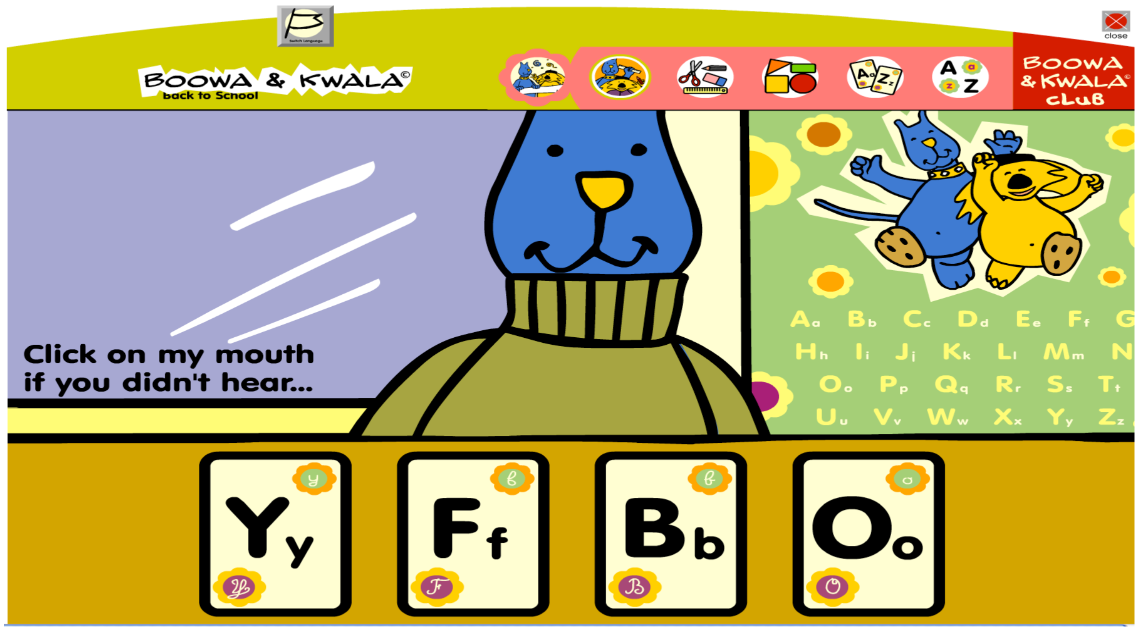


Figure 2.17. Teach characters (the Boowa& Kwala application).

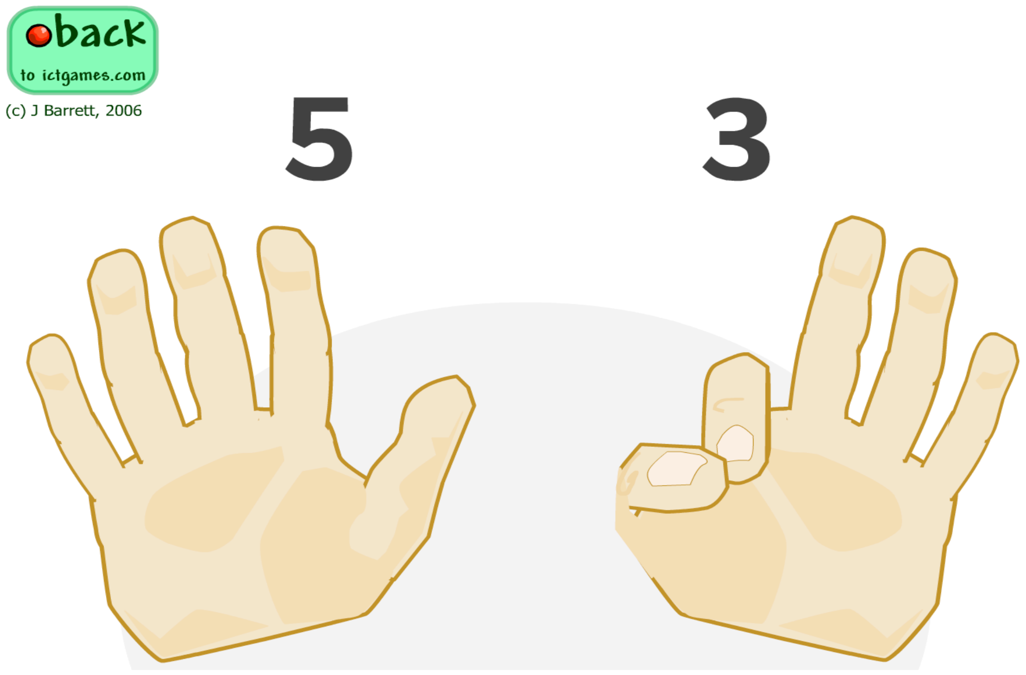


Figure 2.18. Teach how to express digits (ictgames.com).

There are applications that teach digits and characters [1,2]. Figure 2.17 and 2.18 show examples of these applications. However, the problem of these applications is that the applications do not support hand-recognition. To teach how to draw digits and characters on computer, we need sketch-based applications for children. We believe that our application can help children to learn how to draw digits and characters.

3. OVERVIEW OF VISUAL BASED RECOGNITION

Kara et al. introduced a visual-based (image-based) approach [19]. A visual-based recognizer represents shapes as binary templates. The input shapes are internally described as down-sampled bitmap images, which we call “templates”. The recognizer analyzes the input symbols with the templates and returns best-matched shapes. To determine best-matched shapes, the recognizer measures distances between the input shape and the shapes in the templates.

The benefit of a visual based recognition is as follows [19]:

- Free from segmentation errors
- Learning from a single example
- Easy to be extended
- Combined Classifier
- Achieving rotation invariance efficiently

First, the recognizer is free from segmentation errors. Feature-based approaches recognize shapes by segmenting users’ input data. Feature-based recognizers usually have two level recognizers: a low-level recognizer and a high-level recognizer. A low-level recognizer segments the input stroke and returns primitives of the stroke. For example, when a user draws digit “2”, the low-level recognizer segments the shapes as a curve and a line. PaleoSketch is an example of the low-level recognizer [29]. After segmenting the shapes, the high-level recognizer finally determines the shape by using the rules of primitives. However, if the segmentations have errors, the recognizer can have wrong determination. The visual-based recognizer is free from segmentation errors because it does not segment the input shapes. Additionally, feature-based recognizers have difficulty to determine multi-stroke shapes. However, a vision-based recognizer can recognize symbols without concerning about stroke numbers. Figure 3.1 shows an example of “sketch” symbols, which has multi-strokes.

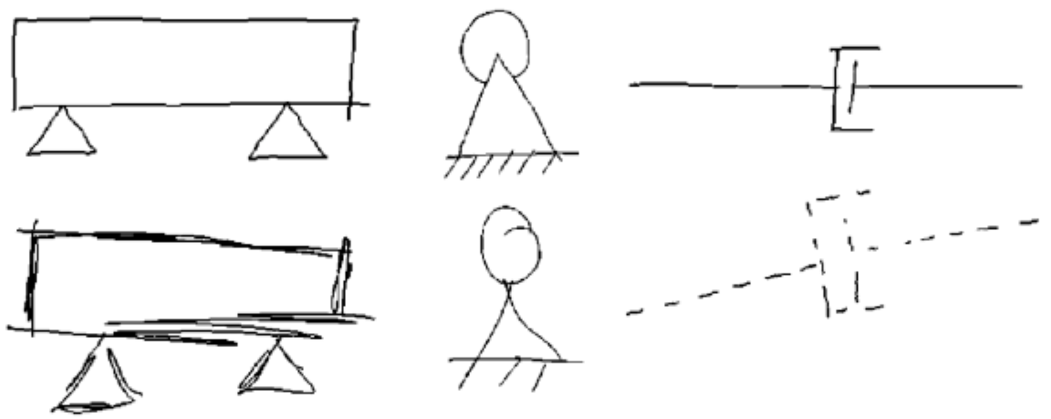


Figure 3.1. The top row shows symbols used in training, and the bottom row shows multi-stroked symbols (from [19]).

Second, the visual-based recognizer only needs a single template per each symbol. The problem of other template-matching approaches is that they need many templates. LeCun et al. introduced a recognizer for handwritten digits [24]. The recognizer uses 60,000 patterns for training purpose. Due to the heavy training data sets, the recognizer is suffers from high time complexity.

However, we found that a single template cannot support variations of digits and characters. To support this issue, our recognizer 3 templates per each symbol.

Third, the recognizer can extend recognizable shapes easily. As we discussed earlier, the feature-based recognizers have difficulty to determine multi-stroke symbols. To recognize these symbols, the feature-based recognizers need to make another rules per each symbol, but it is difficult to describe all the multi stroke symbols. On the other hand, the visual-based recognizer only needs to add template data.

Fourth, the visual-based recognizer has combined classifiers. To calculate minimum distances, the recognizer combines four classifiers: (1) the Hausdroff distance, (2) the Modified Hausdroff distance, (3) the Tanimoto coefficient, and (4) the Yule coefficient. [19]. When they combined the classifiers, the accuracy was higher than the worst performing classifier. However, our result shows that a single measurement (the Tanimoto distance) gives the best accuracy.

Finally, they use the polar analysis to recognize rotated symbols. One of the problems of template-matching approach is that the recognizers can be affected by angle difference. To overcome the problem, Kara et al. implemented the polar analysis.

Figure 3.2 shows the architecture of the visual-based recognizer of Kara et al. [19].

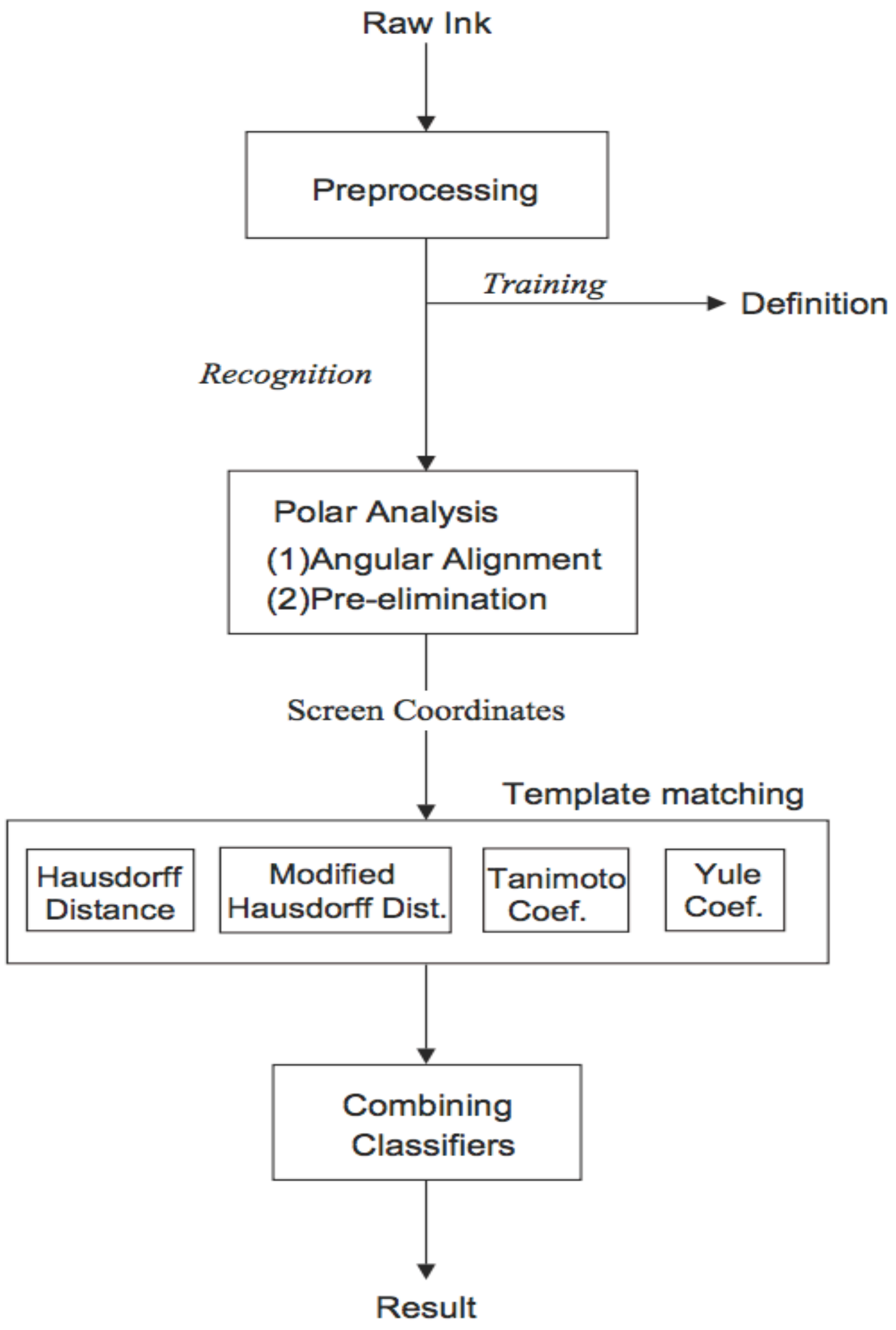


Figure 3.2. Architecture of the recognizer (from [19]).

The following sections describe each step of the recognizer.

3.1 Preprocessing

The raw input data generally needs to be refined to be used for the recognizer. When the visual-based recognizer converts the input data to the image, the image usually has too many or less data points, which can give negative effect for the recognizer. To make the points of the image appropriate for the recognizer, the preprocessing step frames and downs sample the initial image into a 48*48 square grid, which reduces the amount of data to consider while preserving the patterns of distinguishing characteristics.

To frame the image, the preprocessing includes two steps: (1) construct a bounding box aligned with the screen axes, and (2) the recognizer expands the shortest dimension of the bounding box without changing the location of the box's center to produce a square [19].

3.2 Polar Analysis

The problem of the template-matching approaches is that the approaches can be affected by their orientation. The angle difference between the origin data and the input data can make the comparing equations to have wrong values.

To solve the problem, template-matching algorithms are using rotation invariant recognition. For example, $1 recognizer incrementally analyzes the indicative angle between two symbols [46]. However, the approach has two problems as follows:
First, the approach can recognize "A" and "∀" as a same symbol. If a recognizer rotate the "A" to clockwise 90° and the "∀" to counter-clockwise, then they are matched.

Second, the complexity is too expensive for real-time application [19].

To alleviate this limitation, the vision-based recognizer uses the polar coordinate transformation. The main principle of the rotations is that Cartesian coordinates become translation in polar coordinates.

The polar coordinate can be expressed as:

$$r = \sqrt{(x - x_0)^{2} + (y - y_0)^2} \tag{3.1}$$

$$\theta = tan^{-1}\left(\frac{y - y_0}{x - x_0}\right) \tag{3.2}$$

where x_0 and y_0 are the origin and r is the radial distance between two points and $\theta$is the angle between the radius and the x-axis. Figure 3.3 explains the idea. Figure 3.3a illustrates a typical transformation and Figure 3.3b illustrates a pattern is rotated in the x-y plane.

However, this approach still has a size issue. When pattern is scaled in the x-y plane, the corresponding polar image stretches along the r-axis [19]. To eliminate the size issue, they normalized the size by using the “ink length” of the symbol and resampled the points to have uniformly spaced [8].

To find the angular offset between two shapes, the polar-analysis step uses a slide-and-compare algorithm. The algorithm incrementally displaces the images along the θ axis. At each displacement, the two images are compared to determine how well they are matched. To compare the measurement, the step uses a template-matching algorithm. In particular, they use the Modified Hausdroff distance. The measurement is explained in section 3.3.2.

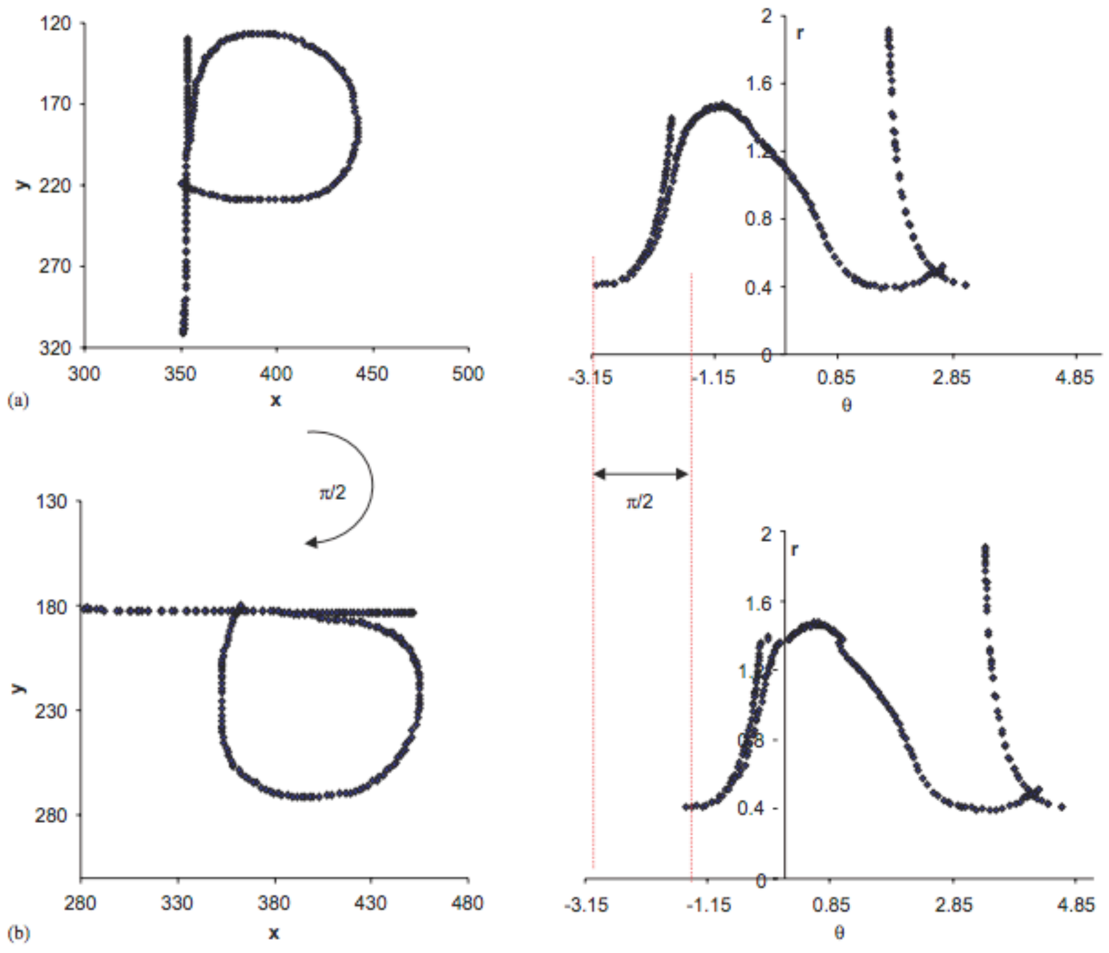


Figure 3.3. (a) letter "P" in screen coordinate (left) and polar coordinates (right), (b) when the letter is rotated in the x-y plane, the corresponding polar transform shifts parallel to the θ axis(from [19]).

3.3 Template Matching

Template-matching step can be simply described as the process of finding similar images between two images. While the other template-matching approaches use a single measuring classifiers, Kara's visual recognizer uses four measuring classifiers, and combines the result. Following subsections describe the measuring classifiers.

3.3.1 Hausdroff distance

The Hausdroff distance is used for calculating dissimilarity between two points. The Hausdroff distance has been successfully used for object detection in complex scenes [34]. However, a few researchers adapted the Hausdroff distance for hand-drawn pattern recognition: Cheung et al. [13] have used the measurement for character recognition, and Miller et al. [27] for digit recognition. However, Kara et al. used the measurement for image recognition.

The Hausdroff distance between two points set A and B can be defined as

$$H(A,B) = \max\big(h(A,B), h(B,A)\big) \tag{3.3}$$

$$H(A,B) = max_{a\in A}(min_{b\in B}\|a-b\|) \tag{3.4}$$

where $\|a-b\|$ represents a measure of distance between two points a and b, and h(A,B) means that the directed Hausdroff distance from A to B and correspond to the maximum of all the distances one can measure from each point in A to closet point in B.

However, the problem of the Hausdroff distance is that it is too sensitive to outliers [19]. For this reason, Dubuisson et al. modified the Hausdroff distance [14].

3.3.2 Modified Hausdroff distance

The Modified Hausdroff distance (MHD) replaces the max operator to average of distances:

$$h_{mod}(A,B) = \frac{1}{N_a}\sum_{a\in A} min_{b\in B}\|a-b\| \tag{3.5}$$

where N_a is the number of points in A.

The modified Hausdroff distance is defined as the maximum of the two directed average distances.

$$MHD(A,B) = MAX\big(h_{mod}(A,B), h_{mod}(B,A)\big) \tag{3.6}$$

3.3.3 Tanimoto coefficient

The Tanimoto coefficient calculates similarity between two images. The equation can be defined as:

$$T(A,B) = \frac{n_{ab}}{n_a+n_b-n_{ab}} \tag{3.7}$$

where n_a is the total number of black pixels in A, n_b is the total number of black pixel in B, and n_{ab} is the number of overlapping black pixels in A and B.

T(A,B) describes the number of matching points in A and B, and the result is between 0.0 (minimum similarity) to 1.0 (highest similarity). The problem of this equation is that if images contain mostly black pixels, the T(A,B) value can be vanished.

To solve the problem, the following equation is used

$$T^c(A,B) = \frac{n_{00}}{n_a+n_b-2n_{ab}+n_{00}} \tag{3.8}$$

, where n_{00} is the number of matching white pixels. T^c is called the Tanimoto coefficient complement.

The two equations can be combined to form the Tanimoto similarity coefficient [16].

$$T_{sr}(A,B) = \propto T(A,B) + (1-\propto)T^c(A,B) \tag{3.9}$$

$\propto$ is a weighting factor between 0.0 and 1.0.

3.3.4 Yule coefficient

Like the Tanimoto coefficient, the Yule coefficient also measures the similarity between two images. The equation is defined as:

$$Y(A,B) = \frac{n_{ab}n_{00} - (n_a - n_{ab})(n_b - n_{ab})}{n_{ab}n_{00} + (n_a - n_{ab})(n_b - n_{ab})} \tag{3.10}$$

, where $(n_a - n_{ab})$ is the number of black pixels in A that do not match in B. Similarly, $(n_b - n_{ab})$ is the number of black pixels in B that do not match in A.

The value of Y(A,B) is between 1.0 (maximum similarity) and -1.0 (minimum similarity). The difference of the Yule coefficient to the Tanimoto coefficient is that the Yule coefficient simultaneously accounts for the matching black and white pixels by the term n_{ab} and n_{00}.

3.3.5 Combining Classifiers

To combine four classifiers, we need to consider these issues:

- The Tanimoto and the Yule coefficient are measures of similarity while the last two classifiers are measures of dissimilarity,
- The classifiers have different ranges.

To solve the issues, Kara et al. transformed the Tanimoto and Yule similarity coefficients by reversing their values. Next, to eliminate the rage differences among classifiers, they normalized the value of all four classifiers to the range 0-1 using a linear transformation function. For each classifier, the transformation maps the distance scores to the range [0,1] while preserving the relative order established by that classifier.

For example, they converted the values of the Modified Hausdroff distance and the Hausdroff distance to have range from 0 to 1 by following equation.

$$H_c = 1 - \frac{H_d}{20} \tag{3.11}$$

Finally, having standardized the outputs of the four classifiers, they combined the results using a method similar to the sum rule introduced by Kittler et al. [19,20].

4. OVERVIEW OF SYMBOL RECOGNITION

The following sections explain our symbol recognition architecture and recognizer.

4.1 Limitation of Visual Based Recognition

Even the high accuracy of the vision-based recognizer (91.8% for digits on their data sets) [19], the recognizer still has limitations as follows:

1. High time complexity: because the recognizer uses four measurements [19] and combines the result of the measurements, it consumes many time resources.
2. Need to change rotation angle when the vision-based recognizer determines digits: initially range of rotation angle is from $-\pi$ to $+\pi$. However, when the recognizer recognizes digits, it changes the rotation angle to $\pm 90^\circ$. It can be a problem when the recognizer needs to analyze digits and other domain such as characters because the recognizer needs to change rotation angle by the shape the recognizer need to describe.

To solve the problem, we eliminate some steps of the vision-based recognizer. We use only one measurement from the vision-based recognizer: the Tanimoto coefficient. We analyzed all combinations of three measurements from the vision-based recognizer (the modified Hausdroff distance, the Hausdroff distance, and the Tanimoto coefficient) and we found that using the Tanimoto coefficient measurement alone gives the best accuracy for adults' data (Table 1).

Measurement	**Accuracy**
Tanimoto	0.9
MHD	0.838
HD	0.863
MHD+Tanimoto	0.888
HD+MHD	0.884
HD+Tanimoto	0.888
MHD+Tanimoto+HD	0.897

Table 1. Accuracies of measurements.

4.2 Symbol Recognition Architecture

Figure 4.1 explains our symbol recognition architecture. When users draw symbols on sketch panels, each points contains an x and y point. These points need to be preprocessed to have more refined drawings. To achieve the refined drawings, we use preprocessing steps. We will explain in more detail in the following subsections.

After preprocessing, the data goes into our template-matching algorithm. We use the Tanimoto coefficient to determine best-matched shapes using our pre-defined data sets.

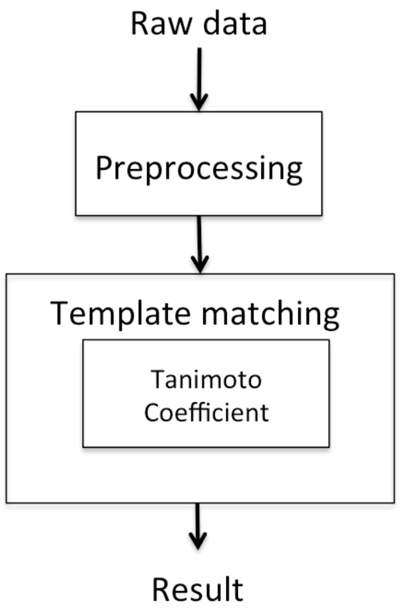


Figure 4.1. Architecture of EasySketch recognizer.

4.3 Preprocessing

The assumption of recognizers is that the input data will be clean data. The clean data means that the data will exclude all noise. The noise is points, which include unexpected points. To remove the noise, we need preprocessing steps. Preprocessing is needed for the following reasons as well:

- Each device can have noise data. For example, if the device has improper configuration, the device can generate the noise.
- The sampling rate is different per each device. For example, a mouse and a digital pen have different sampling rates.

- Each symbol has a different size. To compare each symbol, we need to adjust the symbol size to match the sizes of the other symbols.
- Each writer will make variations in the writing. For example, the speed difference of strokes can make the points to have sparse or concentrated areas (Figure 4.2).
- Strokes can have different stroke directions and orders. For example, Figure 4.3 shows the variations of drawing the digit "2".

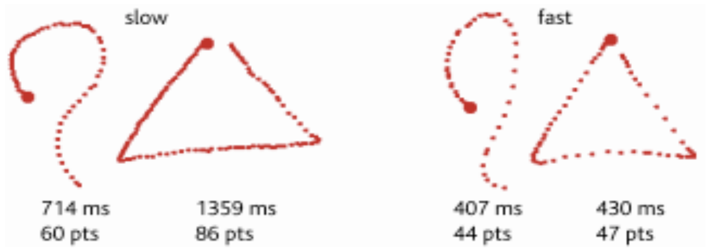


Figure 4.2. Slow and fast drawing makes considerable time difference and number of points (from [46]).

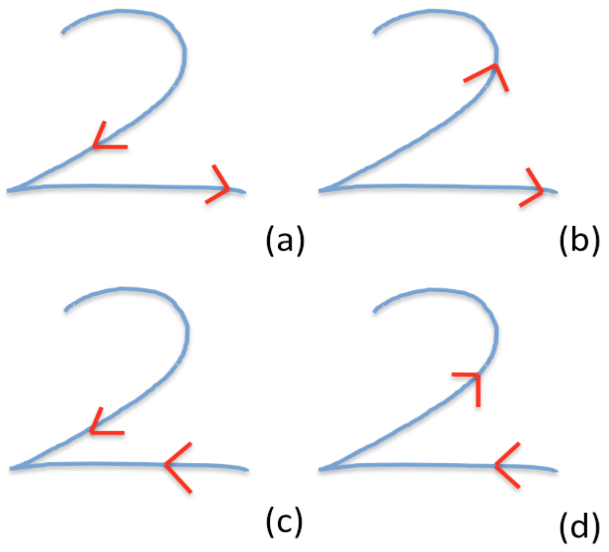


Figure 4.3. Different examples of drawing digit "2".

The following sections explain our preprocessing steps. To remove noise from our raw data, we have three preprocessing steps: (1) smoothing, (2) resampling, and (3) de-hooking.

4.3.1 Smoothing

Because the raw data can have zigzagged lines, the smoothing step makes the raw data to have consistent lines. The raw data is affected by speed and time of user's drawing and each device. Smoothing is done by substitution of a point with the weighted average of its neighboring points.

$$x'_i = \sum_{k=-m}^{m} \alpha_k x_{i+k} \tag{4.1}$$

$$y'_i = \sum_{k=-m}^{m} \alpha_k y_{i+k} \tag{4.2}$$

α_k is the weight on point (x_{i+k} ,y_{i+k}), and 2m is the neighboring range of the point. We set α_k= 1/6, m = 3. Figure 4.4 shows the result of smoothing.

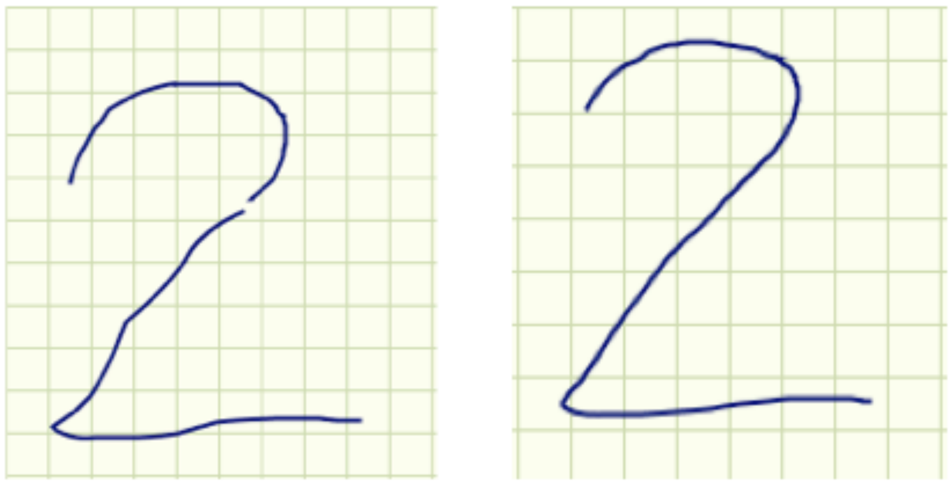


Figure 4.4. Result of smoothing (left) before, (right) after.

4.3.2 Resampling

The resampling step solves the following issues.

- Each samples and input data has different size of points. The data can have different size of points based on the users' input devices and their familiarity with the devices. If users draw sketches slowly, the data will have many points. The difference of point size makes it difficult to calculate minimum distances between each sample.
- The size and location is also different. We need to rescale the samples and make them to have the same coordinate plane.

To solve these problems, the resampling procedure includes following steps: (1) resamples points (64 points), (2) scales and translates to the same coordinate plane (48 pixel).

4.3.2.1 Resample points

There have been approaches that resampling stroke paths [12,38,47].
To resample points, we implemented Wobbrock's $1 recognizer algorithm [46]. The basic idea is that the algorithm makes the shapes to have the same size of points and the points to have equidistance. We defined M as a size of points of original stroke and N as expected size of points we want to make. Wobbrock tried to find the optimal size of N, which range is $32 \leq N \leq 256$. They found that 64 is the optimal value for N and we used the same N size (64). We also verified that 64 points is the optimal value for our recognizer. (Table2).

	64 point, 48px*48px	128 point, 48px*48px	64 point, 48px*48px
Tanimoto coefficient	0.9	0.89	0.88

Table 2. 64 resampling points and 48*48 pixel give the best accuracy (adults)

To resample each point, we first calculated the distance of stroke (*totalLength*), and increment (*I*).

$$totalLength = M.totalLength() \tag{4.3}$$

$$I = totalLength \,/\, (N\text{-}1) \tag{4.4}$$

Secondly, we gradually sum the distance from the first point. If the summed distance is higher than *I*, then the algorithm includes a new point.

```
Algorithm 1 Total length(points)
  Input: an input stroke $s$
  $distance = 0$;
  for $i$ to $sizeofpoints$ do
    distance = $distance$ + getDistance($p_{i-1}$, $p_i$)
  end for
  return distance
```

Algorithm 1: Total Length.

```
Algorithm 2 Resample
  Input: points of input stroke $s$
  $N$= $sizeofpoints$
  $I$= $Totallength$(points) / ($N$-1)
  $Distance = 0$
  $newPoints = p_0$
  for $i$ =1 to $N$ do
    $d$ = Distance($p_{i-1}$,$p_i$)
    if ($Distance + d$) >=$I$ then
      $q_x = p_{i-1x}$ + ($I$-$Distance$/$d$)*($p_{ix}$ - $p_{i-1x}$)
      $q_y = p_{i-1y}$ + ($I$-$Distance$/$d$)*($p_{iy}$ - $p_{i-1y}$)
      APPEND($newPoints$,$q$)
      INSERT($points$,$i$,$q$)
      $Distance = 0$
    else
      $Distance = Distance + d$
    end if
  end for
  return $newPoints$
```

Algorithm 2: Resample points.

Figure 4.5 and 4.6 explain our algorithm. Figure 4.5 illustrates two shapes that have different numbers of points. After our resampling step, each shape has the same points with each other. For convenience, we use 25 points as an example (Figure 4.6).

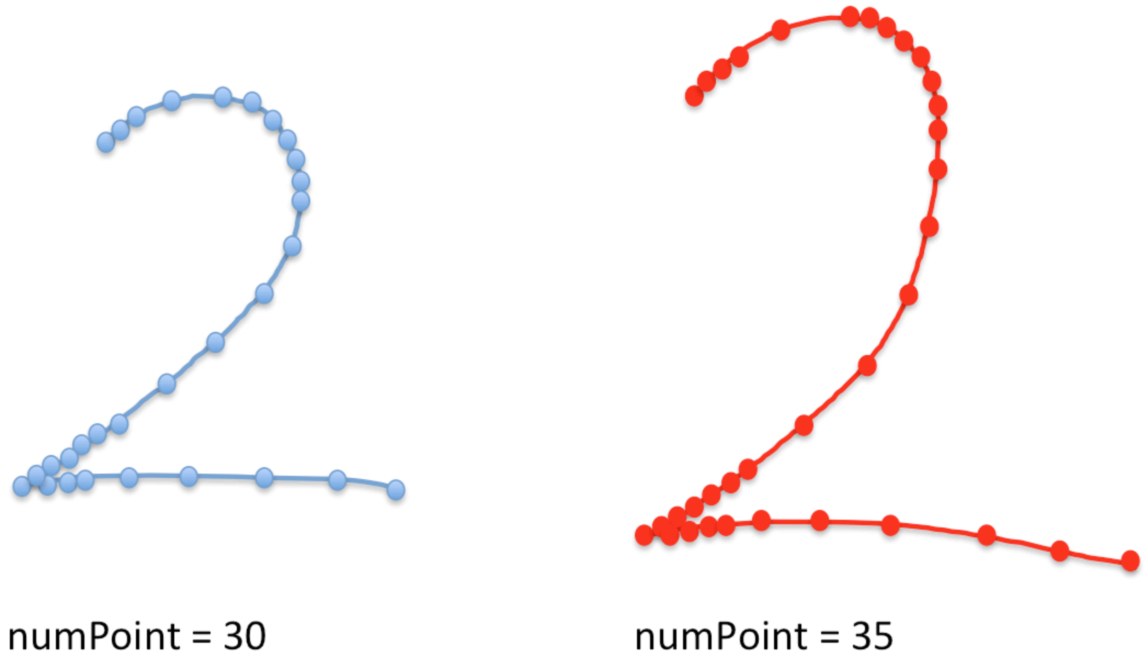


Figure 4.5. Each sample has different size of points.

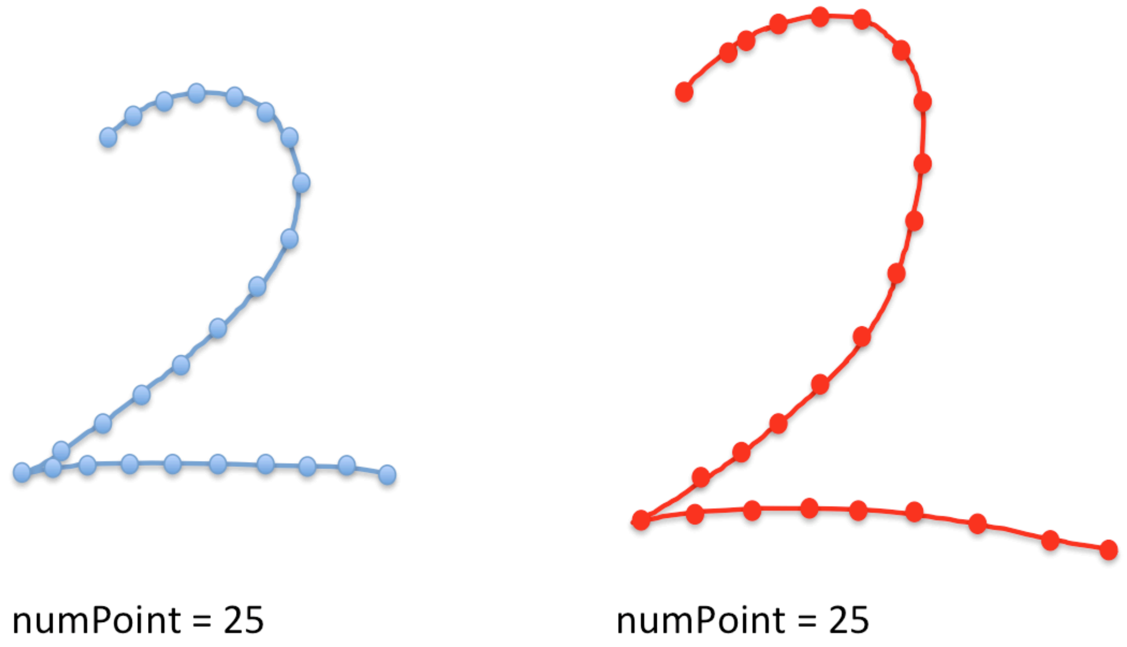


Figure 4.6. Each sample has the same size of points.

4.3.2.2 Scale and Translate

After the resampling step, the raw data have equidistance and the same size of points. However, the raw data still has a problem. The problem is that even the shapes has the same size of points, the size of shapes is still different with each other shape. This could hinder recognition performance.

To make each shape have the same size, we implement scale and translate step. By our analysis, we found that 48*48 pixels give is the best threshold (Table 2). The step has following steps:

- Construct a bounding box (48*48 pixels)
- Expand the shortest dimension of the boxes' center to produce a square (Figure 4.7).

The algorithm3 explains our steps and Figure 4.8 is our results.

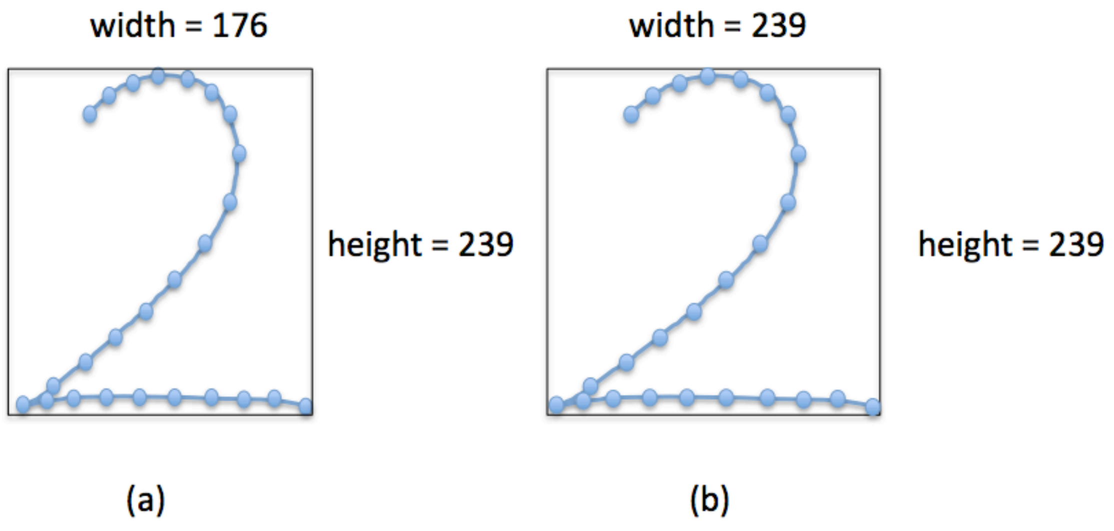


Figure 4.7. (a) Before expand the shape, (b) After expand the shape.

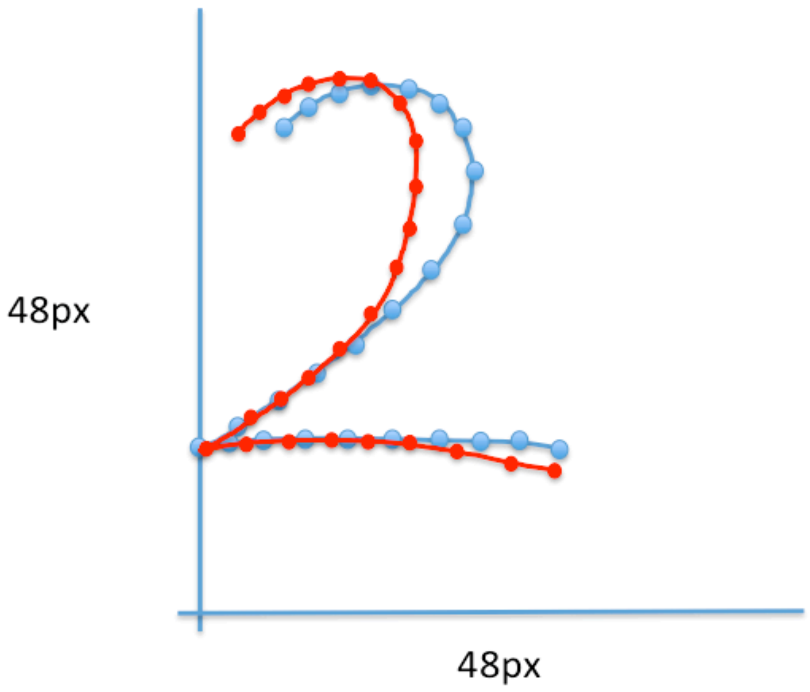


Figure 4.8. Scale and translate shapes.

Algorithm 3 Scale

```
Input: an input stroke $s$, and resampled points
$boundingBox = s.getBoundingBox()$
$MaxBoundingBox = 40$
$NumPoints = 64$
$width = boundingBox.width$
$height = boundingBox.height$
if $width >=$ "height" then
    $ratio = MaxBoundingBox$ / $width$
    $xAddToCenter = 0$
    $yAddToCenter = 40$ - ($height$*$ratio$)/2
else
    $ratio = MaxBoundingBox$ / $height$
    $xAddToCenter = 40$ - ($height$*$ratio$)/2
    $yAddToCenter = 0$
end if
$resampleWidth = Totallength$(points) / $NumPoints$
$newPoints = Resample$(points)
$leftMostX = newPoints.leftMostX$
$topMostY = newPoints.topMostY$
for each $p$ in $points$ do
    $p.x = (p.x$- $leftMostX)$*$ratio + xAddToCenter$
    $p.y = (p.y$- $leftMostY)$*$ratio + yAddToCenter$
end for
return $newPoints$
```

Algorithm 3: Scale points.

4.3.3 De-hooking

When users draw sketches on sketch panels, the sketches contain unexpected points look like "hook" (Figure 4.9). The problem of unexpected points is that it can lead our recognizer to have incorrect answers. Especially, if we use Freeman's chain code [12,17], the directions will be affected by the hook.

To solve the problem, we simply ignored the first three points and the last three points.

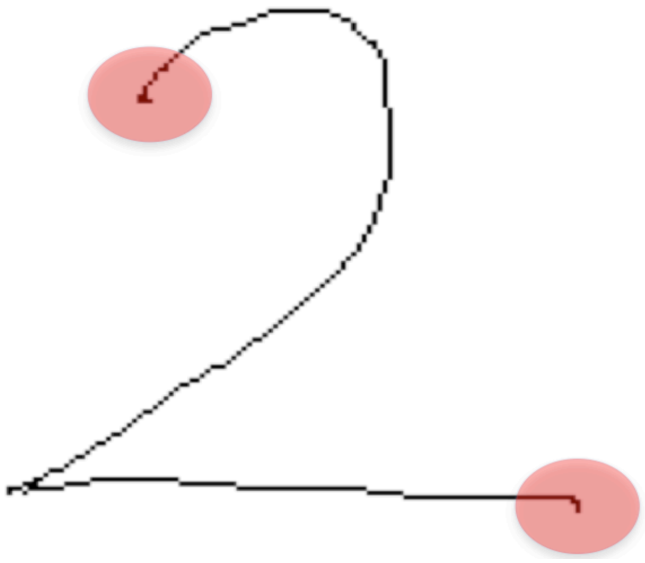


Figure 4.9. Hooks in digit "2".

4.4 Recognition

After the preprocessing steps, the raw input data has (1) the same size of points (64 points), (2) no noise (hook), and (3) the same size with each other shapes (48*48px). The cleaned raw data is given to the recognizer to determine the shapes. To recognize shapes, we use the template matching approach.

There have been approaches [18,41] that use a feature-based recognition, which generates rules of primitives in shapes. One of the examples is Mechanix [41]. To recognize shapes, Mechanix illustrates all the rules of primitives that the shapes should have. For example, a triangle has three lines and the lines should be connected with each other. The benefit of the feature-based recognition is that the recognizer can be extended easily if we can generate rules. However, the problem of the feature-based recognition is that we need to illustrate all the rules of primitives, and, if the shape has many variations, the rules can be more complex. Children especially have difficulty drawing shapes, and if they have less experience with writing devices such as a mouse or a digital pen, the shapes can have inaccurate drawings including zigzags.

To solve the problem, our recognizer uses the vision-based recognizer algorithm, which uses the template matching. As we discussed earlier, we use the Tanimoto coefficient, which gives the best result (Table 1).

5. IMPLEMENTED APPLICATION AND EVALUATION

5.1 Overview of EasySketch

To teach children how to draw digits and characters, EasySketch has following features:

- The system provides a sketch panel to draw symbols naturally.
- The system has an agent. When children correctly or incorrectly draw symbols, the agent keeps track of the children's correctness history and gives emotion as feedback. The system gives response by text and sound per each status.
- To teach children how to draw digits and characters, the system shows animated example symbols.
- To be implemented by instructors easily, the instructors can make assignments by writing questions in an excel file.
- To know the users' scores easily, the system shows a summary when they finish their drawings.

Figure 5.1 shows an interface of EasySketch.

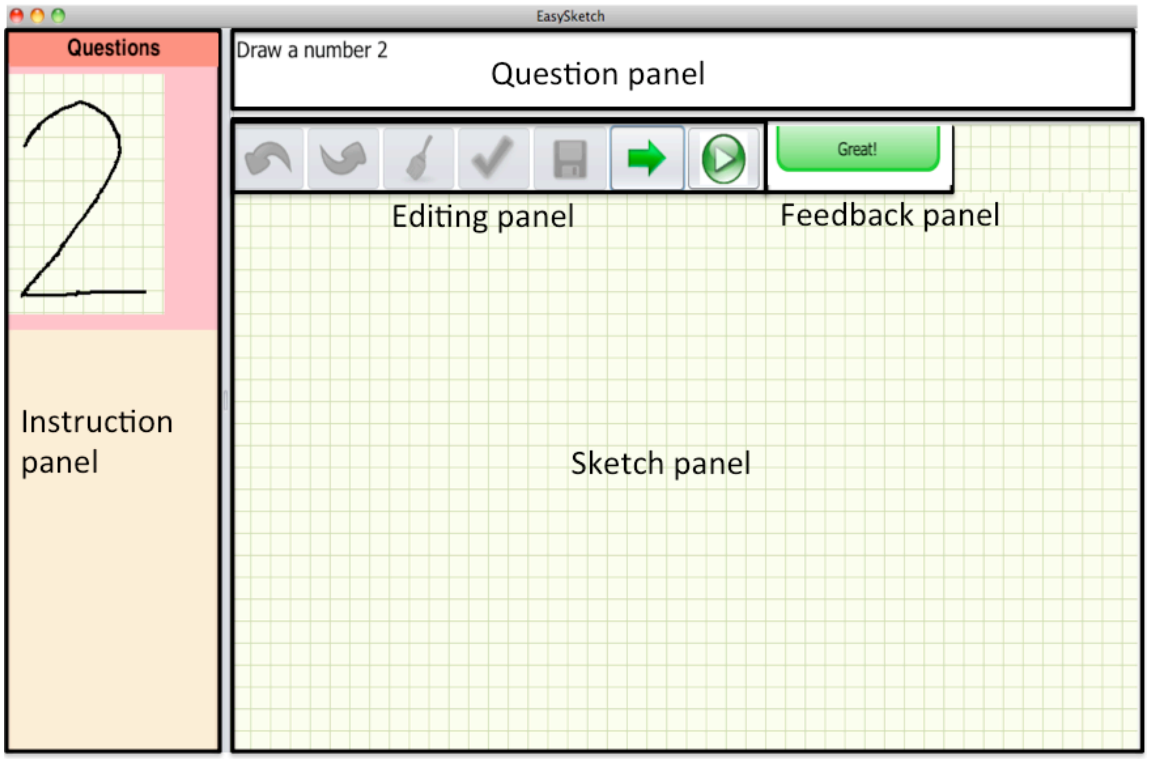


Figure 5.1. Interface of EasySketch.

EasySketch has the following panels:

- Instruction panel: shows an animated sample shapes that help children how to draw the symbols.
- Question panel: describes which shape children need to draw.
- Editing panel: enables children to edit their drawings.
- Sketch panel: allows the users to draw their digits and characters on this panel.
- Feedback Panel: shows the text feedback.

5.2 Agent System in EasySketch

The current systems for children use animations or games [6,7]. These methodologies can make children to feel interesting with the systems. However, the problem of the current systems is that the feedback (true or false) is not interesting and beneficial, which can make children to feel boring to use the systems. We focused that if we change the feedback to have emotion based on the children's correctness history, the children may feel interesting when they use the educational systems.

To support this issue, EasySketch implements an agent system that has emotion status. Whenever users draw their symbols, EasySketch compares the symbols with expected shapes and updates the agent's emotion.

The agent has six emotions: excited, happy, neutral, confused, disappointed, and angry. The agent changes its emotion whenever the children submit and check their drawings. The initial emotion status begins with "neutral" emotion. If the users draw their symbols correctly, the agent changes its status from "neutral" to "happy" and shows a message ("Great!") on feedback panel and plays an audio file. However, if the users draw symbols incorrectly, the agent gives a message ("That does not seem right") and plays an audio file and changes the emotion status to "confused".

Figure 5.2, 5.3, and 5.4 explain the emotions and messages per each status.

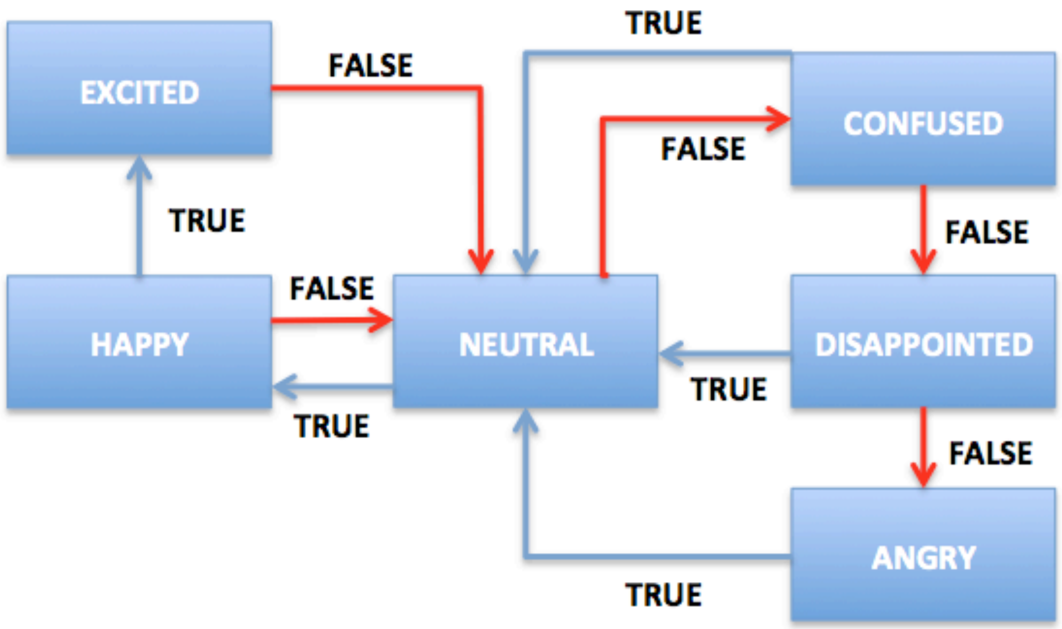
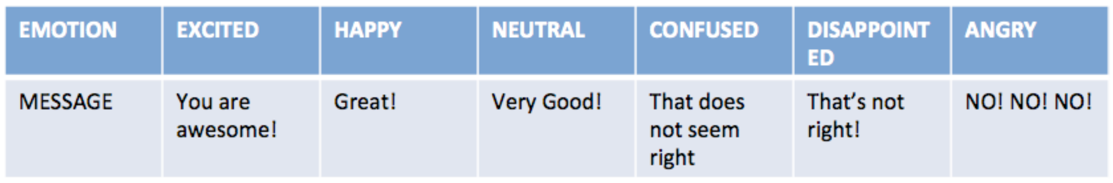


Figure 5.2. Emotion status in EasySketch.



EMOTION	EXCITED	HAPPY	NEUTRAL	CONFUSED	DISAPPOINT ED	ANGRY
MESSAGE	You are awesome!	Great!	Very Good!	That does not seem right	That's not right!	NO! NO! NO!

Figure 5.3. Feedback.

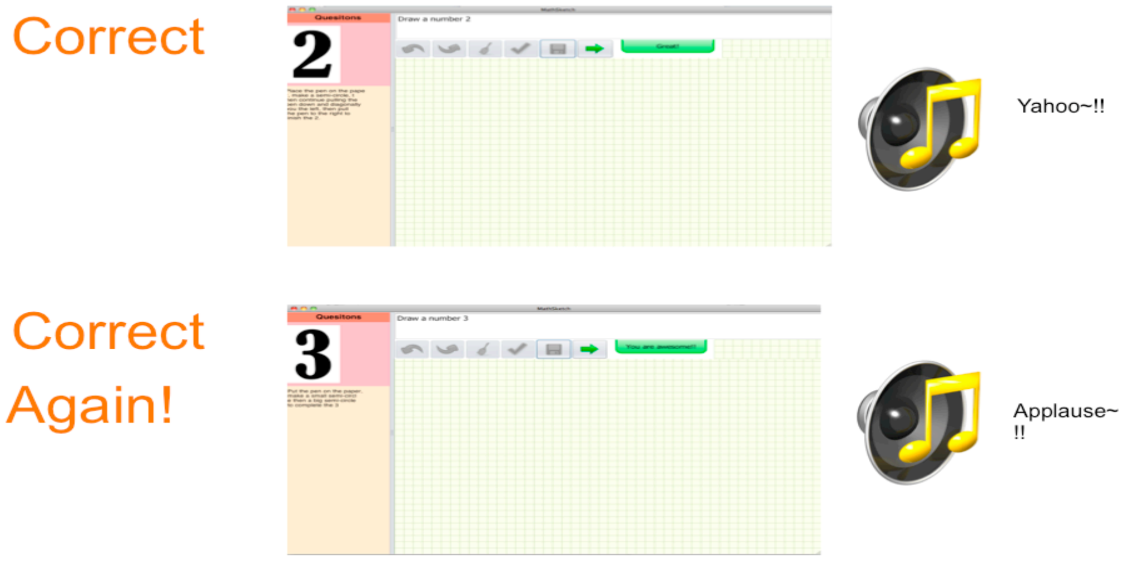
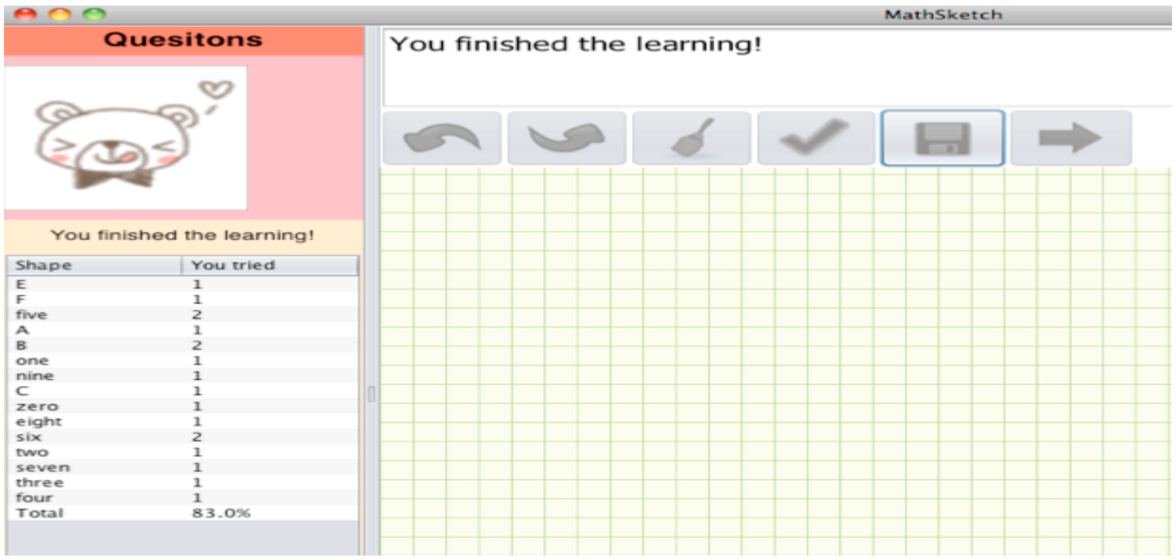


Figure 5.4. Message and audio are changing based on the correctness of drawings.

After finishing their drawings, the system shows overall scores and numbers of attempt per each symbol. The system changes an image based on their scores (Figure 5.5 and Table 3).



You finished the learning!

You finished the learning!

Shape	You tried
E	1
F	1
five	2
A	1
B	2
one	1
nine	1
C	1
zero	1
eight	1
six	2
two	1
seven	1
three	1
four	1
Total	83.0%

Figure 5.5. Shows scores when they finish the drawings.

Score	Image
0 ~ 20	
21~40	
41~60	
61~80	
81~100	

Table 3. Changing the icon by their final scores.

5.3 Interface for Instructor

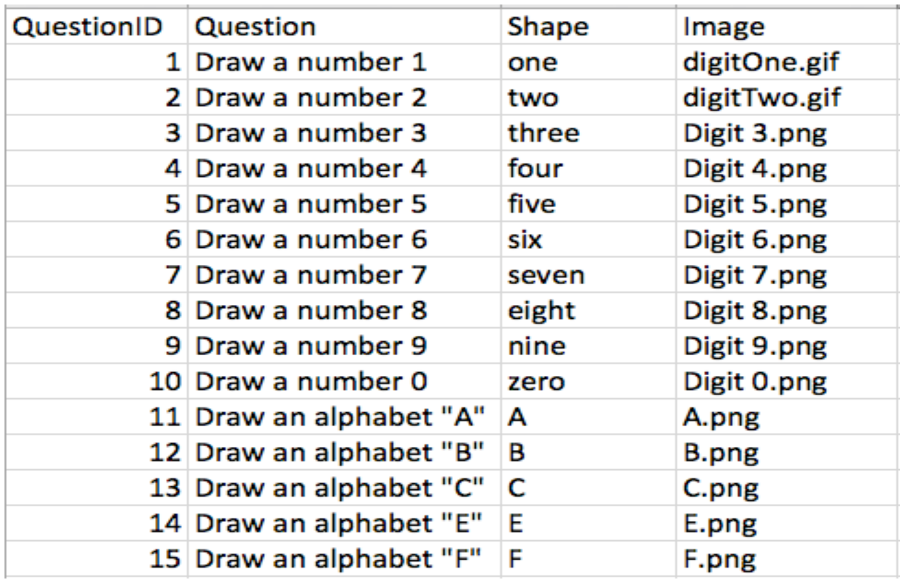
The current systems' problem is that instructors cannot make their own assignments [6,7]. If children need to practice some specific shapes, the only available option for the users is finding questions that seem similar with their expected questions. However, this step needs users' effort to find the question, and if they cannot find the question, there is no way to create a question that their children need to draw.

To support this issue, EasySketch provides an easy way for instructors to implement their own questions. To make their own questions, they require three steps: (1) edit an excel file, (2) sketch the symbols and (3) save them.

First, the excel file has five categories as follows:

- Question ID: represents specific ID per each question.
- Question: describes a question text that children need to draw (e.g. "Draw a number 1.").
- Shape: is an expected shape that the children need to draw
- Image: is an image file name of the expected shape.

Figure 5.6 shows an example of questions.



QuestionID	Question	Shape	Image
1	Draw a number 1	one	digitOne.gif
2	Draw a number 2	two	digitTwo.gif
3	Draw a number 3	three	Digit 3.png
4	Draw a number 4	four	Digit 4.png
5	Draw a number 5	five	Digit 5.png
6	Draw a number 6	six	Digit 6.png
7	Draw a number 7	seven	Digit 7.png
8	Draw a number 8	eight	Digit 8.png
9	Draw a number 9	nine	Digit 9.png
10	Draw a number 0	zero	Digit 0.png
11	Draw an alphabet "A"	A	A.png
12	Draw an alphabet "B"	B	B.png
13	Draw an alphabet "C"	C	C.png
14	Draw an alphabet "E"	E	E.png
15	Draw an alphabet "F"	F	F.png

Figure 5.6. Creating an excel file.

After editing the excel file, instructors need to draw and save their sketches. Instructors can easily save their sketch by clicking the save button in the editing panel. After saving their sketches, they need to create folders and change the name as their expected shapes. Figure 5.7 shows an example of a new question that adds a Korean character “kim”.

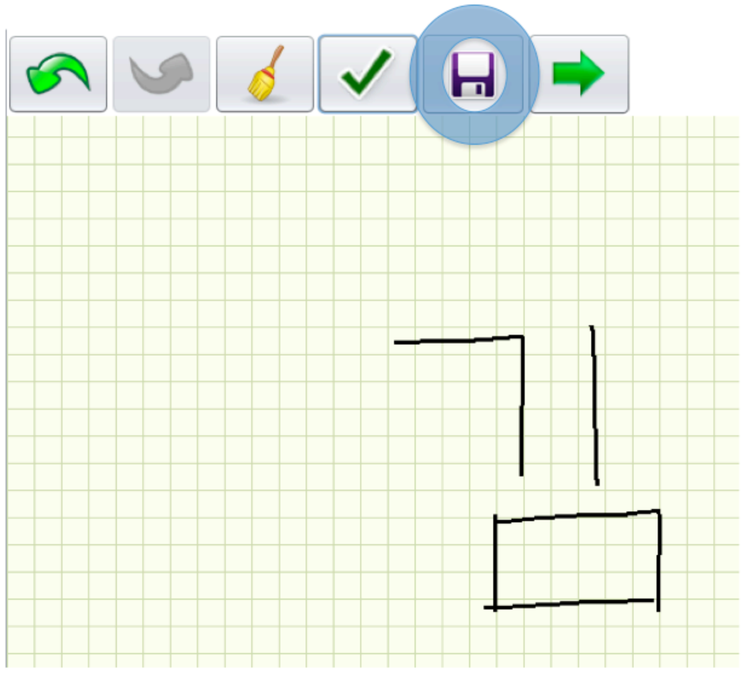


Figure 5.7. Instructors can add question.

5.4 Interface for Children

Because our target users are children, we considered the easiness of the interface. The contents should be easy enough to read and children should be able to edit and submit their sketches easily, and the sketch panel should be large enough to draw.

To support these issues, we made the sketch panel to large enough for children and the sketch background to look like a paper. Additionally, for their editing, the system includes six icons (undo, redo, new sketch, submit, save, and go next question) for each behavior, which looks straightforward for them (Figure 5.1).

5.5 Evaluation

5.5.1 Result of our user test

To collect digits and characters data, we divided the groups into two groups: an adult group and a children group. Additionally, we divided the children group into two groups by age (under 4 year old children and 7 year old children). Generally, 4 year old children are still learning digits and characters, but 7 year old children know how to draw digits and characters.

We had four adult volunteers and each of them was asked to draw five sketches per each shape (digit: 0 to 9 and characters: A to F). All the volunteers were from the department of Computer Science at Texas A&M University. One of them used a digital pen and three of them used mice. To collect data from adults, we used SOUSA [30], which is a web-based application that can collect users' sketch data.

For the children group, we had ten volunteers (3 years old: one child, 4 years old: two children, and 7 years old: seven children) who did not have any experience with sketch-based educational applications. For our user study, the 7 year old children drew one sketch per each digit (0 to 9) and two sketches per each character (A to F). The 7 year old children knew how to draw digits and characters. On the other hand, the under 4 year old children had little knowledge about digits and characters. For the user study of the under 4 year old children group, we reduced the shapes to draw (digits: 0 to 9 and characters: A to D) and they drew one sketch per each shape. During the children's user study, we processed our user studies with their parents to make the children to feel comfortable.

We evaluated the accuracy of the recognizer by dividing numbers of correctly recognized symbols by the total numbers of test symbols.

Accuracy = count of correct symbols / total number of test symbols (5.1)

The total accuracy of the data was 90% for adults with 320 data sets (digits: 89% and characters: 90.8%) and 83.75% for 7 year old children with 123 data sets (digits: 80% and characters: 90%), and 34.4% for 3 and 4 year old children with 28 data sets (digits: 37.5% and characters: 25%). Figure 5.8 and 5.9 show our results.

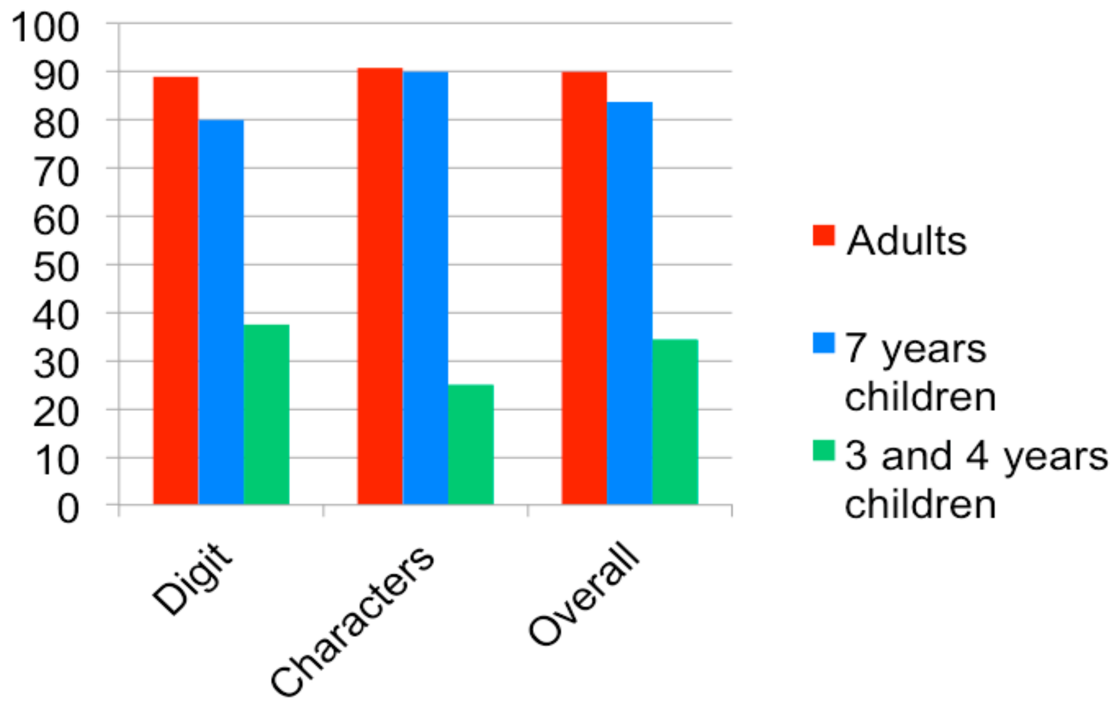


Figure 5.8. Result of user studies.

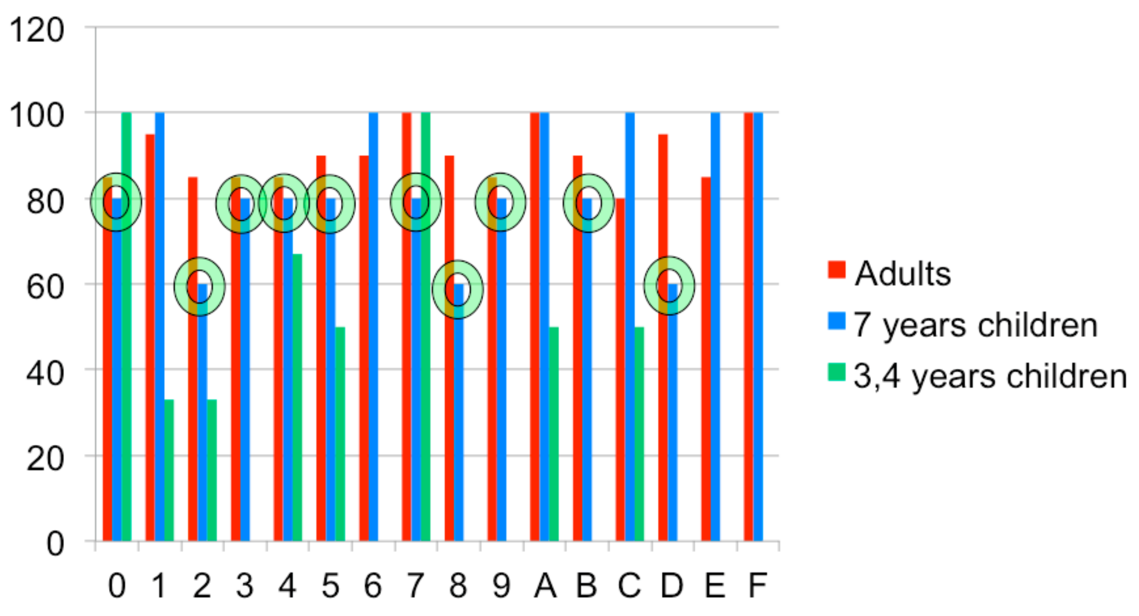


Figure 5.9. Result of user studies per each symbol (circle means there is many recognition difference between adults and 7 years children).

The following Table 4, 5, and 6 show the result of our user study.

		RECOGNIZED															
		one	**two**	**three**	**four**	**five**	**six**	**seven**	**eight**	**nine**	**zero**	**A**	**B**	**C**	**D**	**E**	**F**
EXPECTED	**one**	**0.95**							0.05								
	two		**0.85**	0.05							0.1						
	three			**0.85**					0.1				0.05				
	four				**0.85**			0.05									
	five					**0.9**									0.05		0.05
	six					0.05	**0.9**		0.05								
	seven							**1**									
	eight					0.05			**0.9**				0.05				
	nine							0.15		**0.85**							
	zero										**0.85**			0.15	0.1		
	A											**1**					
	B			0.1									**0.9**				
	C										0.2			**0.8**			
	D													0.05	**0.95**		
	E								0.05				0.1			**0.85**	
	F																**1**
	Overall	**0.9**															

Table 4. Result of our recognizer (adult).

		RECOGNIZED															
		one	**two**	**three**	**four**	**five**	**six**	**seven**	**eight**	**nine**	**zero**	**A**	**B**	**C**	**D**	**E**	**F**
EXPECTED	**one**	**1**															
	two		**0.6**	0.4													
	three			**0.8**		0.2											
	four				**0.8**										0.2		
	five					**0.8**									0.2		
	six						**1**										
	seven	0.2						**0.8**									
	eight					0.2			**0.6**				0.2				
	nine							0.2		**0.8**							
	zero										**0.8**			0.2			
	A											**1**					
	B							0.2					**0.8**				
	C													**1**			
	D												0.2	0.2	**0.6**		
	E															**1**	
	F																**1**
	Overall	**0.8375**															

Table 5. Result of our recognizer (7 age children).

		RECOGNIZED													
		one	**two**	**three**	**four**	**five**	**six**	**seven**	**eight**	**nine**	**zero**	**A**	**B**	**C**	**D**
EXPECTED	**one**	**0.33**						0.33	0.33						
	two	0.33	**0.33**						0.33						
	three			**0**					0.33				0.66		
	four				**0.66**							0.33			
	five				0.5	**0.5**									
	six				0.5	0.5	**0**								
	seven							**1**							
	eight						0.5		**0**				0.5		
	nine		0.5						0.5	**0**					
	zero										**1**				
	A										0.5	**0.5**			
	B				0.5	0.5							**0**		
	C										0.5			**0.5**	
	D										0.5		0.5		**0**
	Overall	**0.344**													

Table 6. Result of our recognizer (3,4 age children).

After finishing our user study, children and parents were asked to answer the following surveys (Table 13 and 14 in Appendix) and Table 7 is the average of their feedback. All the children had experience with a sketch-recognition system (e.g. iPhone), but they have not used a sketch-based educational system. They agreed that our system is appropriate for learning and easy to use. However, they would like more interesting visual feedback such as animations in the system.

Description/Identification of Survey Item	Scale
1. Does your child have experience with sketch-based application?	No
2. The software is easy to understand and use for the children?	5
3. The software can help the children to learn?	5
4. The children seem to prefer these kinds of interactions over than traditional mouse and keyboard?	5
5. The children retain attention for longer periods when using the system compared to similar pen and paper activities?	3
6. It will be feasible to use similar games on a regular basis as part of the course content?	5
7. The feedback messages of the system were understood by the children?	5
8. The feedback messages of the system were appropriate for their age?	5

Table 7. Feedback of user study

5.5.2 Analysis of children's data

When we analyzed the children's sketches, we found that the overall accuracy of 7 year old children (83.75%) is similar to adults' data (90%). However, accuracy of 3-4 year old children (37.5%) had many differences with both adults and 7 year old children. Additionally, when we analyzed the children's sketches, we found the following issues:

1. The children's sketches have more numbers of points than adults'. Because the numbers of points are increased by time to draw, we can know that children need more time to draw than adults (Figure 5.10).

2. The adults drew shapes larger than the children's sketches (Figure 5.11).
3. The under 4 year old children drew shapes smaller than 7 year old children's, and they drew faster than 7 year old children. However, by the observation of their drawings, the author realized that they gave less consideration to draw than 7 year old children because they had less knowledge with the shapes and they did not know what they are drawing (Figure 5.11).
4. The children had difficulty drawing complex shapes, which include curve lines. Figure 5.9 shows that the accuracies of children's charts have lower rates (less than 80%) than adults with 0, 2, 3, 4, 5, 7, 8, 9, B, and D. Figure 5.12 and 5.13 are the examples of children's sketches.

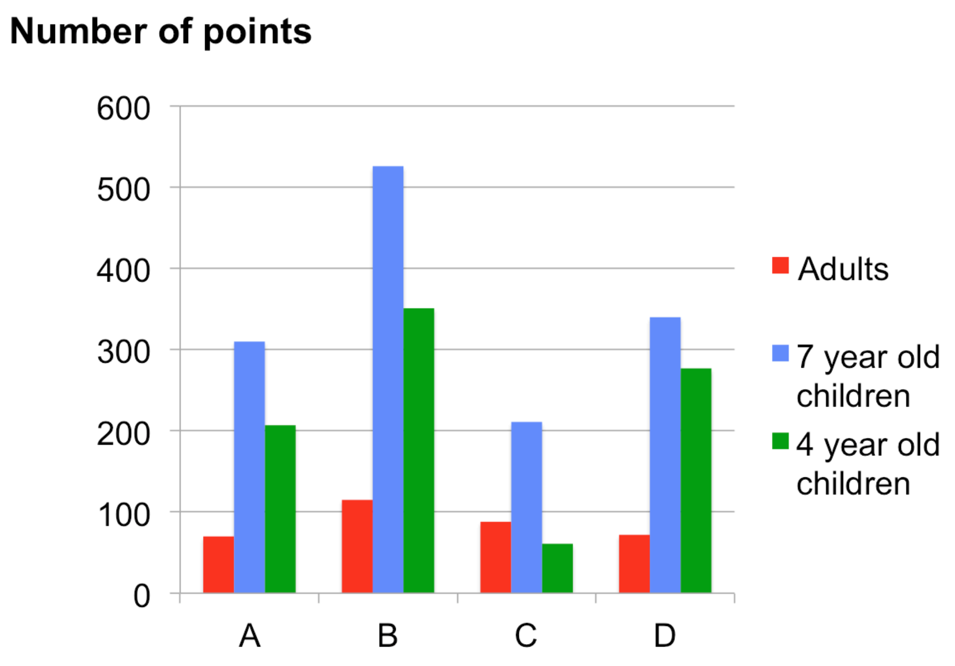


Figure 5.10. Comparing numbers of points.

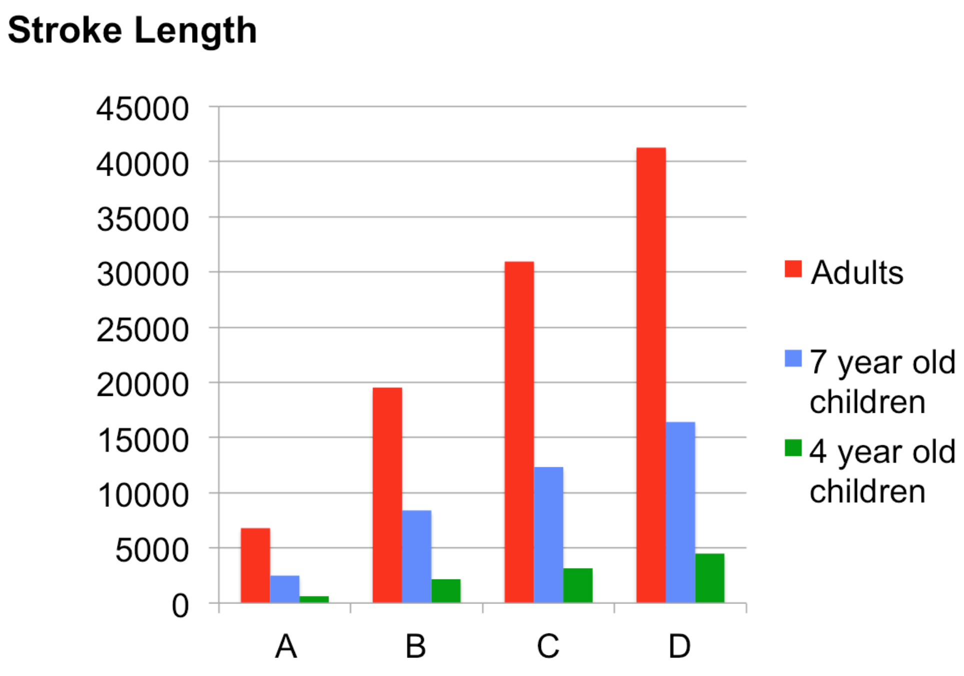


Figure 5.11. Comparing stroke length.

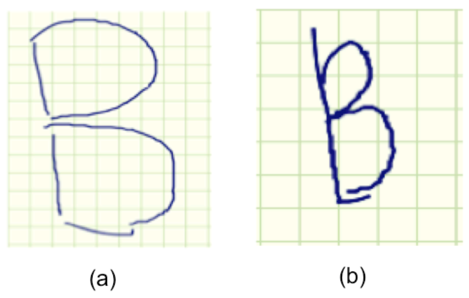


Figure 5.12. Drawing B (left: 7 year child, right: 4 year child).

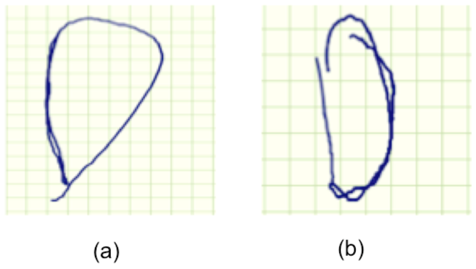


Figure 5.13. Drawing D (left: 7 year child, right: 4 year child).

After our user study, we had the following feedback from the children and their parents:

1. The children had difficulty using a digital pen because the digital pen (Figure 5.14) has a smaller size than a normal pen, and they could not draw sketches naturally. When they drew symbols by using a pen and paper, the symbols had similar shapes to adults' (7 year old children).
2. The sketch screen was improper for children: when they finished their user studies, they still preferred to use a sketch panel rather than traditional devices such as a keyboard or a mouse. However, they preferred a pen and pencil or their fingers than a digital pen.

Figure 5.14. Toughbook pen.

5.5.3 Gender Difference

In Educational and Developmental Psychology, there is interest in potential gender difference within visual-motor skills. While the more dominant belief is that females develop motor skills faster than male, there is much disagreement between researchers. For example, Brown [10] and Tennant [39] described that female children's skill were superior to that of male children's. However, Lotz et al. insisted that the male children had better vision-motor skills than the female children [26]. One potential reason for conflicting result may be from the type of measurement. Additionally, another limitation of these research is that they combined all ages and just compared their gender regardless of their ages.

To verify the gender difference, we used the same age group (7 year-olds) and compared 123 data sets with a total of 7 children (four female children and three male children).

After our test, we realized that the recognition accuracy of the female children is higher than the male children's for both digits and characters. The overall accuracy of the female children was 88.5%. However, the male children's accuracy was 77.8%. The recognition difference was higher in characters (96.6% for female and 80% for male) than digits (83.7% for female and 76.7% male). Figure 5.15 shows the recognition accuracy per gender.

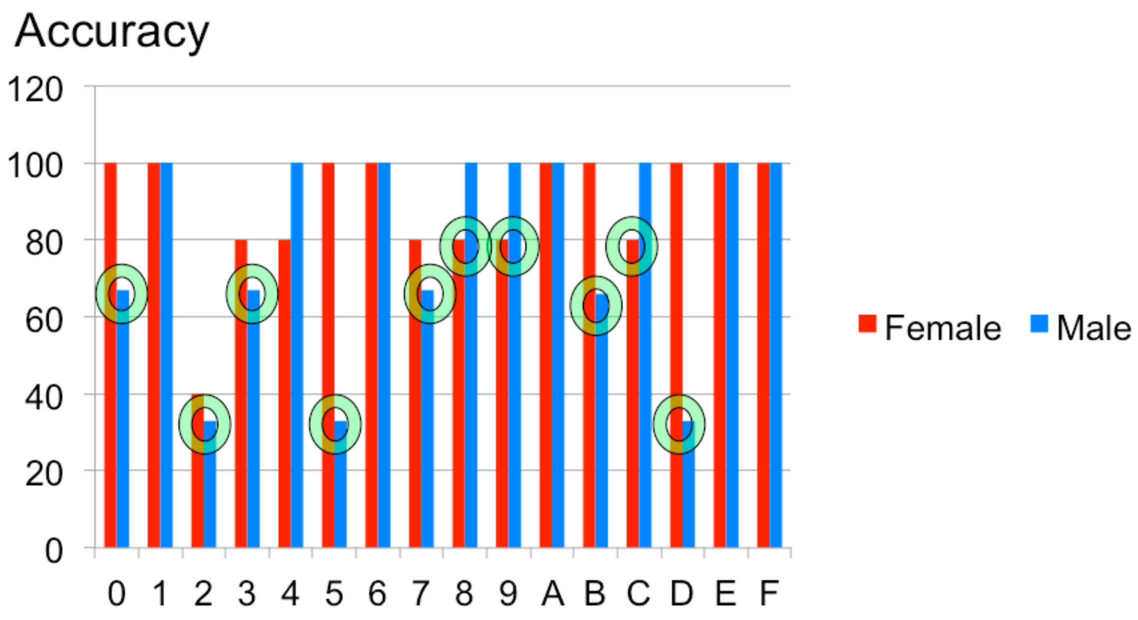


Figure 5.15. The female children draw better than the male children.

5.5.4 Improve the accuracy for children

During the analysis of the children's data, we realized that we should not use the same threshold for adults and children because their drawing patterns are different. As we discussed earlier, we found that the Tanimoto coefficient algorithm, 64 resampling numbers of points, and 48*48 pixel give the best accuracy for adults. However, the children's sketches have more numbers of points and smaller shapes in their sketches than adults.

To increase the accuracy of children, we analyzed 7 year old children's data and found that the 32 resampling points gives the best accuracy for their sketches.

Table 8 and Figure 5.16 show our results, and the accuracy has increased by 32 resampling points.

Points and Pixels	Measurement	**Accuracy**
32 points, 48px*48px	Tanimoto	0.8375
64 points, 48px*48px	Tanimoto	0.8125
32 points, 32px*32px	Tanimoto	0.8
32 points, 64px*64px	Tanimoto	0.7875
64 points, 32px*32px	Tanimoto	0.7875
64 points, 64px*64px	Tanimoto	0.775
128 points, 48px*48px	Tanimoto	0.775
128 points, 64px*64px	Tanimoto	0.75
128 points, 32px*32px	Tanimoto	0.7375

Table 8. 32 resampling points and 48*48 pixels give the best accuracy (7 year children).

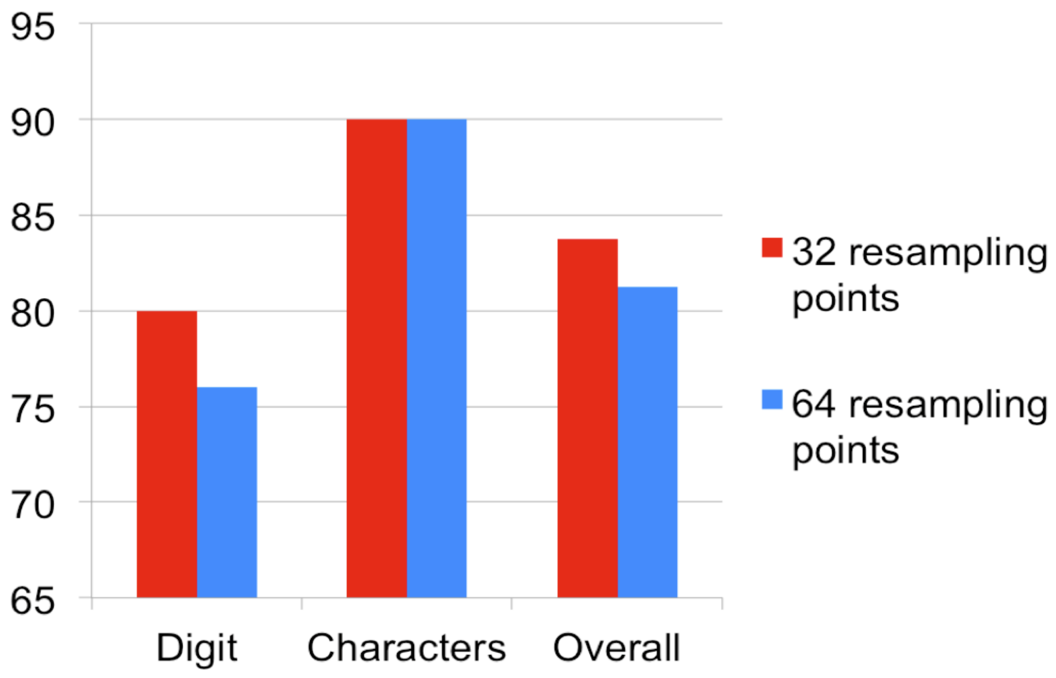


Figure 5.16. 32 resampling points gives more accuracy than 64 resampling points for 7 years.

Figure 5.17 shows the example of the shape that was recognized correctly when we used 32 resampling points and 48*48 pixels, but it was misrecognized when we used 64 resampling points and 48*48 pixels.

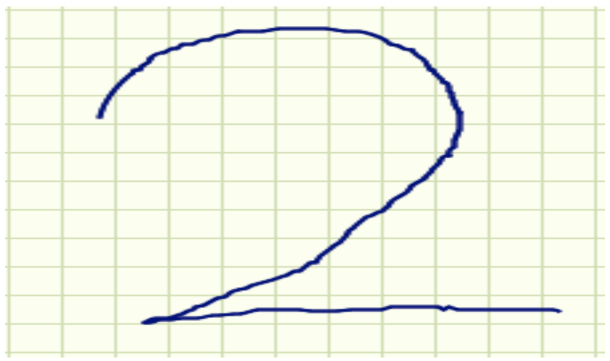


Figure 5.17. The digit “2” is recognized as a digit “8” if we use 64 resampling points.

5.5.5 Analysis of best features for recognition for children and adults

We used the forty four feature sets from [31] to find to further understand the difference between drawings from children and adults.

During our test, we validated best feature sets for children and adults by 10 fold-cross validation. Table 9 and 10 show the top five best accuracy feature sets for adults and children, and we realized that the best features sets were different for each other. Furthermore, the best feature sets for characters were also different from the twenty basic shapes from Paulson's recognizer (Table 11) [31]. By the analysis, we realize that we need to use another feature sets for adults and children to recognize characters (A-F).

Feature	Accuracy
Percentage of strokes that passed the line test	0.733333
The error of the line fit	0.7
Length ratio between the major and minor axis	0.7
The error of the spiral fit	0.683333
The error of the rectangle fit	0.683333

Table 9. Best Feature Set for Children (7 age child)

Feature	Accuracy
The perimeter (of bounding box) to stroke length ratio	0.783333
Number of revolutions of sub dot	0.758333
The error of the line fit	0.725
The error of the rectangle fit	0.725
The percentage of slope test that passed	0.716667

Table 10. Best Feature Set for Adults.

Feature	Accuracy
EndPoint to stroke length ratio	1.0
Total rotation of the stroke	1.0
Normalized distance between direction extremes	0.9
Direction Change Ratio	0.9
Curve least square error	0.9

Table 11. The Best Feature Sets of Paulson's recognizer [31].

5.5.6 Recognize sketcher's age by drawings

When we analyzed drawers' sketches, we found that they drew sketches differently by age. As we can in Figure 5.18, the three children groups (3 year-olds, 4 year-olds, 7 year-olds, and adults) had different drawing styles. If computers can automatically determine sketchers' ages by their drawings, we can have many benefits as follows:

- Teaching their drawings: If we can understand the children's difficulty to draw some shapes, we can guide them more carefully.
- Computer can automatically change the feature sets for the recognizer: If computers can recognize the children's age by just their sketches, the computers can automatically change their recognizers' feature sets and/or thresholds by their age, which will allow them more recognition accuracy. For example, if a three year old child has difficulty drawing a circle, the computer can change the threshold of a circle for the child.

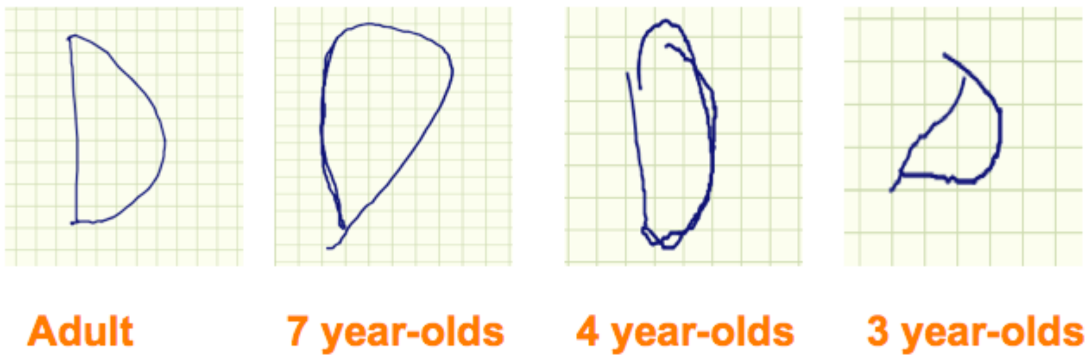


Figure 5.18. An example of character "D".

To determine the sketcher's age, we cross-validated 475 data sets using all forty four features from [31]. Analysis of this data showed that there are several differences between adults and children's drawings. Using such identified features, we were able to distinguish children's drawings from adults' drawings. We correctly identified the drawers' age (age 3, 4, 7, or adult) with a precision of .783, recall of .768 and an f-measure of .773. When we grouped age 3 and 4 into "toddler" group and grouped age 7 and adults into "matured" group, we got a precision of .952, recall of .957, and an f-measure of .952. Removing the adults, and distinguishing between "toddlers" and 7 year-olds, we got a precision of .902, a recall of .902, and an f-measure of .902. When we compared only between 3, 4, and 7, we got a precision of .868, recall of .874, and an f-measure of .862.

6. SUMMARY

In this paper we investigated the different drawings patterns between children and adults. To test if there are differences between children and adults, we collected a total of 475 data sets from four adults and ten children (3 year-olds: one child, 4 year-olds: two children, 7 year-olds: seven children). To make a separate recognizer for children, we changed the number of resampling points and we increased the recognition accuracy for children from 81.25% (using the original adult-tuned recognizer) to 83.75% (using adjusted threshold for children). To further analyze the drawing pattern difference, we used a set of fourty-four geometric features from Paulson et al. [31] and found that different feature sets dominates recognition accuracy for children and adults. The best feature sets for children was "Percentage of strokes that passed the line test". On the other hand, "The perimeter (of bounding box) to stroke length ratio" was the best feature for adults.

Analysis of the data showed that the 7 year-olds had difficulty drawing shapes, which include curved lines (0, 2, 3, 4, 5, 7, 8, 9, B, and D). Additionally, we found that different ages have different distinguishing features. To automatically identify the drawer's age, we cross-validated the 475 feature sets with the set of forty-four features [31], and we identified their ages (age 3, 4, 7, or adult) with a precision of .783, recall of .768 and an f-measure of .773. When distinguishing between toddlers (age 3-4) and matures (age 7 and adults), we got a precision of .952, recall of .957, and an f-measure of .952. Removing the adults, and distinguishing between toddlers and 7 year-olds, we got a precision of .902, a recall of .902, and f-measure of .902. Distinguishing between 3, 4, and 7 ages, we got a precision of .868, recall of .874, and an f-measure of .862.

Furthermore, we found that there is a potential sketch skill difference between female and male children. We compared 7 year olds children group’s data sets with four female children and three male children and we realized that the female children (88.5%) drew more accurately than the male children (77.8%).

Finally, we introduced a sketch-based teaching assistant tool called EasySketch, which includes a virtual agent system that changed its feedback by correctness history.

REFERENCES

[1] Boowa & Kwala, http://www.boowakwala.com/alphabet/alphabet-games-kids.html

[2] Ict Games, http://www.ictgames.com/funny_fingers_v2.html

[3] IXL, http://www.ixl.com/math/kindergarten.

[4] Sheppard Software, http://www.sheppardsoftware.com/math.htm.

[5] Mathematical Markup Language Version 2, http://www.w3.org/TR/MathML2/

[6] Maplesoft, http://www.maplesoft.com/products/maple/

[7] Wolfram, http://www.wolfram.com/mathematica/

[8] F. Alimoglu, E. Alpaydin, "*Combining multiple representations for pen-based handwritten digit recognition*", ELEKTRIK: Turkish Journal of Electrical Engineering and Computer Sciences, pp. 226-239,2001.

[9] L. Anthony, J. O. Wobbrock, "*A light weight multi- stroke recognizer for user interface prototypes*", in Pattern Recognition, vol. 29, no. 4, pp. 641-662, 1996.

[10] E. Brown, "*Developmental characteristics of figure drawings made by boys and girls ages five through eleven*", In Motor Skills, pp. 279-288, 1990.

[11] D.Carlisle, P.Ion, R.Milner, N.Poppelier, Mathematical Markup Language (MathML) Version 2.0, World Wide Web Consortium Recommendation, 2001. Available at http://www.w3.org/TR/MathML2.

[12] K. F. Chan, "*Elastic Structural Matching for Recognizing On-line Handwritten Alphanumeric Characters*", Technical Report HKUST-CS98-07, pp. 1-29, 1998.

[13] K.W. Cheung, D.Y. Yeung, and R.T. Chin, "*Bidirectional deformable matching with application to hand-written character extraction*", IEEE Transactions on Pattern Analysis and Machine Intelligence 2002, vol. 24, no. 8, pp. 1133-1139, 2002.

[14] MP. Dubuisson, AK. Jain, "*A modified hausdorff distance for object matching*", In: 12th International Conference on Pattern Recognition. Jerusalem, Israel, pp. 56-68, 1994.

[15] J. Feder, "*Plex languages*", Information Science, vol.3, pp. 225-241, 1971.

[16] M. Fligner, J. Verducci, J. Bjoraker, and P. Blower, "*A new association coefficient for molecular dissimilarity*", In: The Second Joint Sheffield Conference on Chemo-informatics. Sheffield, England, 2001.

[17] H. Freeman, "*Computer Processing of Line-Drawing Images*", ACM Computing Surveys, vol. 6, no. 1, pp. 57- 98, 1974.

[18] T. Hammond, R. Davis, "*LADDER, a sketching language for user interface developers*", Computers & Graphics, vol. 29, no. 4, pp. 518-532, 2005.

[19] L. B. Kara, T. F. Stahovich, "*An image-based, trainable symbol recognizer for hand-written sketches*", Computer & Graphics 29(2005), pp. 501-517, 2005.

[20] J. Kittler, M. Hatef, RPW. Duin, and J. Matas, "*On combining classifiers*", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 3, pp. 226-239,1998.

[21] P. Kristensson, S. Zhai, "*SHARK2: A large vocabulary shorthand writing system for pen-based computers*", in Proc. UIST 0´4. New York: ACM Press, pp. 43-52, 2007.

[22] G. Labahn, “*MathBrush: A System for Doing Math on Pen-Based Devices*”, The eighth IAPR Workshop on Document Analysis Systems, Nara, Japan, pp. 599-606, 2008.

[23] J. LaViola, R. Zeleznik, “Mathpad2: a system for the creation and exploration of mathematical sketches”, Proceeding SIGGRAPH ’07 ACM SIGGRAPH 2007 courses, 2007.

[24] Y. LeCun, LD. Jackel, L. Bottou, A. Brunot, C. Cortes, JS. Denker, H. Drucker, I. Guyon, UA. Muller, E. Sackinger, P. Simard, and V. Vapnik, “*Comparison of learning algorithms for handwritten digit recognition*”, In International Conference on Artificial Neural Networks, Paris, pp. 5360, 1995.

[25] A. C. Long, Jr., J. A. Landay, L. A. Rowe, and J. Michiels, “*Visual similarity of pen gestures*", in Proceedings of the SIGCHI conference on Human factors in computing systems, ser. CHI '00. New York, NY, USA: ACM, pp. 360-367, 2000.

[26] L. Lotz, H. Loxton, and A. Naidoo, “*Visual-motor integration functioning in a south African middle childhood sample*”, In Journal of Child & Adolescent Mental Health, pp. 63-67, 2005.

[27] E. G. Miller, N. E. Matsakis, and P.A. Viola, “*Learning from one example through shared densities on transforms*”, Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference 2000, Hilton Head, SC, pp. 464-471, 2000.

[28] M. Oltmans, “*Envisioning Sketch Recognition: A Local Feature Based Approach to Recognizing Informal Sketches*”, Ph.D. dissertation, Massachusetts Institute of Technology, 2007.

[29] B. Paulson, T. Hammond, “*Paleosketch: Accurate Primitive Sketch Recognition and*

Beautification", Proc. 13th International Conference on Intelligent User Interfaces, pp. 1-10, 2008.

[30] B. Paulson, A. Wolin, J. Johnston, and T. Hammond, "*SOUSA: Sketch-based Online User Study Applet*", EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling, 2008.

[31] B. Paulson, P. Rajan, P. Davalos, R. Osuna, and T. Hammond, "*What!?! No Rubine Features?: Using Geometric-based Features to Produce Normalized Confidence Values for Sketch Recognition*", VL/HCC Workshop: Sketch Tools for Diagramming Herrsching am Ammersee, Germany15 September 2008, pp. 5763, 2008.

[32] B. Paulson, B. Eoff, A. Wolin, J. Johnston, and T. Hammond, "*Sketch-based educational games: Drawing kids away from traditional interfaces*", Proceedings of the 7th international conference on interaction design and children, ACM (2008), pp. 133-136, 2008.

[33] D. Rubine, "*Specifying gestures by example*", Proceeding SIGGRAPH 91 Proceedings of the 18th annual conference on Computer Graphics and interactive techniques, pp. 329-337, 1991.

[34] WJ. Rucklidge, "*Efficient visual recognition using the Hausdorff distance*", Lecture Notes in Computer Science, vol. 1173, Berlin: Springer, 1996.

[35] A. C. Shaw, "*A formal picture description scheme as a basis for picture processing systems*", Information and Control, vol. 14, pp. 9-52, 1969.

[36] P. Taele, "*Freehand Sketch Recognition for Computer- Assisted Language Learning of Written East Asian Languages*", Masters thesis, Texas A&M University, 2010.

[37] E. Tapia, "*Understanding mathematics: A system for the recognition of on-line hand-written mathematical expressions*", Ph.D. dissertation, the University of Fu Berlin, 2004.

[38] C. C. Tappert, "*Cursive script recognition by elastic matching*", IBM J. of Research & Development, vol. 26, no. 6, pp. 765-771, 1982.

[39] A. Tennant, "*Visual-motor perception: a correlative study of specific measures for pre-school south African children*", In Unpublished masters thesis, University of Port Elizabeth, 1986.

[40] H. M. Twaakyondo, M. Okamoto, "*Structure Analysis and Recognition of Mathematical Expressions*", Proceeding of the third international conference, pp. 430-437, 1995.

[41] S.Valentine, F. Vides, G. Lucchese, D. Turner, H. Kim, W. Li, J. Linsey, and T. Hammond, "*Mechanix: A Sketch-Based Tutoring System for Statics Courses*", The Twenty-Fourth Conference on Innovative Applications of Artificial Intelligence, Toronto, Ontario, Canada, July 22-26, 2012.

[42] F. Vides, P. Taele, H. Kim, J. Ho, and T. Hammond, "*Intelligent Feedback for Kids Using Sketch Recognition*", ACM SIGCHI 2012 Conference on Human Factors in Computing Systems Workshop on Educational Interfaces, Software, and Technology, Austin, TX, May 5- 6, 2012.

[43] B. Wan, "*An interactive mathematical handwriting recognizer for the Pocket PC*", Masters thesis, the University of Western Ontario, 2002.

[44] B. Wan, "*Feature extraction methods for character recognition a survey*", in Pattern Recognition, vol. 29, no. 4, pp. 641-662, 1996.

[45] J.R.Ward, T. Kuklinski, "*A model for variability effects in handprint with implications*

for the design of hand-writing character recognition systems", in IEEE Translation on Systems, Man, and Cybernetics, vol. 18, no. 3, pp. 438-450, 1988.

[46] J. O. Wobbrock, A. D. Willson, and Y. Li, "*Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes*", Proceeding UIST 07 Proceedings of the 20th annual ACM symposium on User interface software and technology, pp. 159-168, 2007.

[47] S. Zhai, and P. Kristensson, "*Shorthand writing on stylus keyboard*", Proc. CHI '03. New York: ACM Press, pp. 97-104, 2003.

APPENDIX A

In Chapter 5, we introduced the survey forms after the user test. Parents fill out the forms when their children finished drawings.

Description/Identification of Survey Item	Answer
1. How many children were present in the session today?	
2. How much time did the children spend using the software?	
3. Does your child know how to draw digits?	
4. Does your child know how to draw characters?	

Table 12. Survey form 1.

Description/Identification of Survey Item	Scale				
9. Does your child have experience with sketch-based application?	1	2	3	4	5
10. The software is easy to understand and use for the children?	1	2	3	4	5
11. The software can help the children to learn?	1	2	3	4	5
12. The children seem to prefer these kinds of interactions over than traditional mouse and keyboard?	1	2	3	4	5
13. The children retain attention for longer periods when using the system compared to similar pen and paper activities?	1	2	3	4	5
14. It will be feasible to use similar games on a regular basis as part of the course content?	1	2	3	4	5
15. The feedback messages of the system were understood by the children?	1	2	3	4	5
16. The feedback messages of the system were appropriate for their age?	1	2	3	4	5

Table 13. Survey form 2.

APPENDIX B

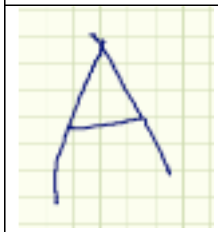
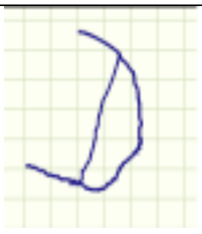
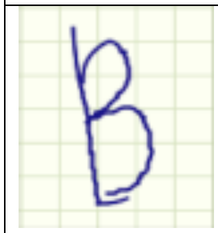
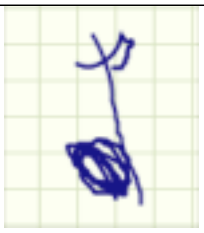
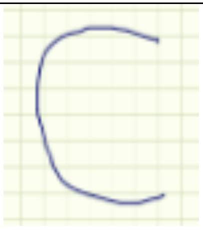
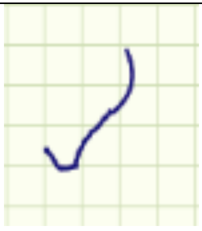
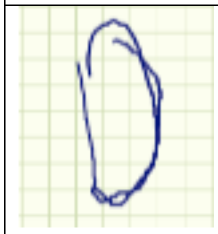
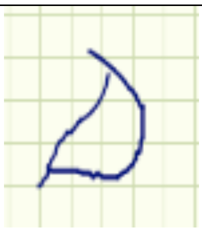
4 years	3 years
A	
	
B	
	
C	
	
D	
	

Table 14. Sketches of 3 and 4 year old children.

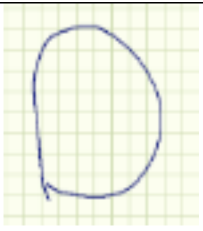
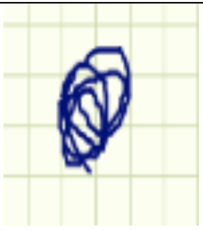
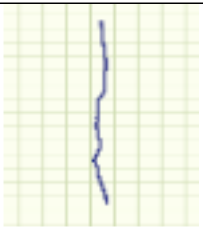
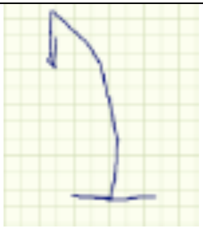
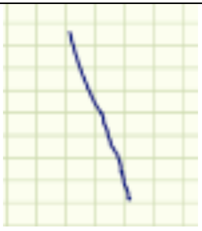
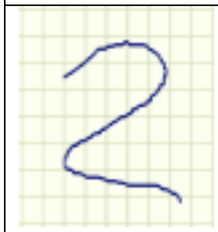
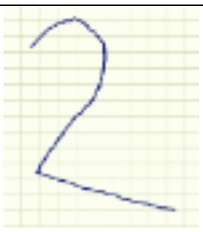
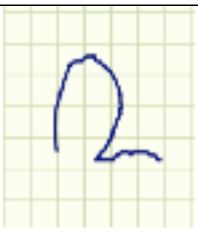
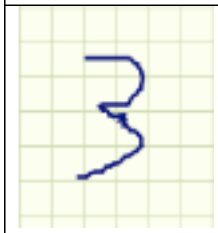
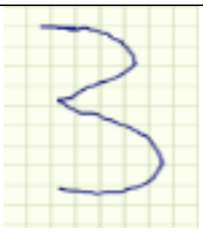
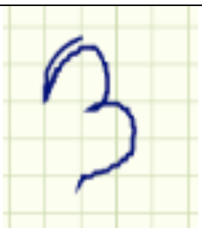
4 years		**3 years**
ZERO		
		
ONE		
		
TWO		
		
THREE		
		

Table 14 Continued.

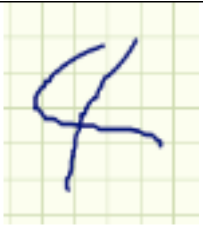
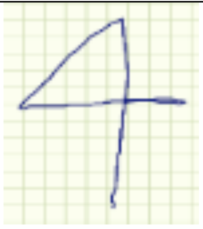
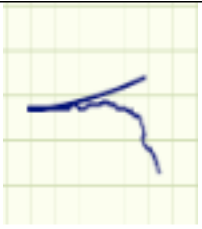
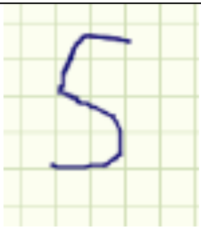
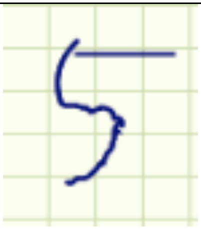
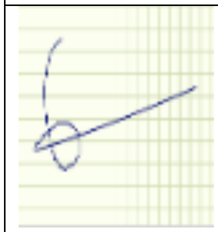
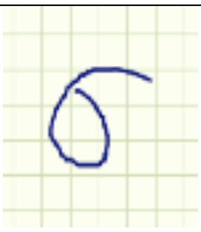
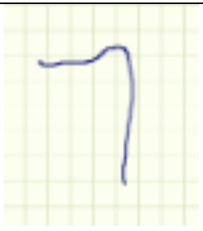
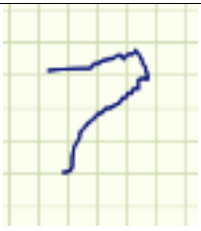
4 years		3 years
FOUR		
		
FIVE		
		
SIX		
		
SEVEN		
		

Table 14 Continued.

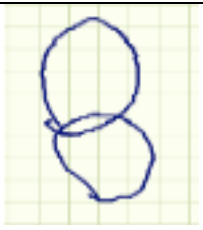
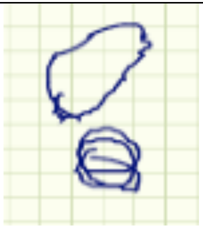
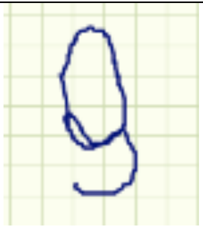
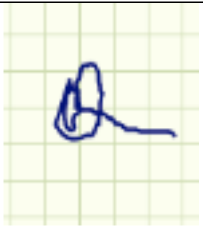
4 years	3 years
EIGHT	
	
NINE	
	

Table 14 Continued.

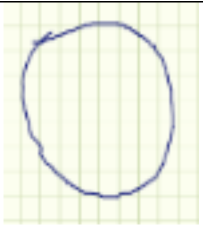
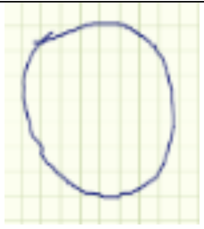
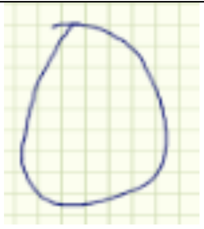
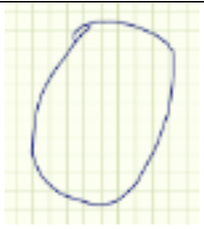
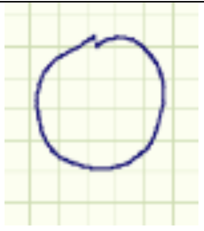
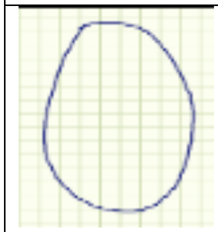
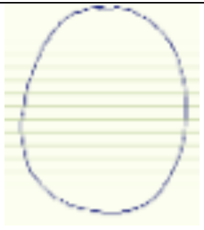
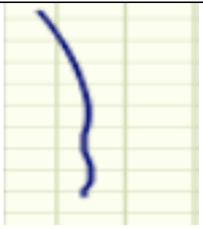
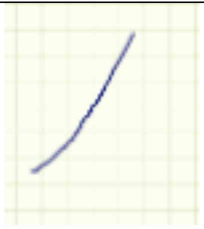
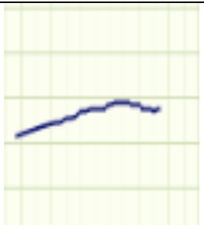
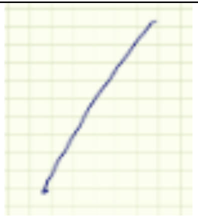
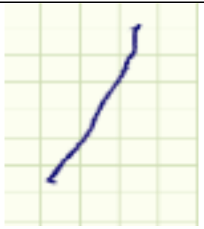
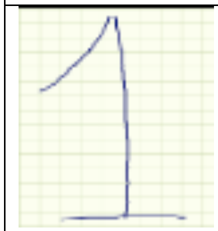
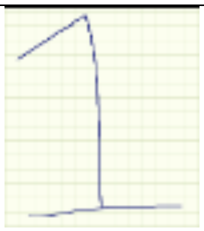
7 years				
ZERO				
				
				
ONE				
				
				

Table 15. Sketches of 7 year old children.

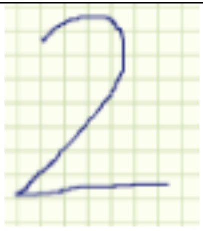
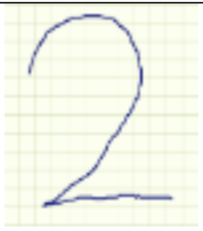
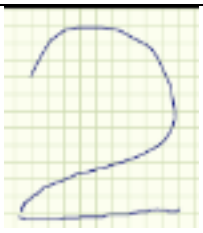
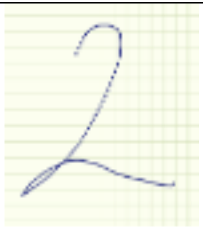
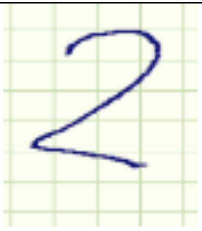
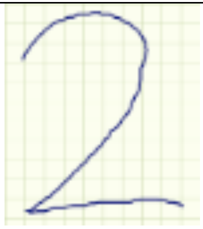
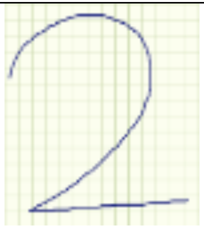
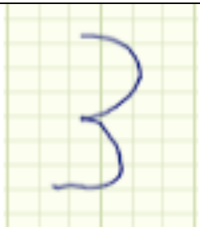
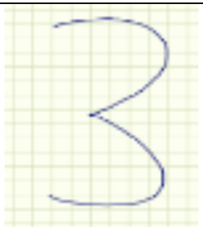
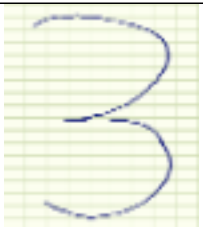
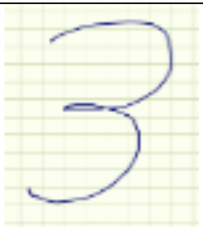
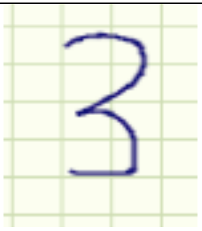
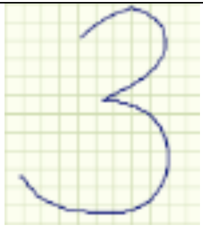
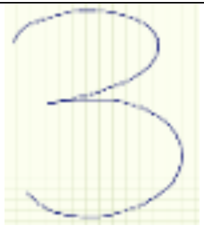
7 years				
TWO				
				
				
THREE				
				
				

Table 15 Continued.

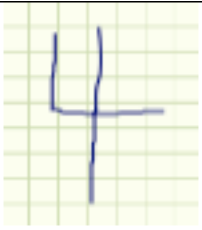
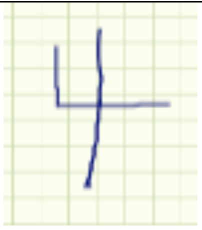
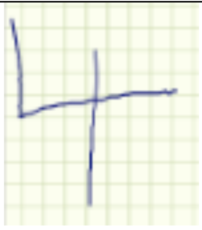
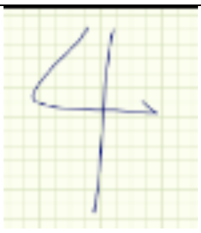
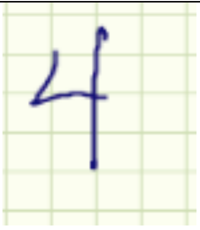
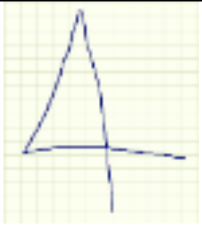
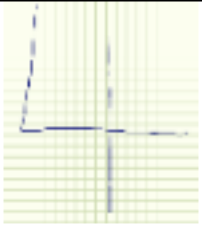
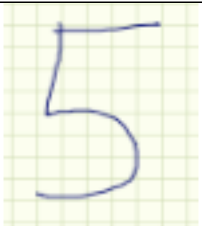
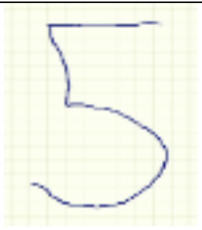
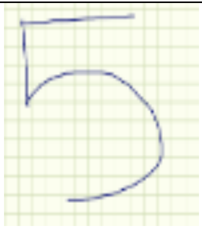
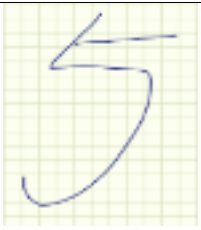
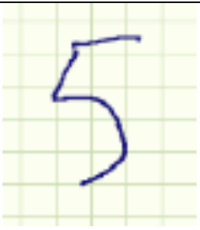
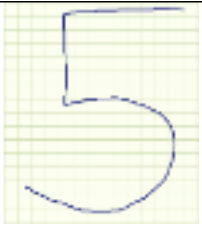
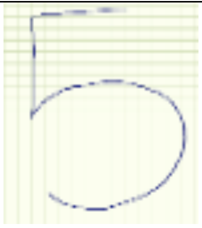
7 years				
FOUR				
				
				
FIVE				
				
				

Table 15 Continued.

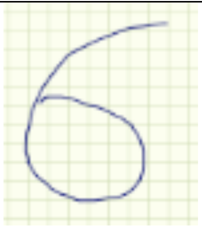
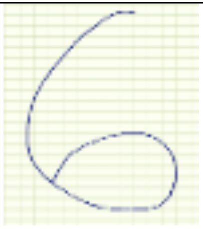
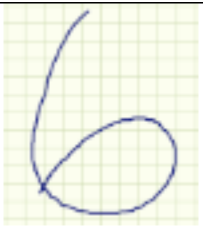
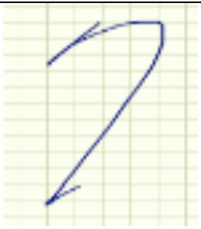
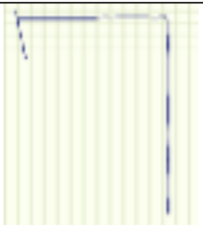
7 years				
SIX				
				
				
SEVEN				
				
				

Table 15 Continued.

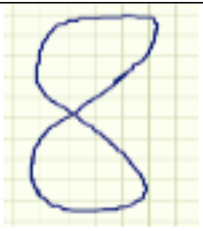
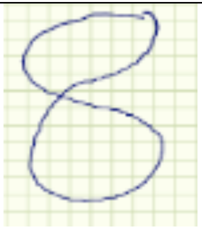
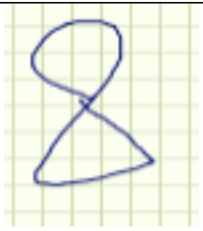
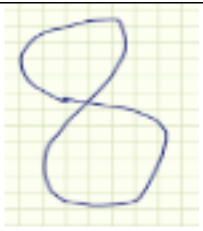
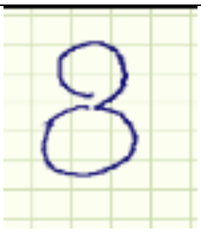
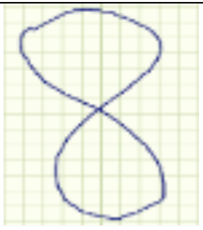
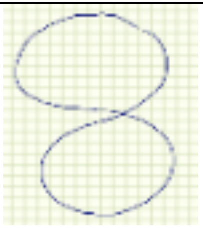
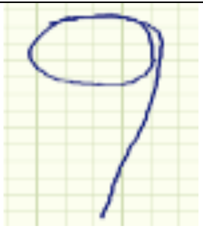
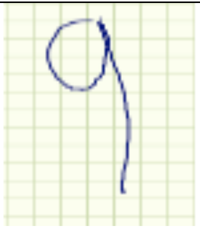
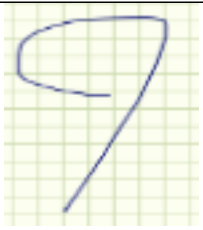
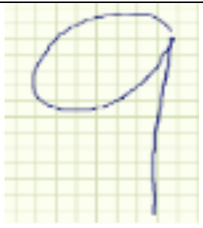
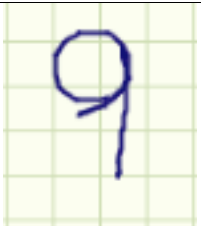
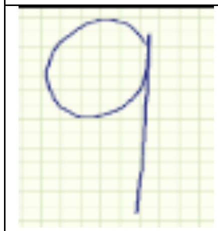
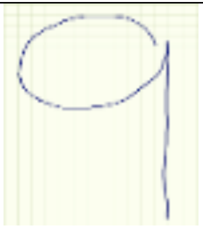
7 years				
EIGHT				
				
				
NINE				
				
				

Table 15 Continued.

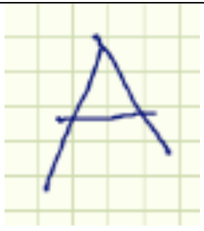
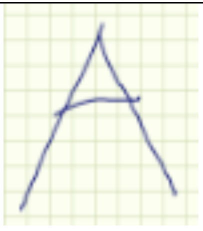
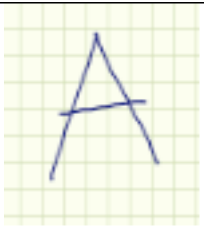
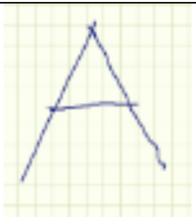
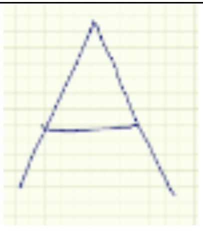
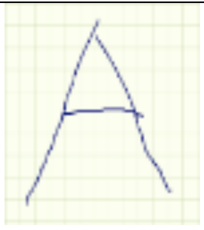
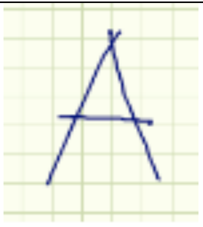
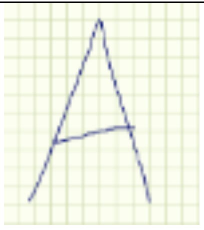
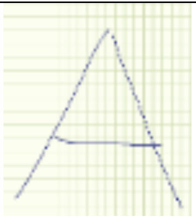
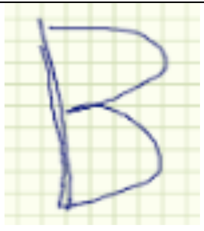
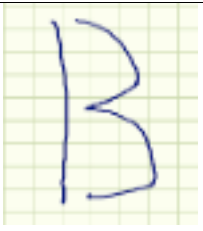
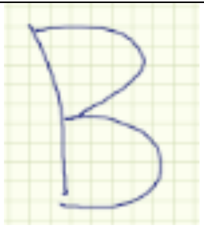
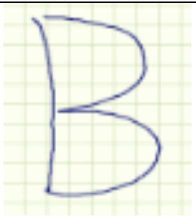
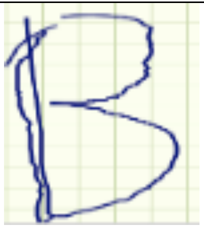
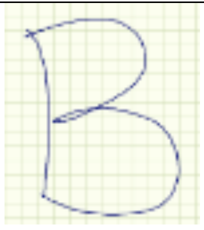
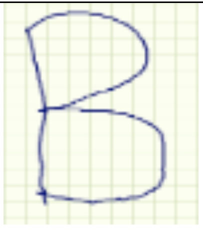
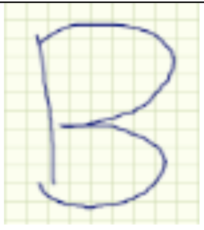
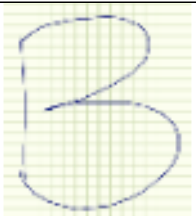
7 years				
A				
				
				
B				
				
				

Table 15 Continued.

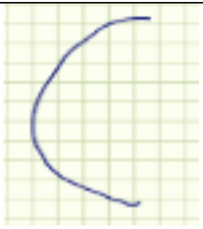
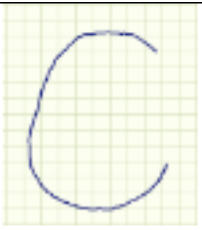
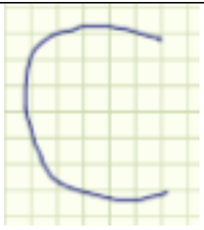
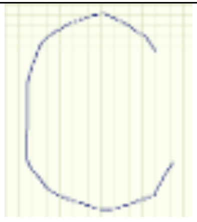
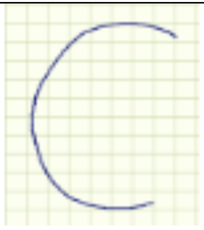
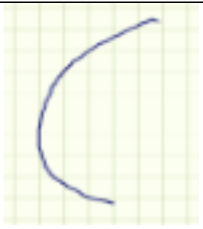
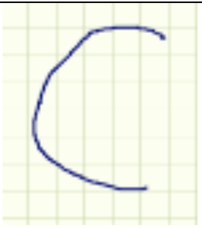
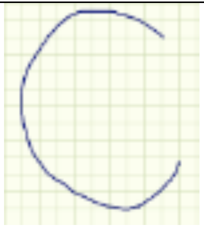
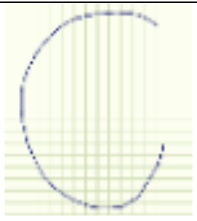
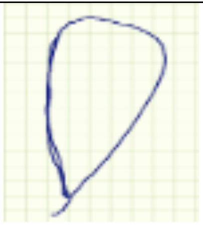
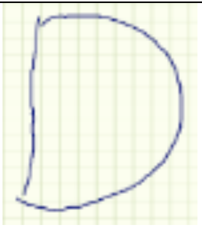
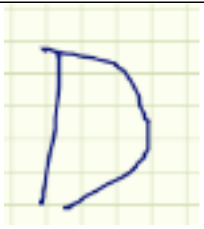
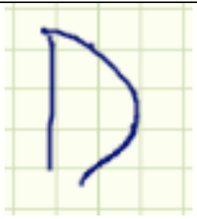
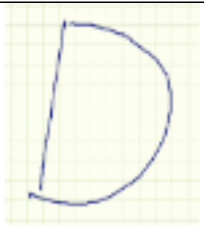
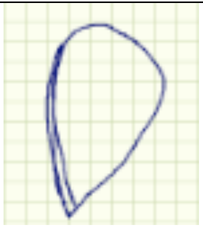
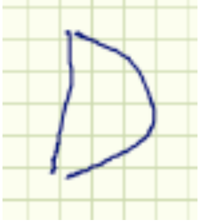
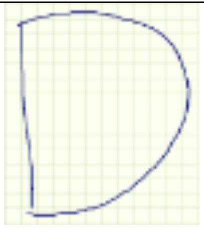
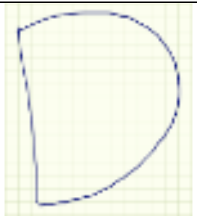
7 years				
C				
				
				
D				
				
				

Table 15 Continued.

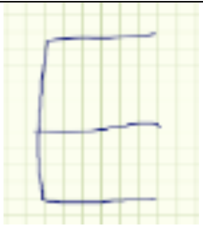
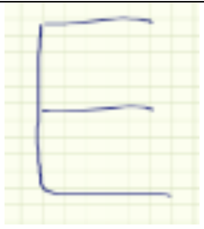
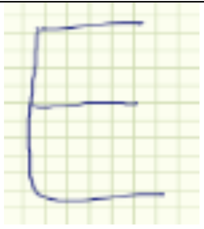
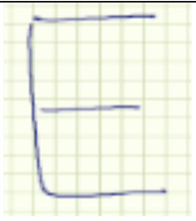
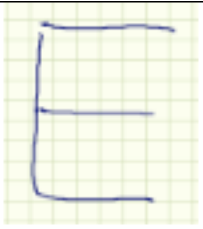
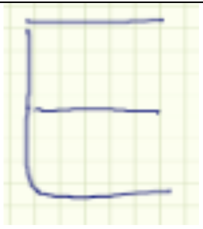
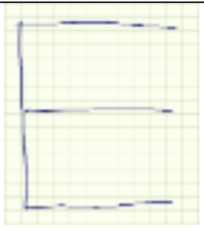
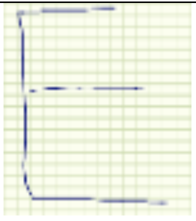
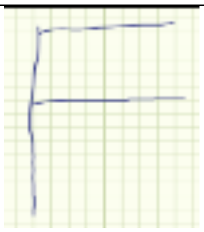
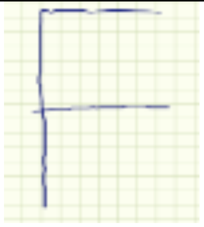
7 years				
E				
				
				
F				
				
				

Table 15 Continued.