A study of the reading of handwriting by machines and humans

Master's Thesis in Cognitive Science

Nijmegen University

Student : P.C.J.Segers

Supervisor : Dr.L.R.B.Schomaker

Date : July 1996

Preface

Thanks to Jan Nijtmans, Janek Mackowiak and Louis Vuurpijl who were always there for me whenever I needed help. Special thanks to Louis for developing the software for the click-density patterns. Many thanks also to my supervisor, Lambert Schomaker, for teaching me all I needed to know about handwriting recognition and for being interested in my work.

Summary

A study has been done on the reading of handwriting by machines and humans. In studying the reading of isolated handwritten words by the stroke-based script recognizer, developed at NICI, it was found that with combining different lexical post-processing methods for machine reading, the performance of a handwriting recognizer becomes significantly better. Especially the top five list of possible solutions will more often contain the correct word.

In studying the reading of isolated handwritten words by humans, we propose a new method which helps to identify and map the features which humans use in reading. With this new method, the subjects themselves can point out which parts of the word are considered important for recognition in a simple point and click setup. Independent variables as global luminance of the word trace can be manipulated.

In comparing machine and human reading, we found that the major difference between machine and human reading is the fact that the computer can not come up with a non-word, which makes the resulting solutions sometimes far from intuitive from the human point of view. This is also the reason why the distance between computer wrong answers and the correct answer is much larger than the distance between human wrong answers and the correct answers. Finally, some suggestions are made in the area of the ergonomics of a pen-based computer. If the first and last letter of a word are correctly recognized, an incorrect solution would be better accepted by the user. Also, if the humans would understand why the computer makes some errors, this would help in the acceptance of computer handwriting recognition.

Contents

Chapter 1

Machine and human reading

1.1 Introduction

When we want to communicate with a computer we currently use a keyboard or a mouse or track ball as a medium. It is expected that in the near future speech and handwriting will also be commonly used for human-computer interaction.

Speech recognition as a medium between humans and computers is very natural, but in an office environment, for example, this may cause difficulties since everybody would be talking to their computers. The use of a pen is also more natural than the use of a mouse or track-ball [30], because we learned this at a young age. An advantage of using a pen is that it is easy to point at something specific on the screen. Using speech for pointing or object selection would be more complex in such a situation.

This thesis concerns handwriting recognition. Many difficulties which are encountered in speech recognition research are the same for handwriting recognition research. Just like among human speakers, there is a great variability between human writers. Everyone has their own personal style, which makes it very difficult to develop a universal handwriting recognizer. Especially when people write in cursive script there are many problems to make the right segmentations [21]. The letters within a word consist of different numbers of strokes and often are connected to each other with an extra stroke. An example is the difference between <*cl*> and <*d*>, which can be very small. This causes words like <*clump*> and <*dump*> to be confused (see figure 1.1). If a computer starts reading a word, already at the first few

Figure 1.1: *The difference between* <cl> *and* <d> *can be very small. This causes words like* <clump> *and* <dump> *to be confused.*

strokes there are many letter possibilities.

The major distinction which can be made in handwriting recognizers is between off-line recognizers and on-line recognizers. In the next two sections these approaches will be described in more detail.

1.2 Off-line handwriting recognition

In off-line handwriting recognition, the recognizer recognises handwriting presented on a piece of paper (Optical Character Recognition, OCR). This type of recognizer can be used in the reading of bank cheques, or can be used as a part of automatic mail-sorting machines. Problems of off-line handwriting recognition are: vagaries of different pen types, wide strokes which frequently overlap and a lack of order information [31] (although some stroke-order information can be determined off-line [1], overlap of stroke and connected letters may pose serious problems). A typical OCR-system consists of the following components [5]:

1. *Optical scanning*: a digital image of the original document is made. A disadvantage is that this cannot be done real-time, at the time of handwriting production.
2. *Document segmentation*: the text has to be found in the scanned image. Problems are that images or backgrounds have to be distinguished from text and that the different regions have to be segmented.
3. *Word segmentation*: problems in finding where one word ends and the next begins.
4. *Stroke or letter segmentation.* A stroke is defined as each ink-segment between two pen lift-ups. There are problems in finding the different strokes or letters.

   After this stage the 'ink' object selection has been done. Subsequent stages are focussed on processing a selected ink object (i.e., either characters or words).
5. *Preprocessing*: smoothing the digitized objects and normalizing size, slant and rotation.
6. *Feature extraction*: capture the essential geometrical characteristics of the objects.
7. *Classification*: identifying the object as being a member of a shape class. This can be done with the help of neural network models, statistical models, etc.
8. *Post-processing*: transforming classified ink objects into words and sentences, making use of a dictionary, syntax, or other forms of high-level context.

Existing OCR-systems have a high recognition percentage in reading machine fonts and reading digits. The performance in reading isolated handwritten characters is reasonable, but reading cursive script is still very difficult [34].

1.3 On-line handwriting recognition

On-line handwriting is a different area in handwriting recognition. A pen-based computer only consists of a flat display which records and displays the movements of the pen tip in time [18]. The advantage of on-line handwriting recognition is that it can be used for interaction with the computer. Recognition is done immediately, in contrast to off-line recognition where the handwriting first has to be scanned (which is time-consuming). On-line handwriting recognition makes it possible to have very small, wireless, portable computers, without a keyboard (Personal Digital Assistent, PDA). Disadvantages are that the user has to adapt to writing on a liquid crystal display (LCD) and that the recognizer has problems when users make corrections in their handwriting, since it uses the time of writing as a parameter.

At the NICI (Nijmegen Institute for Cognition and Information) in Nijmegen, research is being done to develop an on-line handwriting recognition system. This system contains eight major modules or stages [29]:

- Stage 1: *On-line recording of handwriting.* The writing is done on 'electronic paper' consisting of a liquid crystal display plus integrated digitizer.

- Stage 2: *Word-based pre-processing* consisting of low-pass filtering (to remove the noise from the digitizing tablet and physiological and (bio)mechanical sources) and differentiation.

- Stage 3: *Segmentation* into intended movement units ('strokes' - in on-line handwriting recognition a different definition is used than OCR -). This segmentation is based on the writing speed. At the start and end of a stroke, the writing speed is very low, while in the middle of a stroke the speed is high. So if the pen-tip velocity reaches a minimum, this point in time marks a boundary of a velocity-based stroke (VBS). Next to VBSs, white spaces, dots and t-bar crossings are to be found.

- Stage 4: *Normalization* of various motorical degrees of freedom (slant, size, etc.).

- Stage 5: *Computation of feature values* ('feature vector') per stroke which are motor invariants or salient to the human perceptual system, followed by the quantization of stroke shapes using a self-organising Kohonen [16] network. Figure 1.2 displays a picture of a Kohonen net. Each stroke which is written is matched to the prototypical stroke in this network. The identity (i,j) of the best-matching cell is used as a basic stroke code. Thus, the Kohonen network is used as a form of feature vector quantization. Before the recognizer goes to the next step, the original handwriting is 'rewritten' with help of the Kohonen net.

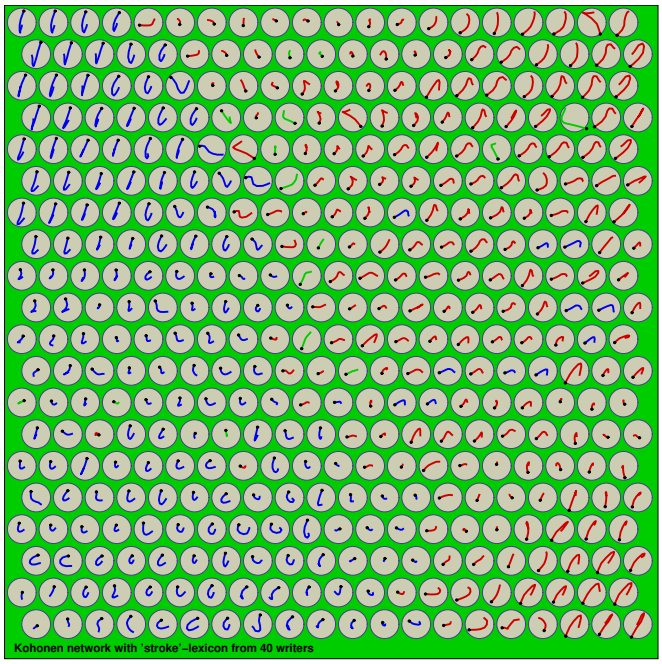


Figure 1.2: *Kohonen network with stroke lexicon from 40 writers.*

- Stage 6: *Construction of allographic (i.e. letter-shape) hypotheses* from sequences of quantized strokes. Figure 1.3 shows an example of this for the word 'breakdown'. For

```
h--]e]a--]h--]r-]  h--] n-] n-]r
l]r]e-]u-]l]s]n-]d--] o]r]o]w--]
b-]o-]oa-]e? a--]c]l] i w--]r]o]
t--]l-?c  t--]r] e]e? v-] o-] r]
k--]a--]d--] a]r]s]b-]u]u-] r-]
b--]l]e]e]t] c]    t--] r-] rr]
t]s]c] o] k--]i-]  k--] a-]r] v]
h-]  d--] h-]o]d---] u-]r]s] i
t-]o]u-]l?t-]i  c-] h-] d-? u-]
f]rr a-] d-]s-]   d-] o-] v]i l]
l i d-? a-?z-] e-]t]n-]r u]s e?
ez]d--] if-]s] l-?b--]r] iz--]
e] u-] o-] b-]o-]o-]k-]s]   jn-]
t]e-]s]r] k-]  a-]t-]i-]   a-]
l-? c]cf]z]   s-]lr]z--]r-]c]
k-]r]i] l s  d-?f] v]i   x-]
a-]c]   e     d-]e   z-]s-]u]
d-? n-] t]    a-?t-] s   i-]
b-] o]d-]        c f-?r-]   r-]
t-] n-]        o-]    s-]  o]r
f-?t-]z          d]    x-]
t] c i          a?    o]
o-]i
v]
z
```

Figure 1.3: *The written word 'breakdown' and the allograph hypotheses from the recognizer.*

this stage a transition network is being used which contains probabilities that two consecutive and consistent stroke interpretations of an allograph follow each other up in an n-stroke allograph (so for example the first stroke of a three-stroke <*a*> is followed by the second stroke of the same three stroke <*a*>, which is followed by the last stroke of the same <*a*>: $a1/3 \rightarrow a2/3 \rightarrow a3/3 \mid a*/3$). In principle, when the allograph is known, the letter-code is known. Confusion can be caused for example between <*0*>, <*O*> and <*o*>, which can have equal allographs.

- Stage 7: *Construction of word hypotheses* from sequences of allographic hypotheses. For this step, a large dictionary is being used.

- Stage 8: *Presentation* of most likely word hypotheses to the user.

A separate mode of operation of this system is the *supervised learning* of the relation between stroke-vector sequences and allographs. Here, at system design time or at the moment of actual system use, human intervention is needed to manually attach letter labels to stroke-vector sequences.

The average performance of this handwriting recognition system is about 75% for hand-written words of handwritings it is familiar with and a bit lower for unfamiliar handwritings. The system is writer-independent and recognizes cursive, mixed-cursive and hand-printed words. Compared to other on-line recognizers the system has an average performance.

This thesis concentrates on the seventh and eighth stage: (1) How to get from letter (allograph) hypotheses to word hypotheses, and (2) how much confusion between words is acceptable for the human users? The basic tenet is, that users will like a handwriting recognizer

better, if the (unavoidable) classification errors are 'intuitive', as opposed to the often bizarre misclassifications of current systems. What we want is the computer to act more humanlike. To achieve this, we have to look at both sides: human reading and machine reading.

1.4 Research on human reading

We first take a brief look at human reading. In psychology there is much research being done on this topic. In 1969, Morton [20] proposed the *logogen* model. Logogens are representations of words in our brains. When a representation is activated enough and a certain threshold is reached, the word is recognized.

McClelland and Rumelhart [17, 25] propose a *parallel* model of word recognition: the interactive activation model. The letters in a word are processed in parallel and lead to identification of the word.

The activation-verification model of Paap *et al.* [22] has a lot in common with the interactive activation model. In their article, Paap *et al.* dedicate a section to the comparison of the two models. Both models can predict and explain the effects of lexicality, orthography, word frequency and priming, although the mechanisms used for these predictions can be different.

Fleming [6] deals with the question whether or not *phonology* contributes to the activation of the meaning of a word during printed-word recognition in the same matter as it does during spoken-word recognition. His experiments show that phonology does not play the same mediating role during printed-word recognition as it does during spoken-word recognition.

Some investigations show that *word shape* information is used to facilitate lexical access [19, 10]. Paap *et al.*[23] on the other hand found evidence that information about visual form is lost early in the process of word recognition, but that word recognition is highly dependent on the identification of abstract letter units. All this research has been done on well segmented, machine printed words.

Handwriting, and especially cursive script contains fused characters. At the NICI, Brassé [2] found that the global contour (see figure 1.4) of words can help in the recognition of *handwritten* words. A frequent word with a unique word contour (with ascenders and

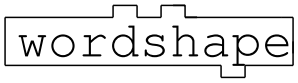


Figure 1.4: *The global contour of a word can help in the recognition of handwritten words.*

descenders) is recognized better than either a less frequent word, a word with a commonly occuring contour, or a contour without ascenders and descenders.

1.5 Use of within-word context in machine reading

Several machine-reading algorithms are based on the above described hypotheses from psychology. In going from letter to word hypotheses, machine reading can roughly be divided in left-to-right reading and parallel reading lexical post-processing methods.

- Left-to-right machine reading methods are methods which are letter-based. The computer reads the word beginning at the first and ending at the last letter, without 'looking' at the whole word. Examples of left-to-right reading methods are:

– *Tree-search.* Wells [38] discusses a 26-way tree, with a route for each letter at every node. To reduce the needed amount of memory, non-existing letter combinations in the recognizer output can be pruned. The tree becomes a representation of the dictionary (see figure 1.5). This method of lexical presentation is called a 'trie'

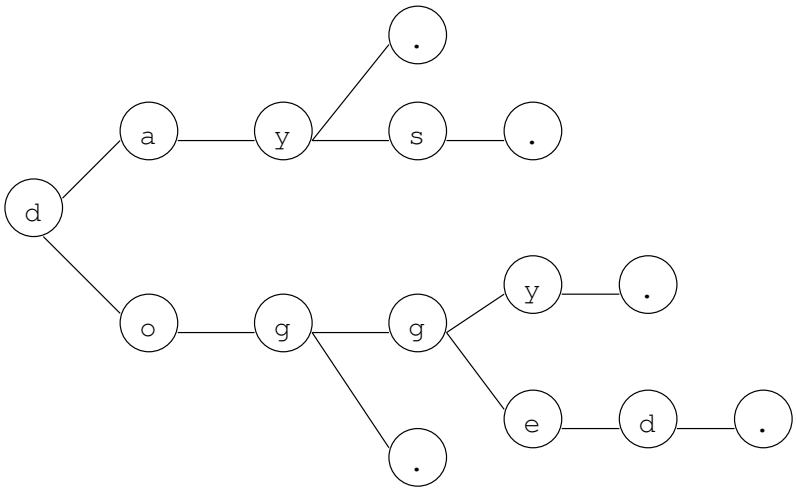


Figure 1.5: *Small tree for the dictionary: dog, doggy, dogged, day, days. A circle with a dot indicates the end of a word.*

[4]. The advantage of this method is that searching is simple, efficient and fast. A disadvantage is that it costs a lot of memory.

– *N-grams for pruning.* The data structure is a letter graph representing the whole word, where nodes are the possible letters [39]. N-grams (bigram, trigram, etc.) are being used to delete non-allowable combinations of nodes (pruning). In a language as Italian there are a limited number of bigrams and trigrams, so this would be a good method. In a language like Dutch, the number of bigrams and trigrams is much larger, which reduces the effectiveness of the method.

– *Markov models.* Guyon and Pereira [9] describe a Variable Length Markov Model (VLMM) as an alternative for the use of n-grams. VLMMs predict the next character given a window of past characters. Unlike n-grams which use a fixed window, VLMMs optimize locally the size of the window. This way, a better performance is achieved, with fewer parameters.

- Parallel machine reading methods are word based. The whole word is considered by looking for example at it's outline shape. Examples of parallel machine reading methods are:

  – *Word-image features.* By making use of the outline shape of the word , the size of the dictionary is reduced considerably [13]. The outline shape of a word is a global feature (every word has an outline shape). Local features (features which are not always present) can also be used, for example crossings, edges etc. Simon refers to these features as singularities [32]. An example of a crossing-feature is given in figure 1.6.

  – *Logogen model.* Higgins and Bramall [14] describe an on-line cursive script recognition system in which the logogen model has been of guiding influence. The system uses a blackboard model in which values are given to candidate words.

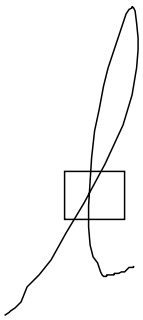


Figure 1.6: *A crossing is a local feature.*

– *N-grams using word-scores.* Words in the dictionary get scores if there is a matching n-gram. The parallel part of this method is the computation of the word-scores.

None of these techniques however produce errors which are 'intuitive' or understandable from the point of view of a human user. For example, <*showman*> can be 'recognized' as <*lieutenant*>.

At the NICI in Nijmegen, the above described ideas have been used to implement four post-processing methods, which need to be evaluated. Also the idea of local features is very interesting, because it has not yet been investigated if humans also make use of these features. In the next section will be described what kind of research will be done in this project.

1.6 Description of the research project

This thesis can be divided into two major parts:

1. Research on machine reading (chapter 2).
2. Research on human reading (chapter 3).

In chapter 4, a comparison is made between machine and human reading.

1.6.1 Machine reading

The next chapter starts with an introduction of several post-processing methods. We want to compare these methods of machine reading, and contrast the errors which are produced with human reading errors. At the NICI currently four different methods are being developed and need to be investigated:

1. Tree-search: a method which makes use of binary search to test if a word exists.
2. Fuzzy word matching: a method which uses word scores.
3. Trigram matching: a method which also uses word scores.
4. Word contour: a method which makes use of lexical subsets.

Word contour for example reduces the dictionary to a list of most probable words by making use of the ascender and descender information in the global word contour. The methods will be described in more detail in the next chapter.

For the case of isolated characters it has been shown that combining methods gives better results [12]. Powalka, Sherkat and Whitrow [24] performed experiments which proved that combining recognition results at the word level also improves the recognition rate. We think it could be very useful to combine left-to-right machine reading methods with parallel methods. The brittleness of a left-to-right method is after all that if one letter in the word is not recognized, the correct solution can not be found anymore. Combination with a parallel method may help to solve this problem. Combination algorithms are currently under development.

A test environment will be prepared in which these four methods can be compared. There is enough test material at hand at the NICI, but a suitable test set has to be composed.

After the preparation, we can run the tests. First, every method will be tested separately. Special attention will be paid to what kind of errors every method makes. It is not useful to combine methods which make the same errors, but it will be tried to combine the right methods to get an optimal recognition percentage. If all the tests are done, the results can be evaluated. An in-between evaluation may be necessary when all the methods have been tested separately.

1.6.2 Human reading

This part of the project starts with a literature study. Findings concerning the different psychological hypotheses will be discussed in chapter 3.

In order to make a comparison with the machine results, an experiment with human subjects will be prepared, viewing the human reader as 'one of the recognizers'. The test material will be the same as for the machine reading experiment. This means that the humans have to read the words from a computer screen, since the words were written on a LCD-integrated digitizing tablet (recorded at 100 samples per second).

From results of an experiment on human recognition rates of handwritten words on paper (see table 1) [28], it can be expected that even humans will not have a perfect recognition score on this material, so enough errors will be made to compare with the errors made with machine reading. With the experiment we want to find out which and how many errors humans make in reading the same test set of isolated words as the computer had to read.

Table 1.1: *Human recognition rates of handwritten words on paper. Experiment A: single-word recognition, B-D: three-word sequences, middle word recognition [28].*

Exp.	*Context*	*Style*	*Writers*	*Target words*	*Readers*	*Words, recognized*	
A	frequently-used Dutch words	handprint	1	30	12	360	98%
	frequently-used Dutch words	neat cursive	1	30	12	360	88%
B	sentence fragments, same writer	cursive	3	15	12	180	85%
C	sentence fragments, same writer	cursive	4	13	20	260	85%
	unrelated words, same writer	cursive	4	13	20	260	77%
D	unrelated words, same writer	fast cursive	12	12	15	180	72%
	unrelated words, different writers	fast cursive	12	12	15	180	54%

Also we want to find out in the same experiment, which features humans need/use to recognize a word. There has been done a lot of research on feature detection, we will present a new method which uses the subject's point of view instead of that of the researcher.

The words will be hardly legible when presented to the subjects. They have to indicate themselves which part of the word they need to be able to recognize it.

In chapter 4, the differences between the results of the human recognition and the computer recognition will be discussed. If the integrated computer method would make the same errors that humans make, we have a device which would be more acceptable for humans, because the errors would be 'understandable'.

1.6.3 Research questions

With the results from the experiments the following research questions can be answered:

- What combination of machine reading methods has the best results?
- Which features do humans use/need to recognize a word?
- Do the errors which the computer makes relate to the errors humans make?

The results of this study may help to improve the quality of the interface of a handwriting recognizer. In practical interfaces, we observed the interesting phenomenon that the human user does not accept counter-intuitive word hypotheses coming from the recognizer. If we can produce a lexical post-processing method which produces an order of word hypotheses which is comparable to the word confusion matrix for humans, the prediction is that users will also consider the system as 'more usable'.

Chapter 2

Experiments on lexical post-processing in machine reading

2.1 Introduction

As has been described in the first chapter, this chapter starts with a description of four lexical post-processing methods. These methods are the connection between letter and word hypotheses in the recognition of handwritten words.. The methods described are being developed at the NICI and up till now, not much research has been done to compare the different methods and to see how a combination of the methods can lead to better recognition results.

In section 2.2 the four lexical post-processing methods will be explained. In section 2.3 the data-set which is used to test the methods is described. Section 2.4 describes which method is used to obtain results and sections 2.5, 2.6, 2.7 and 2.8 show the results of comparing different methods. The final sections of this chapter give a discussion and summary of the results and a conclusion.

2.2 Description of four lexical post processing methods

2.2.1 The whole word method: hard matching

Hard matching is a deterministic tree-search method. It is a left-to-right machine reading procedure, which uses a best first strategy to match the word to be read with a word in the lexicon. First, a letter hypotheses graph is being made (see figure 2.1). The nodes in the graph represent the letter hypotheses, the arcs represent the letter transitions. Subsequently the letter hypotheses graph, a (partial) word-existence test is being performed, with use of binary search to scan the lexicon.

2.2.2 The partial string approach: trigram matching

This method is partly left-to-right and partly parallel. Based on the number of strokes, the word is divided into three zones (see figure 2.2). Trigrams are constructed from the letter hypotheses. Every word in the lexicon which has one of these trigrams in the expected zone gets a score. Also word contour information is used. This means that if the trigram contains an ascender or descender which it should have, based on the ascender-descender pattern, it gets a bonus score which is now set at 1.3 times the normal score.

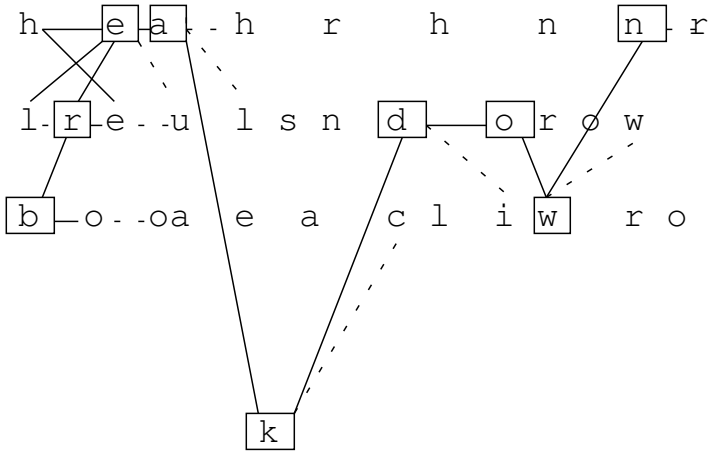


Figure 2.1: *Letter hypotheses graph from the first few lines of the tree from figure 1.3 [11].*

Figure 2.2: *The word is divided in three zones, based on the number of strokes.*

The pseudo-code of **trigram** matching is as follows:

```
forall trigrams klm in lethyp space {
    j = zone-of-trigram-in-input
    forall words in lexicon{
        wordscore[w] := wordscore*(Min(Ne,Nlex)/Max(Ne,Nlex))
        i = zone-of-trigram-in-current-word
        d = abs(i-j)
        wordscore[w]+ = triqual[klm] * (1/(1+d))
    }
 }
```

Triqual is the combined quality of letter hypotheses, this trigram. The letter combination *klm* is letter triplet k,l,m. *Ne* is the number of expected letters from the amount of input strokes. *Nlex* is the number of letters of the word in this lexicon.

The weighting of the wordscore is done in order to compensate for the difference between estimated number of letters in the input and in the current word of the lexicon.

2.2.3 Individual letter presence: fuzzy matching

Fuzzy matching is a method which looks at the whole word. The word is divided in zones (number of zones is number of letters). The zones do overlap eachother, because it is not known where one letter ends and the next begins (see figure 2.3). For every word in the lexicon which has one of the letter hypotheses in the expected zone, the score is incremented.

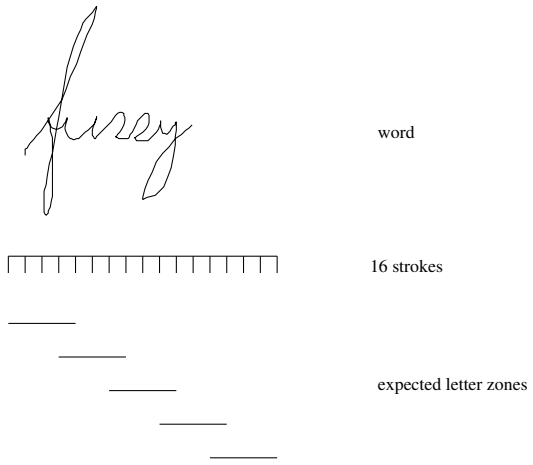


Figure 2.3: *The word is divided in zones (number of zones is number of letters). The zones overlap because it is not known where one letter ends and the next begins.*

The pseudo-code of **fuzzy** matching is as follows:

```
forall letters l in lethyp space {
    j = zone-of-letter-in-input
    forall words in lexicon{
        wordscore[w] := wordscore*(Min(Ne,Nlex)/Max(Ne,Nlex))
        i = zone-of-letter-in-current-word
        d = abs(i-j)
        wordscore[w]+ = fuzqual[l] * (1/(1+d))
    }
 }
```

Fuzqual is the combined quality of letter hypotheses.

2.2.4 Usage of word contour information: the ascordesc method (trained and guessed)

Ascordesc (an abbreviation which combines 'ascender', 'corpus' and 'descender') is a method which comes before **hard**, **trigram** or **fuzzy** matching. Word contour information is used to generate a subset of the lexicon which is sent to (one or a combination of) these three matching methods. The symbolic **ascordesc** pattern is derived on the basis of stroke heights and vertical positions of the input word. The **ascordesc** method comes in two variants: **ascordesc**-trained and **ascordesc**-guessed.

The difference between **ascordesc**-trained and **ascordesc**-guessed is that in **ascordesc**-trained, the generated **ascordesc** code is compared with the **ascordesc** (or contour) codes of lexical subsets which contain the names of words which have been actually written by writers contributing to a training set. In these lexical subsets is stored, for example, that <*dog*>, <*clog*>, <*day*> etc. can be of the form <*xlxj*> (x for corpus-strokes, l for ascender strokes and j for descender strokes).

Ascordesc-guessed, on the contrary, makes use of a grammar (see table 2.1) by which every word in the lexicon gets his or her own **ascordesc**-guess-patterns. For example <*handwriting*> has among others the pattern <*lxxxxlxxxlxxxj*>. This pattern is compressed to <*lxlxlxj*>.

Table 2.1: *A grammar for word contour information. x = corpus, l = ascender, j = descender.*

a	x	A	l	m	x	M	l
b	lx	B	l	n	x	N	l
b	xlx			o	x	O	l
c	x	C	l	p	jx	P	l
d	xl	D	l	p	xjx		
d	xlx			q	xj	Q	l
e	x	E	l	q	xjx		
e	l			r	x	R	l
e	xl			s	x	S	l
f	l	F	l	s	lx		
f	xl			s	xlx		
f	j			t	l	T	l
f	xj			t	lx		
g	xj	G	l	t	xlx		
h	lx	H	l	u	x	U	l
h	xlx			v	x	V	l
i	x	I	l	w	x	W	l
i	l			x	x	X	l
i	lx			y	xj	Y	l
i	xlx			y	xjx		
j	j	J	l	z	x	Z	l
j	xj			z	lx		
k	lx	K	l	’	x	&	l
k	xlx			‘	x	.	x
l	l	L	l				
l	xl						
l	xlx						

Figure 2.4 shows a schematic overview of the difference between the **ascordesc**-trained and **ascordesc**-guessed methodes.

2.3 Data collection

The above described four methods (**hard**, **trigram**, **fuzzy** and **ascordesc**) are going to be compared. The dataset which is used for this comparison was already available at the NICI. It was collected in a collaboration between Hewlett Packard (HP) and the NICI. This section describes what the dataset looks like and how it is recorded. A version of this text already appeared in papers of a workshop [27].

2.3.1 The dataset

Criteria

The data set to be collected:

- Had to capture style variation among writers,
- Had to capture style variability within a writer, as measured at occasions sufficiently spaced apart in time,
- Had to be large enough to allow for a number of large-scale training/testing experiments,

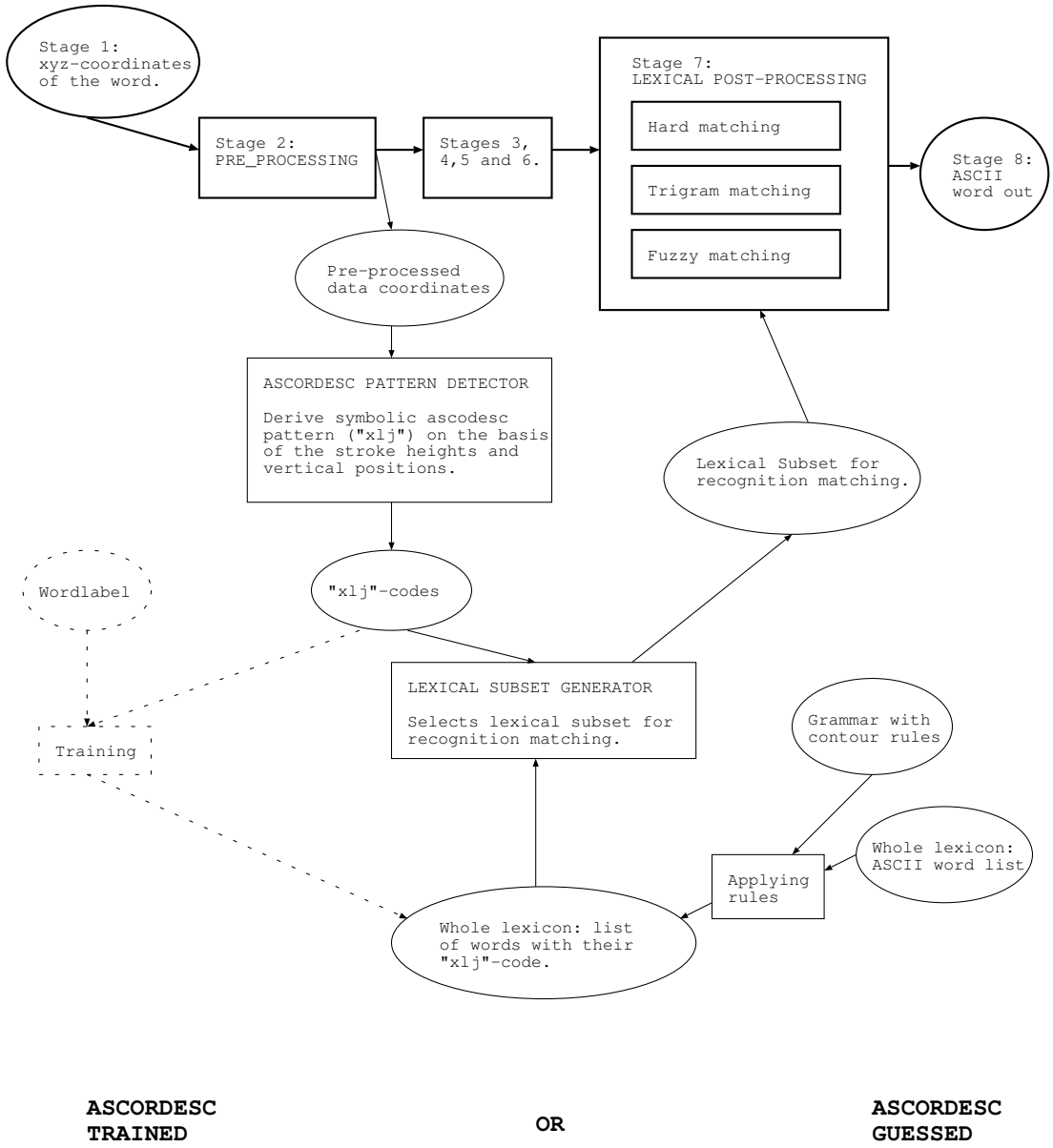


Figure 2.4: *The difference between* **ascordesc**-*trained and* **ascordesc**-*guessed is in the origin of the whole contour-based lexicon, which is trained by the user (left side of the figure) or inferred via a grammar (right side of the figure). Stages 1 to 8 of the recognizer refer to paragraph 1.3.*

- Had to be of high quality as regards the signal properties, since deteriorated signal conditions can easily be imposed post hoc.

Additional constraints: input unit

The data collection is *word*-oriented, since recognizers at both HP and NICI are based on isolated scribble or word recognition. Also, this is the input chunk size currently handled by most free style or connected cursive recognition systems. The *letter* level is only suited for isolated handprint and digit data. The *sentence* level and higher (*paragraphs, pages*) impose

additional word segmentation problems which are difficult to handle at the moment. It is not completely possible to compute word segmentation on the basis of bottom-up features like white space or ink clustering: often lexical or even syntactical top-down information would be necessary to disambiguate here. In many applications, however, the word-based input is already useful, especially if recognition speed can be fast enough to not disturb the human word production process ('train of thought') [33]. The *words* will consist of lower case characters.

Additional constraints: word lexicon

The elements of a word list in handwriting collection setups is usually a subject of hot debate due to the large number of possible criteria for inclusion (size, word length, character content, digram content, trigram content, linguistic frequency of usage, etc.). In the collection setup, two basic constraints were chosen, sacrificing some other criteria:

1. *Bilinguality*: the list must be bilingual in the sense that the same list can be written by Dutch and English writers. This allows for the incremental collection of words in both Nijmegen and Bristol. It will ensure that the Dutch writers will not feel uneasy writing a foreign language.

2. *Maximum Digram Coverage*: in connected-cursive and mixed-cursive handwriting, the current character shape is determined by both predecessor and successor. The connecting strokes come from a previous character, retaining effects from the starting position and the angular velocity (clockwise, sharp, counter-clockwise), and may exert an effect on the first strokes of the current character itself. Similarly, the anticipation of the next character may lead to distortions of the final stroke(s) of the current character. To obtain a reliable overview on character production strategies, as much digrams from the 26x26 transition matrix must be present in the word list. Actually, there are 27 symbols, including the space symbol (identifying Begin-Of-Word and End-Of-Word conditions).

In order to build a word list that fulfills the aforementioned criteria, the following approach was taken:

- Word List 1: 50k Dutch words.
- Word List 2: 50k English words.
- These two word lists were ran through the Unix *comm* command, yielding a list with 3251 words common to both languages.
- As the resulting list was too large for the data collection process, it condensed with a dedicated program in C which created a subset of words with the criterion of maximum digram coverage. This means that all (27x27) digrams present in the input list will be present in the output list. The program is based on stochastic optimization, iteratively picking a word from the input list with a low probability, and only adding it to the output list if it contains new unseen digrams. This was done several times, choosing a final list which was acceptable (decency, not too difficult to spell, etc.). The resulting word list contained 210 words. Due to the selection algorithm, the words are slightly longer than average English words.

- A number of words was manually added because of their interesting (but low frequent) digrams. An example is the <*x*-*y*> digram in 'xylophone'. For this word, the English spelling was used which is more acceptable to Dutch writers than 'xylofoon' would be for English writers. The final list consists of 210 words (Appendix I).

The word list contains many international concepts (e.g., 'algebra'), geographical names, technical terms, Latin-origin words, French-origin words, as well as words which happen to be spelled the same in both languages, but may have a different meaning ('trekking'). After the writing sessions, the subjects were asked from which (unmentioned) language they thought the word list was, and also they were asked to mark words which they thought were difficult to write. The list appears to be of medium difficulty, and there were no specific complaints by the subjects.

2.3.2 Recording setup

Session

Since a representative 'real-life' application does not yet exist, it was decided to collect words in a visually prompted word setup with a provision for rewriting words the subject considers badly legible him/herself. Words are randomized on each session. Writers sat at a table in a room with dimly lit fluorescent lamps to prevent glare from the Wacom PL-100V LCD screen. The Wacom was placed on a normal desktop in an orientation preferred by the subject. A separation panel was placed between experimenter and subject to prevent additional stress or performance pressure which often develops in experimental setups. Subjects are eager to please experimenters, and sometimes weary of hidden motives (intelligence or personality tests). For our purpose it was important that writers used *their own*, i.e., their mostly-used handwriting, rather than a style they thought was acceptable. There was an introductory text on a sheet of paper, and writers were allowed to get accustomed to the setup by writing 20 habituation words.

Session schedule

The subject came to the lab three times (Sessions), spaced two weeks apart. At each Session, two Sets of the 210 words were produced, yielding six Sets (totalling 1260 words written per writer). Within a Set, the writer was allowed to pause after about 100 words.

Session 1:
- Set 1
- Set 2

(at least two weeks)
Session 2:
- Set 3
- Set 4

(at least two weeks)
Session 3:
- Set 5
- Set 6

Recording hardware

- PC: IBM 486SLC2-66 MHz motherboard, 4 MB.

- Tablet: PL-100V
- 3COM 3C509 Ethernet adaptor.

Tablet details are contained in the UNIPEN files.

Subject group

In this data collection setup, at first we tried to avoid the usual population of co-researchers and students. The target group was older than 20 years, and a number of professions in which writing is a usual activity was included. This was done by recruiting people through a newspaper advertisement in a medium-sized Dutch paper. The population was completed to 42 subjects with students and researchers. The average age is about 29 years (varying from 20 to 61). The group consisted of 24 male subjects and 18 female subjects. Handedness L/R is distributed proportional to the whole population (approx 1 in 8 is left handed). 17 of the subjects have finished university, 14 are students, the rest has various backgrounds. The majority of the subject wrote mixed cursive (26), according to their own judgment. The others claimed to write cursive (12) or hand print (4). (They were shown four words samples from the categories Handprint, Mixed cursive, and Cursive).

2.3.3 Data for training and testing

Five of the six sets of each of the 42 writers were used to train the Kohonen stroke network. A sixth set was used for testing. It was taken care that the sets used for training consisted of an equal amount of sets written in different sessions of the collection process.

In training the Kohonen network, the data of the 42 writers was combined with handwritings (about 30 writers) which were collected earlier in the Esprit II Papyrus project at NICI. The handwriting recognizer which is used for the experiments is named VHS V11.3.

2.4 A method for testing different post-processing methods

2.4.1 What is the difference between the post-processing methods?

There are four post-processing methods which need to be investigated: **hard**, **trigram**, **fuzzy** and **ascordesc** (guessed and trained) matching. **Hard** matching can be seen as a basic method. On it's own it has fair results. **Trigram** and **fuzzy** matching use 'tricks' to get results, but on their own their recognition is not very well. In combination with other methods they can probably help to improve recognition. **Ascordesc** is of another category, since it is a method which does a preselection on the lexicon and then uses **hard**, **trigram** and/or **fuzzy** matching.

2.4.2 A description of the experiments

In the first experiment, **hard**, **trigram** and **fuzzy** matching will be compared. Recognition results of each method separately will be obtained and also combinations of the methods will be investigated. The results of this experiment are described in section 2.5. The second experiment compares **hard** matching to **ascordesc**-trained and **ascordesc**-guessed. The results are printed in sections 2.6 and 2.7. A final experiment combines the first two experiments to see if this helps to improve the recognition rate. This experiment is described in section 2.8.

All the different methods and strategies will be put together in a summary, to get a general view. The final sections of this chapter give a discussion and conclusion.

2.4.3 Three methods for combining post-processing methods

For the combination of the post-processing methods, three different combination methods are used:

1. *Rank-sort combination*
   The first combination technique which is used is not very complicated. For every word, every method has produced a rank number from 1 to 30 in the list of possible solutions. We now attribute 30 points to a word with rank 1, 29 points to a word with rank 2, etc. A word which is not recognized at all (solution '????') gets null points. This means for example that if a word <*bobby*> is correctly recognized by **hard**, it gets 30 points. But if that word <*bobby*> is recognized as <*lolly*> by **fuzzy** and **trigram** at places three and four, <*lolly*> gets (28+27)=55 points. So <*bobby*> would be 'recognized' as <*lolly*>. The advantage of this combiner is that there is a kind of *majority voting*. In this method, use can be made of the knowledge that some recognizers are better than others. That is, the method is order dependent. In case of a tie, words are entered in an order from best to worst recognizer origin. In this thesis, the symbol % will be used for this combination method.

2. *Rank-sort with weighting*
   Another combination technique adds a weighting to the above described combiner. The scores for the best performing method will be multiplied by 3 and the scores for the second-best method will be multiplied by 2. In this thesis, the symbol * will be used for this method.

3. *Stand-in* or *'defaultory'*
   Stand-in is a very simple method. There are cases where **hard** matching does not have any solution. If in those cases **trigram**, **fuzzy** or **ascordesc** are used as a stand-in, the results can only improve, or in the worst case, remain the same.

2.4.4 Statistical analysis

On the NICI handwriting recognition home page, Schomaker explains which method to use and why, to compare the results from two versions of a recognizer (http://www.nici.kun.nl/clbinom.html):

> "When using the same test set samples in testing two recognizers, the obtained recognition rates are not statistically independent. This raises the question whether a test for dependent data, such as Chi-square test for correlated (dependent) data must be used. This issue can be solved by realizing what is the goal of the statistical test. If one is (pragmatically) interested in the recognition rate levels of recognizer A vs B as such, then the underlying differences between recognizers may be ignored, and a test for independent data (binomial, normal or Chi-square) is suitable. On the other hand, if one is (scientifically) interested whether the recognizers can be considered as two versions of the same signal generator or are in fact different signal sources, then a test for dependent data must be used. Consider for example the extremal case were two recognizers have a recognition rate of 50%. Applying a statistical test for independent data clearly yields

> a non-significant difference. However, such a test misses the fact that each recognizer may have recognized a distinct subset of 50% of the characters (or words) in that test set. In such a case, we cannot consider the recognizers as having "similar performance". The combination of these recognizers would have yielded a highly desirable 100% recognition rate."

It was found that there is no interaction between writers and different versions of the recognizer. This means that a Chi-square test for independent data can be used if we want to see if there are statistically significant differences between the obtained results. For all the tests an α of 0.01 is used, the number of degrees of freedom (df) is 1, which means that the critical value of χ^2 is 6.64 for a bi-directional test.

2.5 Experiment I: comparison of hard, trigram and fuzzy matching

2.5.1 Recognition results of each method

The first test consisted of testing each method apart. The results of this first test are plotted in figure 2.5 (for exact numbers, see table 2.2). On the horizontal axis of the figure, we see the position of the recognized words. If this position is one, the percentages refer to the top-word recognition of the method. If, for example this position is five, it means that the correct word has to be on position five or lower (position four, three, two or one) in the list of possible solutions. The vertical axis represents the recognition percentage. From this figure we can learn several things:

- **Hard** matching seems to have reached its top in the top 10 recognition. If a top 20 recognition would be used, the results would not get much better. The line is almost horizontal.

- **Fuzzy** matching has a very low top-word recognition percentage, but the line keeps climbing.

- The same goes for **trigram** matching. The top-word recognition percentage is higher though, and the results do not increase as much as for **fuzzy**.

- **Trigram** and **fuzzy** match have not reached their maximum. At the tenth position, the results are still climbing. **Fuzzy** is climbing harder than **trigram** in the end. The results give an indication that if enough hypotheses would be made, **fuzzy** and **trigram** will eventually have the correct word in their list, if no thresholding in word hypotheses quality is performed.

In the histogram in figure 2.6 is shown how the recognition rates for **hard** matching are divided between the 42 writers. Ten writers have a score between 160 and 165 recognized words out of 210. Three writers are below 100 recognized words, and only two writers are above 185 recognized words out of 210. We learn from this histogram that there are 7 out of 42 writers from which the results are much worse than for the rest. These writers suppress the recognition rate considerably.

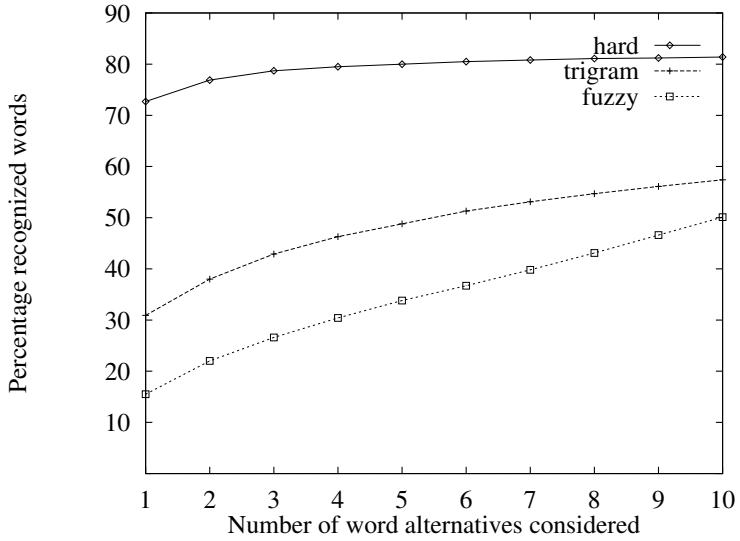


Figure 2.5: *Comparison between* **hard**, **trigram** *and* **fuzzy** *matching.*

Table 2.2: *Recognition results (percentages and standard deviation) for* **hard**, **trigram** *and* **fuzzy** *matching. Pos.rec.wrd.=position of the recognized word, H=***hard**, *T=***trigram**, *F=***fuzzy** *matching.*

Pos.rec.wrd.	*H % (s.d.)*	*T % (s.d.)*	*F % (s.d.)*
1	72.7 *(14.4)*	30.9 *(10.0)*	15.5 *(07.6)*
2	76.9 *(14.1)*	38.0 *(11.4)*	22.0 *(09.3)*
3	78.7 *(14.0)*	42.9 *(12.1)*	26.6 *(10.6)*
4	79.5 *(14.0)*	46.3 *(12.3)*	30.4 *(11.1)*
5	80.0 *(13.9)*	48.8 *(12.6)*	33.8 *(11.7)*
6	80.5 *(13.8)*	51.3 *(12.8)*	36.7 *(12.1)*
7	80.8 *(13.8)*	53.1 *(13.0)*	39.8 *(12.2)*
8	81.1 *(13.7)*	54.7 *(13.1)*	43.1 *(12.4)*
9	81.2 *(13.6)*	56.1 *(13.4)*	46.6 *(12.6)*
10	81.4 *(13.6)*	57.4 *(13.3)*	50.1 *(12.5)*

2.5.2 Relation between the methods

In figure 2.7 we see how the three methods relate to each other both for top-word recognition and top-five recognition. Neighbouring grey areas in the picture indicate that the methods have equal solutions. **Trigram** has the most in common with **hard** matching and also recognizes some words which **hard** did not recognize. **Fuzzy** has some solutions in common with both **hard** and **trigram**, some with only **hard** and some with only **trigram**. A few solutions are recognized by **fuzzy** uniquely. For the top-five recognition, we see that the shape of the figure is still the same, but that the areas do overlap each other more. The recognition percentages rise, so also the number of words which the methods recognize in common rises.

Notice that the surplus value of **trigram** and **fuzzy** over **hard** are small, but if they could be added up with the **hard** recognition, this would yield a considerable improvement.

For the exact results: see tables 2.3, 2.4 and 2.5. Table 2.3 shows the total number of recognized words out of 210 for each method, including standard deviation and recognition

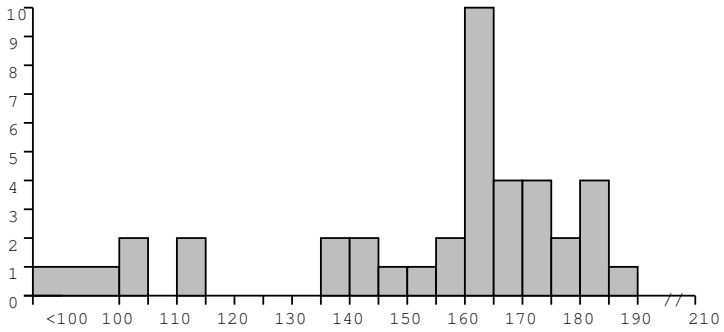


Figure 2.6: *Histogram of the results of the* **hard** *recognition method for 42 writers. Horizontal axis: number of recognized words out of 210, vertical axis: number of writers.*

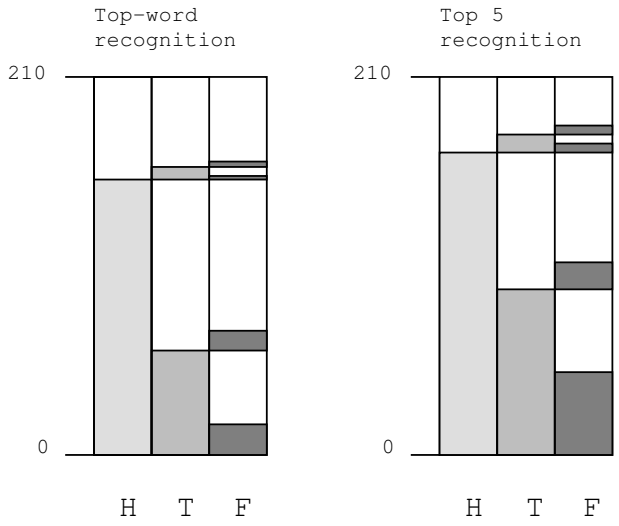


Figure 2.7: *Relation between* **hard**, **trigram** *and* **fuzzy** *matching. The overlap and the differences between the three methods can be seen. Horizontal axis: H=***hard**, *T=***trigram**, *F=***fuzzy**. *Vertical axis: lexicon of 210 words, average recognition results over 42 writers.*

percentage. Table 2.4 and 2.5 show successively the amount of overlap and the uniqueness of each method.

2.5.3 Results of the combination methods

The results from the two combining methods (rank-sort and rank-sort with weighting) are printed in table 2.6. At the results of the rank-sort combination (%), it can be seen that the top-five recognition is better than the top-five recognition of **hard** matching ($\chi^2 = 8.59$, $p < 0.01$, $df = 1$), but that the top-word recognition is much worse ($\chi^2 = 244.6$, $p < 0.01$, $df = 1$). This means that the results from **trigram** and **fuzzy** have a negative influence if used in this way. For the top-word recognition the results for the rank-sort with weighting method (*) are worse ($\chi^2 = 24.8$, $p < 0.01$, $df = 1$) than if only **hard** matching was used. For the top-five recognition the results are 4.4% better than the results from **hard** matching ($\chi^2 = 50.1, p < 0.01, df = 1$).

The results from the third combining method, stand-in, are printed in table 2.7. H??T

Table 2.3: *Recognition results for* **hard**, **trigram** *and* **fuzzy** *matching. The total number of recognized words out of 210 with the standard deviation and the recognition percentage are printed in the table both for top-word recognition and top-five recognition. H*=**hard**, *T*=**trigram**, *F*=**fuzzy** *matching.*

	H		*T*		*F*	
	# (s.d.)	%	# (s.d.)	%	# (s.d.)	%
Top-word	152.6 *(30.2)*	**72.7**	64.9 *(21.1)*	**30.9**	32.6 *(16.0)*	**15.5**
Top-five	168.1 *(29.2)*	**80.0**	102.5 *(26.6)*	**48.8**	71.1 *(24.6)*	**33.8**

Table 2.4: *Recognition results for* **hard**, **trigram** *and* **fuzzy** *matching. The number of overlap in the results from table 2.3 with the standard deviation and the recognition percentage are printed in the table both for top-word recognition and top-five recognition. H*=**hard**, *T*=**trigram**, *F*=**fuzzy** *matching.*

	$H \cap T \cap F$		$H \cap T$		$H \cap F$		$T \cap F$	
	# (s.d.)	%	# (s.d.)	%	# (s.d.)	%	# (s.d.)	%
Top-word	17.0 *(10.4)*	**8.1**	41.0 *(17.0)*	**19.5**	11.0 *(6.6)*	**5.2**	1.7 *(2.1)*	**0.8**
Top-five	45.9 *(20.1)*	**21.9**	46.5 *(18.7)*	**22.1**	15.5 *(7.2)*	**7.4**	4.8 *(4.4)*	**2.3**

is used to indicate that this method is used. H stands for **hard** matching and ??T stand for the use of **trigram** if **hard** has no solution. The top-word recognition percentage is now 74.5%, which are the best results for top-word recognition so far. For the statistical analysis, we can use a directional (one-tailed) test since the H??T recognizer can never be worse than the recognizer which uses only **hard** matching. It turns out that we have to reject the null hypothesis that the two recognizers are the same ($\chi^2 = 7.18$, $p < 0.01$, $df = 1$).

2.6 Experiment IIa: comparison of hard and ascordesc-trained matching

2.6.1 Rank-sort combination of hard matching and ascordesc-trained

For this experiment the **ascordesc**-trained method is used which first does **hard** matching on the reduced lexicon and then adds **fuzzy** recognized words to the tail of the word hypotheses

Table 2.5: *The number of words out of 210 which are uniquely recognized by one of the three methods with the standard deviation and the recognition percentage are printed in the table. H*=**hard**, *T*=**trigram**, *F*=**fuzzy** *matching.*

	H unique		*T unique*		*F unique*	
	# (s.d.)	%	# (s.d.)	%	# (s.d.)	%
Top-word	83.6 *(14.9)*	**39.8**	5.1 *(3.2)*	**2.4**	2.9 *(2.6)*	**1.4**
Top-five	60.2 *(16.6)*	**28.7**	5.3 *4.5)*	**2.5**	4.8 *(5.3)*	**2.3**

Table 2.6: *Results of combination techniques for* **hard**, **trigram** *and* **fuzzy** *matching. The number of words out of 210 which are recognized by these techniques with the standard deviations and the recognition percentages are printed in the table, both for top-word and top-five recognition. Also the overlap and unique scores of the two are printed. Abbreviations: %=rank-sort combination (no weighting), *=rank-sort with weighting, H=***hard**, *T=***trigram**, *F=***fuzzy**.

	H-T-F%		*H-T-F**		*H-T-F% ∩ H-T-F**		*H-T-F% unique*		*H-T-F* unique*	
	# (s.d.)	%	# (s.d.)	%	# (s.d.)	%	# (s.d.)	%	# (s.d.)	%
Top-word	123.9 *(30.2)*	**59.0**	146.4 *(27.2)*	**69.7**	120.5 *(30.4)*	**57.4**	3.4 *(2.3)*	**1.6**	25.9 *(9.6)*	**12.3**
Top-five	171.7 *(25.6)*	**81.8**	177.3 *(25.0)*	**84.4**	169.0 *(27.0)*	**80.5**	2.7 *2.7)*	**1.3**	8.3 *(4.7)*	**4.0**

Table 2.7: *Results from combining* **hard** *with* **trigram** *when* **hard** *does not have a solution. Abbreviations: H??=average number of words out of 210 and percentage where* **hard** *matching does not have a solution, ??T=number of words and percentage where* **trigram** *has a good solution if* **hard** *does not have a solution, H??T=use* **hard** *matching until there is no solution, then use* **trigram** *as stand-in.*

	H??		*??T*		*H??T*	
	# (s.d.)	%	# (s.d.)	%	# (s.d.)	%
Top-word	15.2 *(13.7)*	**7.2**	3.7 *(3.6)*	**1.8**	156.3	**74.4**

list. Only the **hard** and **ascordesc**-trained rank-sort combination (no weighting) was tested, because **ascordesc**-trained already makes use of **hard** matching, putting an extra weight would only give a method which looks more like **hard** matching. If more methods are involved, weighting becomes interesting, but not for two methods which already have that much in common. The top-word recognition of the combination method is significantly worse than the results from **hard** matching alone ($\chi^2 = 21.16$, $p < 0.01$, $df = 1$), whereas the top-five recognition is significantly better ($\chi^2 = 94.4$, $p < 0.01$, $df = 1$). The results from **hard**, **ascordesc**-trained and the combination are printed in table 2.8.

Table 2.8: *Results from* **hard**, **ascordesc**-*trained and the combination of the two using rank-sort combination (no weighting). The number of words which are recognized with the standard deviation and the recognition percentage are printed in the table, both for top-word and top-five recognition. Notation: H=***hard**, *At=***ascordesc**-*trained, % rank-sort combination (no weighting).*

	H		*At*		*H-At%*	
	# (s.d.)	%	# (s.d.)	%	# (s.d.)	%
Top-word	152.6 *(30.2)*	**72.7**	138.5 *(35.5)*	**66.0**	146.0 *(31.8)*	**69.5**
Top-five	168.1 *(29.2)*	**80.0**	154.4 *(35.2)*	**73.5**	179.7 *23.5)*	**85.6**

2.6.2 Ascordesc-trained as stand-in for hard matching

Just like **trigram** was used as stand-in in one of the former sections, now **ascordesc**-trained is used as stand-in when **hard** does not have a solution. Note that the part of **hard** matching within **ascordesc**-guessed is not relevant in the stand-in method since **hard** did not have a solution. The effects are due to the **fuzzy** matching method within **ascordesc**-trained.

The results from this stand-in method are printed in table 2.9. The top-word recognition percentage is now 74.4%, which are the best results for top-word recognition so far. For the statistical analysis, we can use a directional (one-tailed) test since the H??At recognizer can never be worse than the recognizer which uses only **hard** matching. It turns out that we have to reject the null hypotheses that the two recognizers have the same performance ($\chi^2 = 8.44$, $p < 0.01$, $df = 1$).

Table 2.9: *Results from combining* **hard** *with* **ascordesc**-*trained when* **hard** *does not have a solution. Abbreviations: H??=average number of words out of 210 and percentage where* **hard** *matching does not have a solution, ??At=number of words and percentage where* **ascordesc**-*trained has a good solution if* **hard** *does not have a solution, H??At=use* **hard** *matching until there is no solution, then use* **ascordesc**-*trained as stand-in.*

	H??		*??At*		*H??At*	
	# (s.d.)	%	# (s.d.)	%	# (s.d.)	%
Top-word	15.2 *(13.7)*	**7.2**	4.0 *(4.2)*	**1.9**	156.6 *(28.4)*	**74.4**

2.7 Experiment IIb: comparison of hard and ascordesc-guessed matching

As has been described in section 2.2, the difference between **ascordesc**-trained and **ascordesc**-guessed is in the use of different lexical subsets. **Ascordesc**-guessed makes use of a grammar. The grammar which has been used for these experiments is printed in table 2.1. From several pretests, this grammar turned out to give the best results, though this has not been investigated thoroughly. The recognition percentages of **ascordesc**-guessed and **ascordesc**-guessed as stand-in for **hard** matching are printed in table 2.10. Note again that the part of **hard** matching within **ascordesc**-trained is not relevant in the stand-in method since **hard** did not have a solution. The effects are due to the **fuzzy** matching method within **ascordesc**-guessed. The results from the H??Ag recognizer are significantly better than the results from the H recognizer, both for top-word ($\chi^2 = 15.2$, $p < 0.01$, $df = 1$) and top-five recognition ($\chi^2 = 20.6$, $p < 0.01$, $df = 1$).

The next experiment uses **ascordesc**-trained in stead of **ascordesc**-guessed in combination with **hard**, **trigram** and **fuzzy**, because **ascordesc**-trained in combination with **hard** had better results. The difference between **ascordesc**-guessed and **ascordesc**-trained is that **ascordesc**-guessed makes use of a grammar by which every word in the lexicon gets a contour-based pattern. **Ascordesc**-trained compares the generated countour-code with contour-codes of lexical subsets which contain the names of words which have been actually written by writers. The advantage of **ascordesc**-guessed is that works also for words which have not been written before.

Table 2.10: *Results from combining* **hard** *with* **ascordesc**-*guessed when* **hard** *does not have a solution. Abbreviations: H??Ag=use* **hard** *matching until there is no solution, then use* **ascordesc**-*guessed as stand-in.*

	H		*Ag*		*H??Ag*	
	# (s.d.)	%	# (s.d.)	%	# (s.d.)	%
Top-word	152.6 *(30.2)*	**72.7**	126.5 *(34.2)*	**60.3**	158.0 *(28.5)*	**75.2**
Top-five	168.1 *(29.2)*	**80.0**	151.3 *(36.2)*	**72.1**	175.9 *(25.0)*	**83.8**

2.8 Experiment III: combination of hard, trigram, fuzzy and ascordesc-trained matching

In this test **hard**, **trigram** and **fuzzy** are going to be combined with **ascordesc**-trained. The best combination of **hard**, **trigram** and **fuzzy** (H-T-F*) is combined with the combination of **hard** and **ascordesc**-trained (H-At%), without making use of weighting: ((H-T-F*)(H-At%))%. The results are printed in table 2.11. Again no weighting was done, for the same reasons as in the combination of **hard** and **ascordesc**-guessed.

The top-word recognition is not significantly different from **hard** recognition ($\chi^2 = 1.03$, $p > 0.01$, $df = 1$), but the top-five recognition is much better ($\chi^2 = 137.8$, $p < 0.01$, $df = 1$)

Table 2.11: *Results from rank-sorting* **hard**, **trigram** *and* **fuzzy** *rank-sort with* **hard** *and* **ascordesc** *rank-sort with weighting. H=*__hard__*, T=*__trigram__*, F=*__fuzzy__*, At=*__ascordesc__*-trained, %=rank-sort combination, *=rank-sort with weighting.*

	((H-T-F)(H-At%))%*	
	# (s.d.)	%
Top-word	154.0 *(27.5)*	**73.3**
Top-five	181.9 *(22.0)*	**86.6**

2.9 Summary of the results

In the previous paragraphs many results are put in many tables which may cause some confusion. In figure 2.8 all the experiments with the recognition percentages are put together in one picture. We see that the best top-word recognition percentage that is reached is 75.2% and the best top five recognition percentage is 86.6%. The best top-word results come from using **ascordesc**-guessed as stand-in when **hard** does not have a solution. The best top-five results come from combining **hard** and **ascordesc** rank-sort with **hard**, **trigram** and **fuzzy** rank-sort with weighting. The results from two of the stand-in methods do not have a top five recognition result, because this option was not available at the time of measuring. It is not possible to run the same tests again, since the recognizer is constantly being improved.

We realize that figure 2.8 is by far not complete. Different combinations can be used and different weights can be tested. Still these results show that there are interesting possibilities in this area.

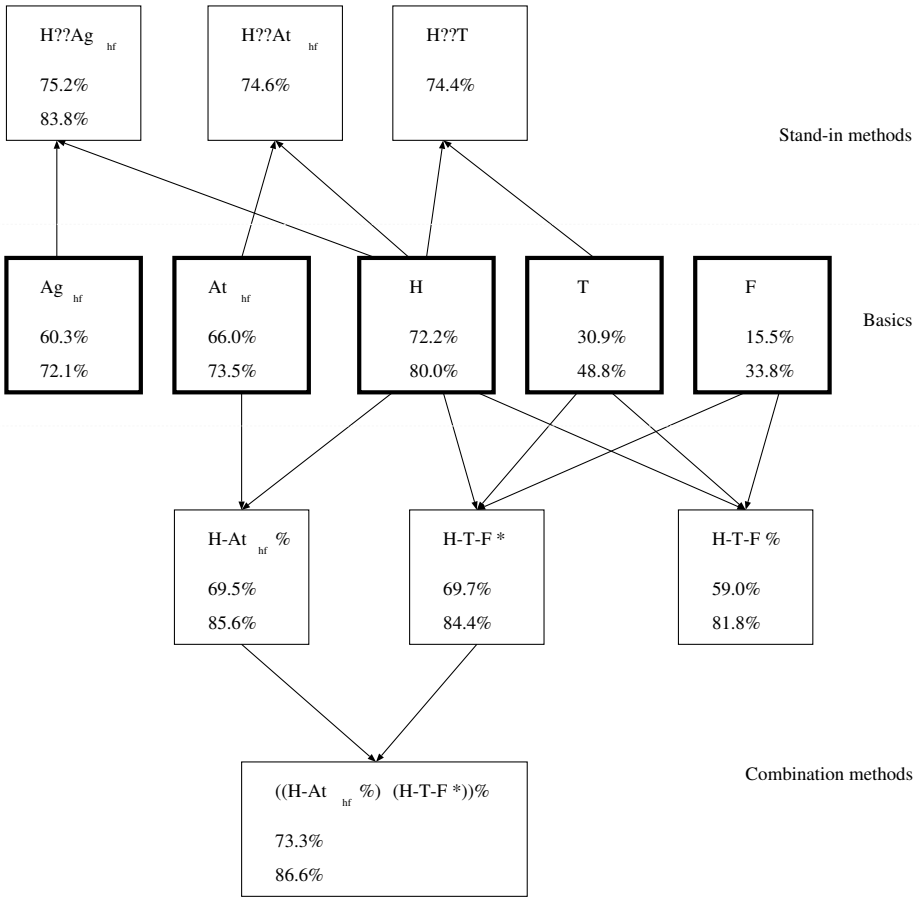


Figure 2.8: *Everything in one picture. The top-word recognition percentage is printed on top, the top-five recognition percentage below. Ag=***ascordesc***-guessed, At=***ascordesc***-trained, H=hard, T=***trigram***, F=***fuzzy***, hf=***hard*** *completed with* ***fuzzy***. *%=rank-sort ranking, *=rank-sort plus weighting, ??=stand-in.*

We could think of other ways to use the combination methods. For example, in the previous paragraphs the stand-in combination method has been used as a stand-in if **hard** matching did not have a solution. In two cases, **ascordesc** has been used as stand-in. One time **ascordesc**-guessed and one time **ascordesc**-trained. Both the ascordesc methods first used **hard** and then **fuzzy**. The fact that **hard** was used first is unnecessary since **hard** did not have a solution to begin with. A better method would have been to use an **ascordesc** method which uses **trigram** and if that method does not have enough solutions then again use an **ascordesc** method which uses **fuzzy**.

A final point which has to be made is that the methods which were combined were not extremely different from eachother. **Trigram** and **fuzzy** for example did not recognize many words which **hard** did not recognize. If there would have been a method available which would have been more different, the question can be raised if the combination techniques which we now have used would still be adequate. The rank-sort techniques may not give such good top-word results anymore, because if one method is not better than the other, but complementary, the techniques cannot decide which word is the correct word. The top-two recognition percentage of the rank-sort combination of two complementary methods would be 100% though. The stand-in method will always give equal or higher recognition percentages

than one method in isolation, since one method is used as a stand-in for the other.

2.10 Discussion

In several experiments it was found that the combination of different methods led to worse top-word recognition results and better top-five recognition results, when compared to the standard left-to-right machine reading method (**hard**). This can be explained as follows. The standard left-to-right machine reading methods has the best results of all tested methods in isolation. If such a method is combined with a less well performing method, the combination method is not 'smart' enough to always put the correct word on the top-position. So sometimes, the correct word from the left-to-right reading method is pushed away from the top-position, but is still in the top-five. This way, the top-word recognition can decrease, while the top-five increases. Because of the combination techniques, the top-five of combined methods was always better than the top-five of the left-to-right method in isolation.

The combination techniques made smart use of the fact that the left-to-right reading method's top-five recognition is not much higher than it's top-ten recognition results. In left-to-right reading methods, if one letter cannot be recognized, the whole word cannot be recognized. Parallel methods on the other hand can overcome such problems. As we found, their performance is much worse than the left-to-right reading method, but eventually they will find the right solution. A method which uses trigrams (**trigram**) has also problems if one letter cannot be recognized, because this one letter causes the loss of three trigrams which cannot be used for recognition. A method which looks at individual letter presence (**fuzzy**), does not have this problem. That is why the recognition percentages of such a method will eventually (if many possible solutions are considered) be higher than the recognition percentages of other methods. This phenomenon can partly be seen in figure 2.5.

The performance of contour-based recognition methods (**ascordesc**) was also worse than the performance of the left-to-right reading method in isolation. It turned out that a contour-based method which makes use of a training set (which contains words -which have been actually written- with their contour-codes) performs better than a contour-based method which makes use of a contour-grammar. The foreknowledge of the former (**ascordesc**-trained) is better than the foreknowledge of the latter (**ascordesc**-guessed). An advantage of using a grammar is that no training is necessary, which makes it a more universal method. A disadvantage is that a contour-grammar can not predict how a particular word will be written, while a training-set has foreknowledge about different ways to write a word, so it is able to make useful predictions.

2.11 Conclusion

The conclusion that can be drawn from the experiments is that combining different post-processing methods in a smart way leads to significantly better recognition results. Especially the results from the top-five recognition are much better. These are important findings to increase the usability of the recognizer. If a pop up window with five alternatives does include the right solution, even though it is not on the top position, the recognizer is still user friendly.

In the future we also want to do research on a post-processing method which focuses on the first letters of the word. If these are correctly recognized, the results will be more understandable for the user and cause less irritation, even if the correct word is not found.

The next chapter enters the second part of this thesis: research on human reading.

Chapter 3

Experiment on human reading

3.1 Background of the experiment

3.1.1 Reading handwriting is different from reading printed text.

The vast majority of research in the area of human reading is focused on printed text. Unfortunately there has not been much attention to the reading of handwritten material, even though several experiments show that there is a difference between the two.

Corcoran and Rouse [3] conclude from their experiments in which they present handwritten words, typed words, or a mixture, that the processes which occur in the perception of handwritten words may well be different from those underlying the recognition of printed letters, since in the mixed condition the subject first takes time to decide whether (s)he is reading a printed or a handwritten word. Grossly different handwritings may even require different 'sub-routines'. Ford and Banks [7] examined if the different subroutines are truly perceptual mechanisms of analysis or whether they reflect strategies of inspection that depend on memory retrieval and 'educated guesses'. They found the former to be true. Zuniga, Humphreys and Evett [40] did a follow-up experiment in which it was found that unique mental procedures exist for the normalization of handwriting after which processing proceeds in the same way for hand- and typewritten stimuli. Van Jaarsveld [15] criticized the results from Ford and Banks. From his research he concluded that also top-down information can influence the perceptual processes during reading of handwriting. He claims that the distinction between handwriting and print may only be one manifestation of more fundamental distinctions like legibility. Brassé [2] puts question-marks at the way the research on printed text is used to generalize a model used for both printed and handwritten text. He searches for features which do play a role in the processing of reading handwriting and not in the processing of reading printed text. His experiment showed that people make use of ascender and descender information in handwriting. It was found that the presence of ascenders and descenders may enhance human recognition of isolated words written in cursive script.

3.1.2 Determination of features

In the history of research on reading, there was always a large interest in the geometrical features that humans use in reading handwriting. Suen [36, 37] tried to determine important features by manipulating the stimulus material by masking parts of the text image with a pattern or by leaving out parts (rectangles) of the bitmap (see figure 3.1).

The disadvantage of masking methods is that such methods severely influence the percep-

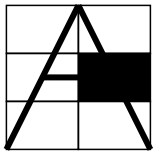


Figure 3.1: *Masking is done by making a rectangle of the bitmap black, so part of the letter 'A' can not be seen anymore.*

tual process. Since it is not known in advance where the features are located, the masks will usually only partly overlap the 'actual' features used by the human reader.

Masking is not a good method when cursive script is used, because all letters are connected so subjects cannot tell the different letters anymore when a part of the word is masked. Also, masking is a method in which the researcher works from his/her own hypotheses and does not let the subject determine which parts of the word can/should be masked.

We propose a method in which the subjects themselves can point out which parts of the word are important for recognition. With this method we hope to be able map the important letter features.

3.1.3 A new method

In our reading experiment, we present a method which allows *the reader* to indicate points of interest in the deciphering of handwriting patterns, without a pre-imposed manipulation of the handwriting image. The words that the subject has to read are barely visible, because the color of the ink will be only one shade of grey lighter than the color of the background. By clicking with the mouse, the subject can make the part of the words (s)he clicks on more visible, since the color of the ink becomes lighter. The subject has to try to read the word with as little clicks as possible.

We assume that the position of the clicks indicate points of interest. The amount of clicks on a certain position is a measure to determine the relevance of this position. Since subjects have to click as little as possible, it is safe to assume that they will not click on every letter of a word for reassurance. The clicks which are made are necessary for the recognition of the word.

3.2 Variables and expectations

In this section will be described why certain variables were chosen to be measured during the experiment. Also our expectations will be summed up and explained.

3.2.1 Variables

Since we assume that the position of the clicks indicate points of interest, the *XY-coordinates* of the clicks have to be recorded. The number of clicks (*Nclicks*) on a position are important to determine if this position is a relevant feature.

To be able to reconstruct the clicking-behaviour of each *subject*, the *time of clicking* was measured. With these measurement, we could even make a sort of film in which the word becomes more and more visible, just as the subject saw it as a result of his/her clicking.

The number of letters from every word (*Nletters*) are necessary information to determine if the letter position has influence on the number of clicks and if the length of the word is relevant.

The different *writing qualities* also have to be taken into consideration. We have seen in the previous chapter that some writers suppress the recognition rate of the computer reading considerably. The same can be true for the human reading.

3.2.2 Expectations

We have the following expectations:

1. The number of clicks (Nclicks) in the upper part of the word is higher than the number of clicks (Nclicks) in the lower part of the word. This expectation is based on the well-known fact that most of the information is in the top of the word. An example is given in figure 3.2.

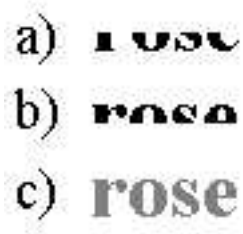


Figure 3.2: *If the lower half of the word is eliminated, the word 'rose' can still be read. If the upper part of the word is eliminated, there are already more possibilities at the first letter, which could be an 'f' or a 'p' or an 'r' etc.*

2. The number of clicks (Nclicks) on the first half of the word is higher than the number of clicks (Nclicks) in the second half of the word. This expectation is deduced from the logogen model which indicates that for reading, the beginning of the word is more important than the end of the word.

3. The average number of clicks on a letter of a long word is lower than the average number of clicks on a letter of a short word: Nclicks / Nletters decreases when the length of the word gets higher. The reason for this expectation is that a long word contains more information for the reader than a short word. Not all the letters have to be readable to be able to read a longer word. Missing letters can be easily filled in on the basis of context.

4. There is a large difference in the recognition of different handwritings. A sloppy handwriting probably needs more clicks than a neat handwriting. The computer also had a large difference in recognition percentages of different handwritings.

5. The fifth expectation is that if the average number of clicks on a letter increases, the word should be better recognized: when Nclicks / Nletters increases, Nrecognized should also increase. If a subject makes the word more clear by clicking on the trace, (s)he should also be better able to read the word.

6. The average number of clicks on a letter decreases if a word is more common: Nclicks / Nletters lessens when the word frequency gets higher. If a word is frequent in a language, it is more easily recognized, because it has a lower threshold, according to the logogen model. This means that the subject should need less information (clicks) to be able to recognize the word.

7. The final, and very important expectation is that subjects will especially click close to what Simon calls *singularities* [32]. We expect subjects to click on features which are important to distinguish one letter from the other, for example a crossing, a sharp stroke ending, a cusp etc. This expectation is based on the assumption that the clicking behaviour of the subject is focussed on reading the word as quick and with as little clicks as possible. The subject will not click on every part of every letter just for reassurance. The number of clicks on a certain position are a measure for the relevance of this position.

3.2.3 Purpose of the experiment

The purpose of the experiment is to get more information on which elements are important for letter- and word recognition by humans. With these materials a better and more 'human-like' recognizer may be build, whose solutions are better understandable for the user.

If for example, subjects do click more in the beginning of the word, then it would be interesting to build an extra routine in the handwriting recognizer, which focuses on the begin (i.e. left) part of the written word. If a recognizer correctly recognizes the begin part of the word, but fails in the end, this presumably is more acceptable for the user then if it recognizes only the end part correctly.

3.3 Method

3.3.1 Materials

From the data used in the previous chapters the handwritings of ten subjects were selected. We chose to only use cursive script since that is a better follow-up for Brassé's [2] research. Also, Suen [36] found that the reading of cursive *letters* produces more errors than the reading of printed letters. We wanted the task to be not too easy.

The word list consisted of 210 words (see Appendix A.1). It was divided randomly in ten sets of 21 words. Each set was assigned to one of the ten writers. It was checked that no misspelled or completely unreadable words were included in the dataset.

The length of the words was between 3 and 13 letters. In table 3.1 the distribution is shown.

In a pilot experiment it was found that it takes a subject about 45 minutes to read one-fifth of the word list. Since subjects voluntary joined in the experiment, we did not want to take more than one hour of their time. That is why it was decided to split up the word list in five different datasets. Each set contained an equally amount of the handwritings of all ten writers (the 21 words from every writer were randomly divided in five sets. One set contained six words, the other four sets contained five words.)

3.3.2 Subjects

35 subjects (10 female, 25 male), mainly students in Cognitive Science, voluntarily joined the experiment. They were not paid for their contribution. Their age was between 19 and 37 (av-

Table 3.1: *Distribution of lengths of the words in the dataset. Nlett=number of letters in a word, Nwords=number of words.*

Nlett	*Nwords*
3	3
4	11
5	28
6	30
7	47
8	36
9	36
10	13
11	3
12	2
13	1

erage 23.3). The subjects all had good vision during the experiment (10 wore contact-lenses, 5 glasses). The subjects were not familiar with the handwritings used in the experiment.

3.3.3 Procedure

The subjects were seated in a dimly lit room (equal conditions for every subject). First the instructions (see Appendix A.2) were read to them.

During the experiment each time a handwritten word appeared on the screen. The word was hardly visible, because the ink was only one shade of grey lighter than the background. By clicking on a location on the screen with the mouse pointer, the area was lighted up with a luminance curve which radially tapered off towards the grey background level. The effect is a tri-angular shaped function of luminance which only affects the handwritten trace, not the background. In table 3.2 the RGB-values and luminances of the gray colors are given, in figure 3.3 the luminance is plotted for a click.

Table 3.2: *RGB-values and luminance (in cd/m^2) of the gray colors used in the experiment. Gray50 is the color of the background, Gray51 the initial color of the ink, Gray53 .. Gray71 are the colors around the click (where Gray71 is the center of the click).*

Color-name	R	G	B	cd/m^2
Gray50	116	113	112	06.35
Gray51	120	116	115	06.85
Gray53	124	121	121	07.64
Gray56	133	131	129	09.22
Gray59	141	139	137	10.75
Gray62	149	148	146	12.69
Gray65	158	156	155	14.57
Gray68	165	164	163	16.59
Gray71	174	173	172	18.93

The subjects were told to try to correctly recognize each word using as little clicks as possible. There was a string stretched below eye-height at the border of the table the subjects seated on, to prevent them from going too close to the screen. The distance between string

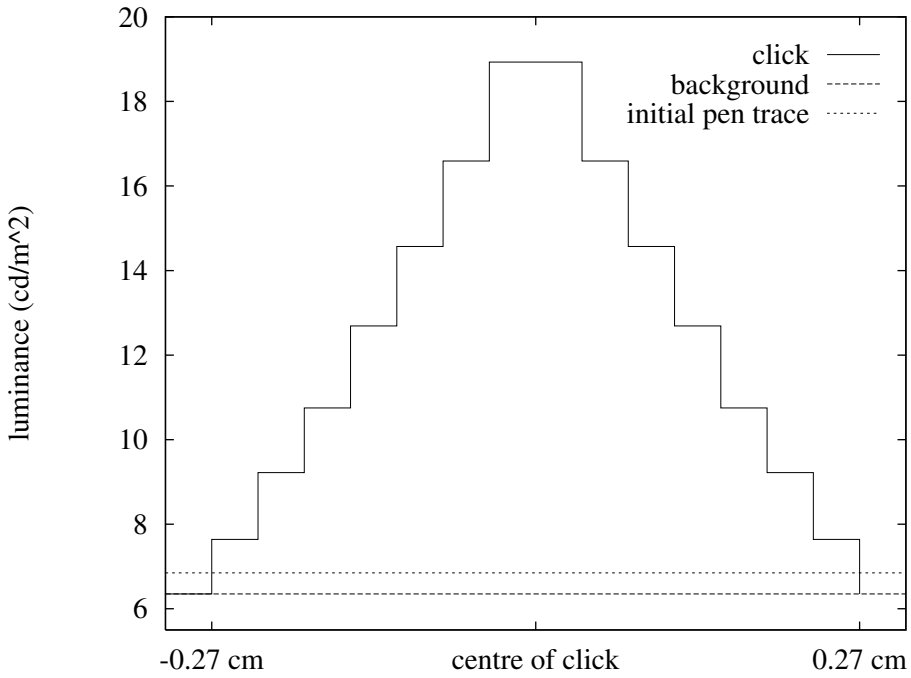


Figure 3.3: *The luminance in the center of the click is highest and gradually decreases to the initial color at the sides. One click has a view angle of about 0.6'.*

and screen was about 50 centimetres. The maximum width of a word on the screen was 16 centimetres. The corpus height was, dependent on the writer, between 0.5 and 1 centimetre. The words which appear on the screen are relatively large. For some handwritings, one click is enough to clear up a whole letter, for another handwriting this takes several clicks. This may influence the results.

It was stressed that the subjects directly when the word appeared on the screen had to click on the first thing they saw. This was primarily done to prevent that the subjects would start to gaze at the screen until they recognized the word.

The subjects had to operate under time pressure to prevent conscious reasoning. In the pilot it was found that six seconds per letter gave a reasonable pressure. If the pressure was too high (time too short), the subjects start clicking too quick, without carefully watching. The subjects saw how much time they had for each word. A time bar was running on the left of the screen.

There was one button on the screen The subjects could click on it to get a new word. If the word appeared on the screen, it was accompanied by a beep. If the subjects recognized the word, they could click on the button again, to stop the time from running and write down the word on the fill-in paper (see Appendix A.3) without any time-pressure. On this paper they also indicated if they were familiar with the word or not. A next click on the button caused the next word to appear on the screen.

After the instructions the subjects first trained with five words from a writer who was not in the real experiment. After this training session the real experiment began. At this point of time about ten minutes had gone by, so the subjects were adapted to the relative darkness of the room. The words were presented randomly to each subject. There were 35 subjects and five datasets, so each dataset was seen by seven subjects.

In figure 3.4 a picture of the screen (Hewlett Packard A2094A, Model no. GDM-1934,

Color Graphic Display) in the experimental environment is included.

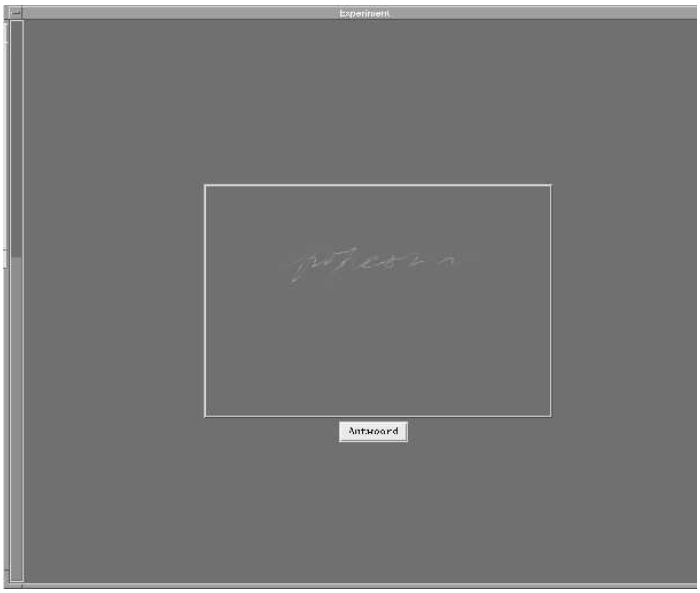


Figure 3.4: *Experimental environment. On the left of the screen is a time-bar running from bottom to top. In the middle is a window (width 16 centimetres, view angle 18') in which the word appears. Below this window is a button which either says 'antwoord' (answer) or 'volgend woord' (next word).*

Figure 3.5 shows a word which appears in the central window of the screen. The word which is written is *popcorn* and a few clicks have already been made.

Figure 3.5: *A word in the central wibndow of the experimental environment: popcorn. A few clicks have been made, to make the word more visible.*

After the experiment, the lights were turned on and the same words were randomly presented to the subject, this time in a clear condition. The subjects had to read the words aloud. This way we could see how well subjects are in reading cursive script. It has to be noted, however, that the subjects at this stage were familiar with the handwritings and already had seen the words before.

3.4 Results and discussion

3.4.1 Vertical click position

We expected the subjects to click more on the upper part of the word than on the lower part. The average vertical click-position turned out to be almost equal to the average vertical pen-trace position. So subjects did not click more on the upper part of the word. Figure 3.6 shows the average vertical position Y of the clicks on lower case letters, taking the handwriting baseline as zero. The values are sorted in order of increasing Y, yielding the given distribution of letters on the x-axis. As can be seen, the letters with descenders are on the left, the small, corpus-sized letters are in the middle, whereas the letters containing an ascender are on the right of the distribution. The <*f*>, which in most cursive writers has both a descender and an ascender stroke in cursive handwriting is located between corpus-sized letters.

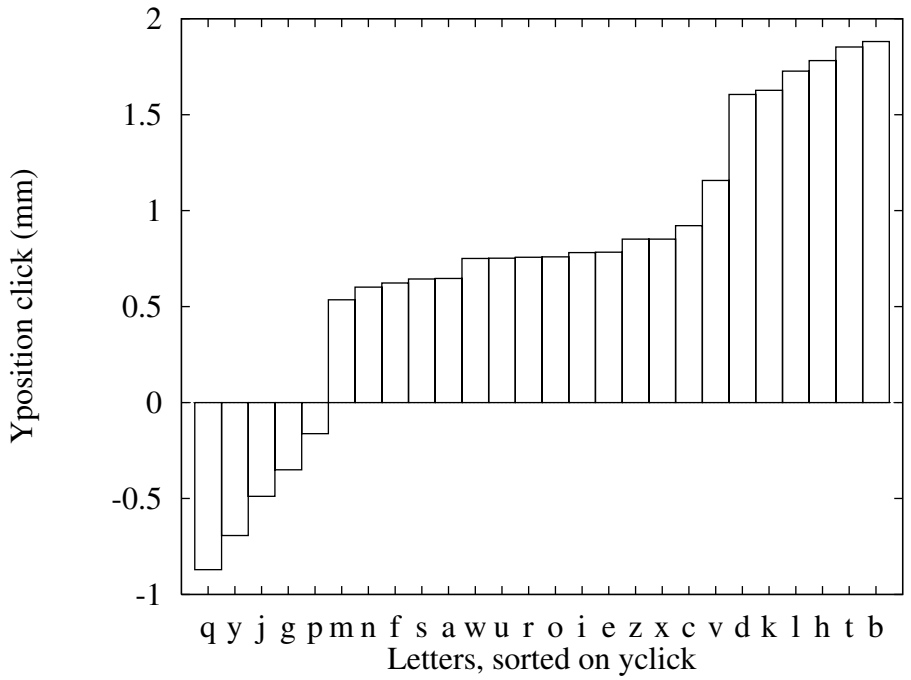


Figure 3.6: *Average vertical click position. Notice that the descenders are concentrated on the left, the ascenders on the right and the non-scenders in the middle.*

These findings are contrary to our expectations. The subjects follow the shapes of the letter instead of clicking only at the upper part of the words. Ascender and descender information thus seems important for the recognition of the word. It is difficult to tell whether the subjects use this shape-information for reassurance of a letter hypotheses they already had in mind, or if they actually need this information to decide which letter it is. Since the subjects were instructed to use as little clicks as possible, we believe that the ascender and descender information is important for recognition.

3.4.2 Clicks per letter

Figure 3.7 shows the distribution of average number of clicks on a letter for the lower case letters of the alphabet. Notice that he number of clicks on each letter have been normalized, which means that the number of clicks on a letter i have been divided by the number of times

this letter i appears in the word list. This normalization is necessary, because otherwise no comparison can be made between the clicks per letter.

This list in the figure is sorted in increasing number of clicks, resulting in the given order of letters on the x-axis. The average number of clicks on corpus-sized letters ($aceimnorsuvwxz$) is 10.6, whereas the average number of clicks on '-scender' letters $bdfghjklpqty$ is 12.8, which is statistically significant (t test, $p < 0.05$).

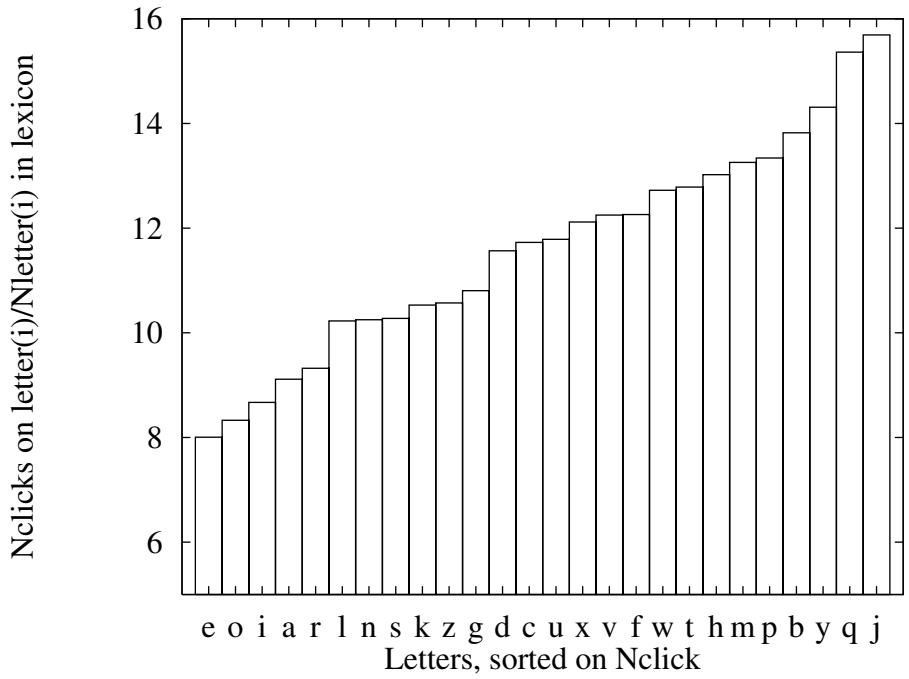


Figure 3.7: *Normalized number of clicks on each letter. Notice the concentration of vowels on the left.*

We can draw two conclusions from these results. The first one is a confirmation of the conclusion that shape information is important for recognition. Letters with ascenders or descenders are more clicked on than corpus-sized letters. The second conclusion is that vowels seem less important for recognition than consonants. The vowels concentrate on the left of the figure. There can be several reasons why vowels are letters on which subjects click less. One is that vowels contain few important features which are necessary for recognition. Another reason is that a word can often be read if the vowels are left out, but not if the consonants are left out. The vowels can be added based on letter context.

3.4.3 Click distribution in words

We also wanted to find out if the subjects click more on the initial, leftmost part than on the final, rightmost of the word. Figure 3.8 shows the distribution of clicks over the word pattern, from letter $-N/2$ to $N/2$, where different curves are shown for words of 4 up to 10 characters. Words with length 3, 11, 12 and 13 letters are left out this figure, since they appear less than 10 times in the dataset. Most 'clicks' are on the first and last letters of the word. There is no significant difference in the distribution of clicks on the first and second half of the word (number of clicks first half word: 8783, on second half word: 7414).

We can conclude that the first and last letter of a word are of great importance. It has to

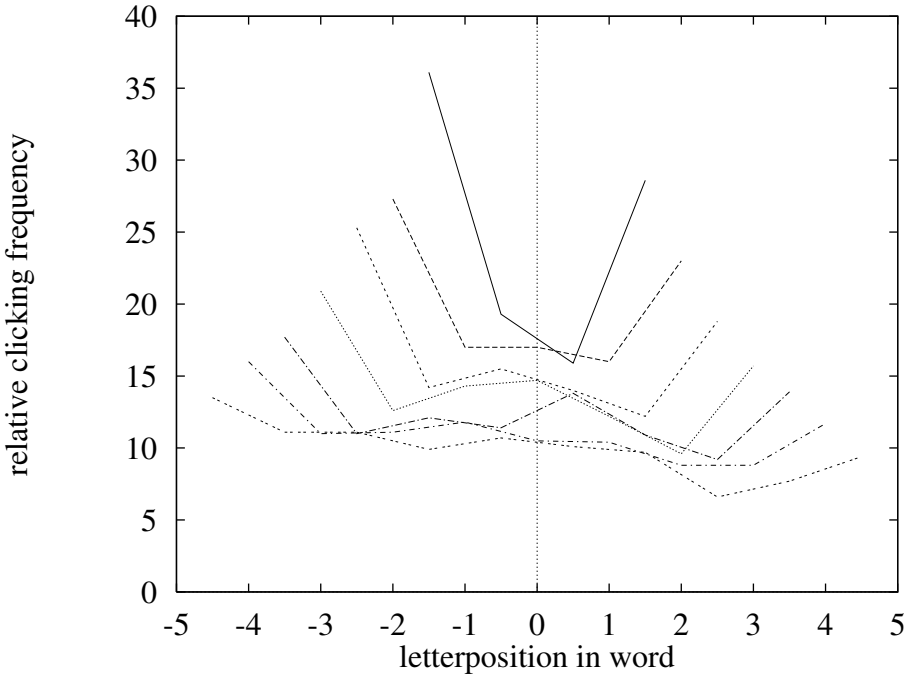


Figure 3.8: *Relative clicking frequency per letter position in words, for words of 4 to 10 letters.*

be checked now if the last letter was more clicked on because of 'clicking-strategies'. Maybe subjects only click on the first and last letter to determine the beginning and end of a word. This is why it was important to also have measured the time of clicking.

If we look at the distribution of clicks on the word in time, figure 3.9 shows clear evidence that the subjects click on the words from left to right. For the different word lengths (4 to 10 letters), the clicks on the letters were rank ordered in time. This way, an average rank order of the clicks per letter position was obtained.

With help of figure 3.9 we can conclude that the last letter does not receive so many clicks because the subjects first want to determine the beginning and ending of a word. The subjects may have this strategy, but it takes only one click on the last letter to determine the end of the word. On average, the clicks on the last letter are at the end of the session per word. This means that the subjects need the last letter to determine which word they are reading. This is a positive argument for the parallel reading model from McClelland and Rumelhart, and a negative argument for Morton's logogen model. If the logogen model was completely followed, the subject by the end of the word would already have decided which word they were reading. On the other hand, the fact that the subjects click on the word from left to right is a negative argument for the parallel reading model.

We also had the expectation that Nclicks/Nletters decreases when the length of the word gets higher. We determined for each word the average number of clicks on a letter. A regression-analysis was done on this data, but the results turned out not to be significant (R-Squared: 0.0667). There is no significant evidence that the length of the word has influence on the number of clicks on each letter. This can be explained by the fact that the subjects often did not have the word to be recognized in their mental lexicon. This makes it difficult to fill in missing letters if only some letters of the word have been recognized. The subjects had to click on almost every letter before they could read an unknown word.

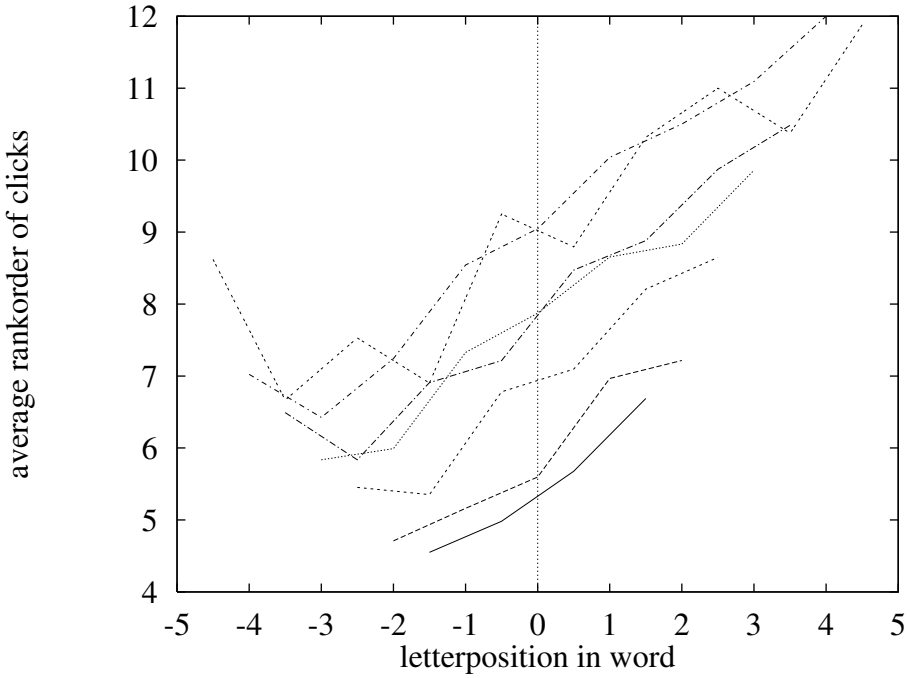


Figure 3.9: *The clicks on each word are rank ordered in time. We see that the subjects click on the words from left to right.*

3.4.4 Recognition percentages of human reading

Table 3.3 shows that the average recognition percentage of human reading is 62.3%. The *kind of writer* has influence on this percentage. It is clear that some handwritings are better recognized than others. If a handwriting has a high recognition percentage, the number of clicks is low. The lower the recognition percentage of a handwriting, the higher the total number of clicks on this handwriting.

Table 3.3: *Recognitions results for each writer. Each writer wrote 21 words. The dataset of 210 words was divided in 5 subsets. Every subset was read by seven different subjects.*

Writer	*Nclicks*	*% correct*
1	1184	85.7
2	1106	81.6
3	1382	76.9
4	1492	76.9
5	1757	62.0
6	1796	56.5
7	1849	55.1
8	1713	44.9
9	2042	44.9
10	1876	38.8
all	16197	62.3

These results are contrary to the expectation that if the number of clicks increases, the recognition percentage also increases. If we check the expectation (when *Nclicks/Nletters* gets

higher, Nrecognized should also get higher) with a regression analysis, this turns out to be not significant (R-Squared: 0.3295), there is no linear relationship. The explanation for this results is that a bad handwriting can often not even be read if the word is completely clear. Subjects keep clicking on a word they cannot read, this increases the number of clicks, but not the recognition percentage.

In clear conditions (no clicking necessary, since the pen-trace is distinctly legible), the subjects recognized 87.9% of the words. It should be noted here that this percentage is positively biased, since the subjects were now used to the handwritings and had seen the words before.

There is also no relationship between Nclicks/Nletters and word frequency (R-Squared: 0.0016) nor between the number of times a word is correctly recognized by the subjects (N=7) and the word frequency (R-Squared: 0.0123).

Word frequency was determined with help of the CELEX database:

> "All lexical data are stored in three separate databases, all of which are open to external users. The Dutch database, version N3.1, was released in March 1990 and contains information on 381,292 present-day Dutch word forms, corresponding to 124,136 lemmata. The latest release of the English database (E2.5), completed in June 1993, contains 52,446 lemmata representing 160,594 word forms. Apart from orthographic features, the CELEX database comprises representations of the phonological, morphological, syntactic and frequency properties of lemmata. For Dutch and English lemma homographs, frequencies have been disambiguated on the basis of the 42.4 m. Dutch INL and the 17.9 m. English Collins/COBUILD text corpora. Furthermore, information is being collected on syntactic and semantic sub-categorisations for Dutch. "

(This text is copied from : http://www.cis.upenn.edu/ ldc/readme_files/celex.readme.html)

The fact that word frequency does not influence the results is contrary to our expectations. In Morton's [20] logogen model, word frequency has the effect of reducing the criterial evidence necessary for word recognition. The fact that frequency does not have an effect in this experiment could be explained because many words have frequency zero.

3.5 Click-density in allographs

This section describes the next step in the experiment: finding out where on the letters the subjects clicked. Our seventh expectation was that they will click close to *singularities*. To find this out, we had to take all the letters of one type together, make an average letter out of it and project all the clicks on this letter type in that picture. Another word for a type of letter is an allograph. The <*a*> for example can be written as an '*a*', or as an 'a', these are both allographs of the letter 'a'.

The making of an average allograph (a prototype) is done as follows:

- Every letter in the dataset is labeled by hand. If there are three types of the letter 'a', these are labeled 'aI', 'aII' and 'aIII'.
- Each allograph is temporally re-sampled to 30 samples.
- The X and Y-coordinates of every sample per allograph are averaged over a number of instances of this allograph.

- A prototype allograph of 30 average samples can be plotted.

Then in every prototype the click-density's had to be projected. As has been described earlier, every click is a two-dimensional Gaussian function. If there are clicks on a prototype where these Gaussians overlapped, the values were added. This way the center of attention on every prototype is displayed. In the displaying of the figures, the allograph with the most clicks got more weight than an allograph with less clicks. This was done to get the clicks closest to the lines of the prototype. A disadvantage is that this sometimes can deform the prototype (for example xI in Appendix B.2). If there is a concentration of clicks on a certain position, this position in the prototype could be an important feature.

Simon & Baret [32] classify features in two broad classes: the *regular class* and the *singular class*. They define the singular class as made of features which are the complements of the features of the regular class: extremities, crossings, cusps, etc. A feature from the singular class is called *singularity*. This corresponds to the term structural feature. A structural feature is complementary to a metrical feature, which is a feature such as the size of a letter. Every letter has a size, but not every letter has to have a crossing, or a sharp edge, etc.

In figure 3.5 are five examples of the kind of results we obtained. The allographs *jI, lI* and *yI* all three have comparable loop shapes, yet we see in the figure that the subjects have indicated different points of interest. For *jI* and *lI*, the crossing seems most important, but for *yI* the sharp edge is more important (more clicked on) than the crossing. An explanation could be that for *yI* the subjects click especially on the sharp edge, because that is the point where they can see if they are reading one or two letters. If that is decided, and they know this sharp edge is connected to a descender, the only letter decision that can be made is the *y*. Allographs *aI* and *aIII* are both examples of clear features coming out. Allograph *aI* shows the importance of vertical strokes, *aIII* the importance of a flaw curve on the end of a stroke.

In Appendix B.2 the results of all allographs can be seen.

The results show that our method of determining features by letting the subject decide turned out well, since they confirm known phenomena, such as the importance of vertical strokes [24] (for example allograph *aI*) and crossings (for example allographs *jI* and *lI*). We also have found new features which subjects seem to use for recognition. A curve for example attracts attention (*dII, dIII, oII, rII, sI* etc.). The most important part of the curve seems to be where the writing speed has reached a minimum. Also a round ending of an end-stroke often gets many clicks (*aIII, kII, tIV*, etc.). Again, this is a point of low writing speed. Both these features indicate that maybe for humans, just as has been implemented in the handwriting recognizer, points of low writing speed are important.

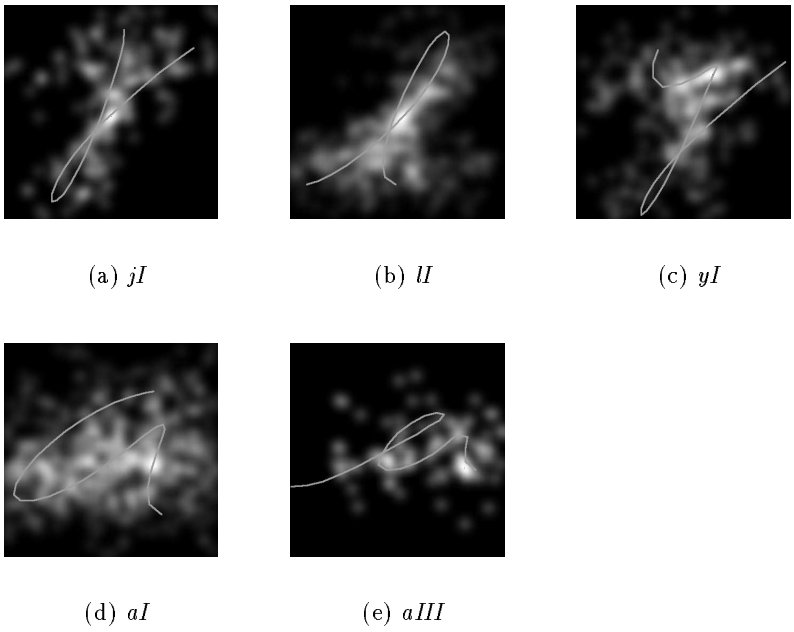
3.6 Critical remarks about the experiment

3.6.1 Allographs

Some readers may wonder if the results we showed on letters are the same as for allographs. In Appendix B.1 figures 3.6 and 3.7 are reprinted, this time for allographs. We see that the results are grossly the same.

3.6.2 Normalization

In the results the number of clicks each subject made were not normalized. We wanted to show these raw results, since even they show everything clearly. We did check if there are much differences if the results are normalized, but this hardly had influence. There can be a



(a) *jI* (b) *lI* (c) *yI*

(d) *aI* (e) *aIII*

Figure 3.10: *Click distribution in allographs. Even though the loop shapes of j, l and y are comparable, different features do approach. The two a's are good examples of clear features coming out.*

lot of discussion about the choice between presenting normalized or raw results. The former may give too much weight to the clicks of a subject which makes few clicks and overlook the importance of clicks of a subject which uses many clicks. The latter on the other hand might have the opposite effect.

3.6.3 First click

As has been described, the subjects were told to directly click on the first thing they saw when the word appeared on the screen. This first click is of course contaminates the data since it is not a click on a point the subjects *wanted* to see.

3.7 Conclusions

We have conducted an experiment on human reading in which the subjects themselves could point out which parts of the word they needed for recognition. The results of the experiment proved the value of this method, because already known phenomena have been confirmed. We found for example that vertical strokes are important for recognition. Powalka, Sherkat and Whitrow [24] did experiments which also indicated the importance of vertical strokes. In our experiment subjects often have a concentration of clicks on vertical strokes.

Another phenomenon that has been confirmed is the fact that subjects in reading handwritten script make use of the shape of ascender and descender information [2]. The average vertical click position turned out to be almost equal to the average vertical pen-trace position. Subjects follow the shape of the word in their clicking-behaviour. Presence of ascenders and/or descenders does not improve recognition results, but subjects use their information for

recognition. We saw that in average, letters with an ascender or descender receive more clicks than corpus-sized letters. For the corpus-sized letters we found that consonants receive more clicks than vowels. This is also a result which agrees with the well-known fact that vowels are not as important for recognition as consonants. In a language like Arabic, vowels are even left out of the written text.

The method also made it possible to determine new features which are used for recognizing handwritten words. The subjects showed a preference for obtuse or blunt endings in down-strokes. They appear to be more important than expected. In improving handwriting recognition, this feature should be taken into consideration. It was also found that both the first and the last letter of the word seem to contain needed information. We did not expect the last letter of the word to be that important. This result shows that a module in a handwriting recognizer which concentrates both on the first and last letter would help to make the recognition results of the computer more human-like.

We want to stress that with this experiment we do not pretend to mimic the reading-process. It is initially an experiment to serve technical goals: improving handwriting recognition by determining features which humans use in reading. We think we have succeeded in that. In the fundamental research on reading, this experiment could best be seen as a pilot experiment on reading handwriting.

The next chapter gives a comparison of the human and machine reading of the same word list. We will try to find an answer to the question why computer results often seem counter intuitive from the point of view of the human user.

Chapter 4

Human and machine reading compared

4.1 Introduction

We now have answered the first two research questions from the introductory chapter ('What combination of computer methods has the best results in reading handwritten words?' and 'What features do humans need/use to recognize a word?'). The third and last question was if the word-classification errors which the computer makes, are in any way related to the errors which humans make. To answer this question, we will used the combined handwriting recognition method which gave the best results in chapter 2 to process the same word list which humans read in the reading experiment which is described in chapter 3. These machine results will be compared to the human results.

4.2 Which combined handwriting recognition method gave the best results?

In chapter 2 we studied a script recognizer which has been developed at NICI (i.e. VHS V11.3) [29]. We tried to find a combination of post-processing methods with a high recognition performance. The method which gave a high top-word recognition and the highest top-five recognition was a method which combined the list of possible solutions of four post-processing methods (**hard** (H), **trigram** (T), **fuzzy** (F) and **ascordesc**-trained (At)) in the following way: a rank-order combination of a) **hard** and **ascordesc**-trained rank-order combined and b) **hard**, **trigram** and **fuzzy** combined with the rank-sort plus weighting method. The abbreviation for this combined recognizer which we used in chapter 2 was ((HA%)(HTF*))%. We found two combined methods with a higher top-word recognition, but since their top-five performance is not known, we chose for the above described method to use for comparison with human reading, because this method has the best results we found.

4.3 Questions about machine and human readfing

As has been described in chapter 3, the wordlist of 210 words was divided in five sublists. Every sublist was presented to seven different subjects.

We have several questions about the relation and difference between the computer and human performance in reading this wordlist.

1. The first question to be answered is: What are the recognition rates of machine and human reading? For machine reading, we need the top-word and top-five recognition rates. For the humans, we have the results from the experimental condition and the post-experimental condition. In the experimental condition, the words were hardly visible, and the subjects had to click with the mouse to make part of the word more legible. In the post-experimental condition, the same words as during the experiment were presented to the subjects again, this time clearly visible.

2. What is the relation between the errors from the computer and the humans? Do the same words cause problems in reading, both for the computer and the humans? To answer this question, the answers of the seven subjects who read a word were combined. We chose to use majority voting: if more than three of the seven subjects are able to read the word, we assume that this word is legible for humans. It was decided to compare the top-word recognition of the computer with the results from the experimental condition of the humans, and to compare the top-five results of the computer with the post-experimental results of the humans. We realize that comparing these results in this way is not completely correct, but it will give some insight in the differences between machine and human reading.

3. What are the differences between incorrect readings of a word by the computer or by humans? To answer this question, we measured the Wagner-Fischer string distances between incorrect readings (of computer and humans) and the correct words. The computation of the Wagner-Fischer string distance will be described in section 4.5. The number of incorrect readings from the computer, is different from the number of incorrect readings from the humans. To statistically analyze the results of the measurements, a non-parametrical test is necessary, the Mann-Whitney U test. The null-hypotheses of this test is that the population distributions from which the samples are drawn are identical.

A major difference between machine and human reading is that the computer is forced to come up with a real word from a finite word list, as opposed to the strategy which humans use. Humans can invent pseudo-words or leave out part of the correct word in their answer, if they can not read the actual word. To get a better understanding of the 'human recognizer' we also have to examine the incorrect readings of the humans. If they can not read a word, do they take another, real word, which resembles the correct word? Or do they invent a pseudo-word or letter string? We want to find out how many times an incorrect reading of humans is another existing word. To see if they use uncommon letter strings, we took all the trigrams from the incorrect human reading and compared these to trigrams from a Dutch and English dictionary. We chose to use trigrams, because the computer also has a trigram-method.

Section 4.4 answers the first two questions about the difference between machine and human results in reading the word list. Section 4.5 first gives a theoretical introduction about Wagner-Fischer string distances and then describes the results from answering the third question about differences between machine and human reading. In section 4.6, the human incorrect readings are further examined. The final sections give a discussion and conclusion of the results.

4.4 Results of human and machine reading of the word list

In Appendix C the results from the combined handwriting recognition method are put next to the human results. The computer has a top word recognition of 71.4% and a top five recognition of 88.1%. The human results were 62.3% in experimental condition and 87.9% in clear condition (see figure 4.1).

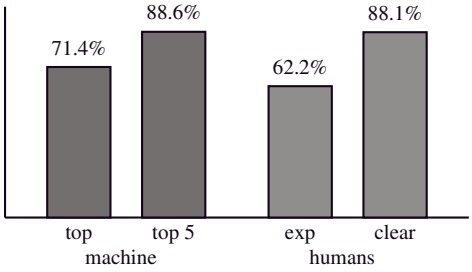


Figure 4.1: *Word recognition results (percentage correctly classified words) of machine and human reading of a word list of 210 words, cursively written by 10 different writers.*

The second question to be answered was if the same words cause problems in reading both for the computer and the humans. We see in the table in Appendix C, thet the machine results sometimes relate to the human results. A beautiful example is the word 'masker' which is being recognized as 'master' both by the computer and by the subjects. Table 4.1 shows the number of times the computer does not have the correct word in the top-position and in the top-five position and the number of times the majority of the humans can not read the word in the experimental and in the post-experimental (clear) condition. The third column of the table shows how many times the computer and the humans agree about not being able to read a word. From these results, we conclude that mostly, there is not much agreement between humans and computer about which words are legible and which are not.

Table 4.1: *Column 1: number of times that the computer does not have the correct word in top-position and top five-position, column 2: number of times that the majority of the humans do not have the correct word in the experimental (exp) and post-experimental (clear) condition, column 3: agreement between computer and humans.*

incorrect	*computer*	*humans*	*comp ∩ human*
top-word/exp	60	65	25
top-five/clear	24	14	4

The next section describes how a distance can be measured between incorrect readings and the actual words. The differences between human and computer answers are further examined.

4.5 Wagner-Fischer string distance between wrong outputs and actual words

4.5.1 Theoretical introduction and expectations

There are several methods to compute a value for the distance between two strings. One of these methods is called Wagner-Fischer [35], an algorithm which uses the Levenshtein distance metric. The Levenshtein distance metric gives the same weight to a substitution, a deletion and an insertion of a symbol:

$w(a,\epsilon) = 1 \quad \rightarrow$ *deletion*
$w(\epsilon,b) = 1 \quad \rightarrow$ *insertion*
$w(a,b) = 1$ if a $\neq$ b $\quad \rightarrow$ *substitution*
$\quad = 0$ otherwise $\rightarrow$ *match*

Table 4.2 gives an example of how the Wagner-Fisher algorithm computes the distance between the strings *zeitgeist* and *preterit*. The distance between the strings, i.e. $d_{8,9}$, is equal to 6. Calculation of $d_{i,j}$ is done as follows:

for $i = 1$ to m
  for $j = 1$ to n
    $d_{i,j} = \min\{d_{i-1,j} + w(x_i,\epsilon), \quad d_{i,j-1} + w(\epsilon,y_i), \quad d_{i-1,j-1} + w(x_i,y_i)\}$

Table 4.2: *Demonstration of the calculation procedure of the distance between 'preterit' and 'zeitgeist'. The optimal path is printed bold font.*

	j	0	1	2	3	4	5	6	7	8	9
i			z	e	i	t	g	e	i	s	t
0		**0**	1	2	3	4	5	6	7	8	9
1	p	1	**1**	2	3	4	5	6	7	8	9
2	r	2	2	**2**	3	4	5	6	7	8	9
3	e	3	3	**2**	3	4	5	5	6	7	8
4	t	4	4	3	**3**	**3**	4	5	6	7	7
5	e	5	5	3	4	4	**4**	**4**	5	6	7
6	r	6	6	4	5	5	5	5	**5**	6	7
7	i	7	7	5	5	6	6	6	**5**	6	7
8	t	8	8	6	6	5	6	7	6	**6**	**6**

We expect that if we compute the distance from the incorrect computer readings to the original list and the distance from the incorrect human readings to the original list, that the average computer distance will be larger. This expectation is deduced from the fact that the computer is forced to come up with a real word, in opposition to the humans.

4.5.2 Results of measuring Wagner-Fisher distances

From all incorrect computer readings, the distance was measured to what should have been the answers. The same was done for all incorrect human readings. If more than one subject could not read a word, the distances were averaged. The results are put in table 4.3.

Table 4.3: *Average Wagner-Fisher distances for all incorrect readings. N = number of times a word is incorrectly read by one or more subjects , $\bar{d}$ = average distance, sd = standard deviation.*

	N	$\bar{d}$	sd
Computer	60	5.5	2.5
Humans	169	2.9	1.4

To statistically analyze these results, a Mann-Whitney U test was performed. The results turned out to be significantly different ($z = 7.25$, $\alpha = 0.05$).

In figure 4.2, all the values of the Wagner-Fischer distances are put in one picture. The distances from the human wrong answers are the top half of the figure, the distances from the computer wrong answers are mirrored in the bottom half of the figure. What can be seen is that the human distances show a curve, which has a top at the distance between two and three. The curve then quickly drops to almost zero. The machine distances on the other hand have a top between four and five, but the curve does not drop to zero quickly. The difference between machine and human distances which was statistically determined, is easy to see in figure 4.2.

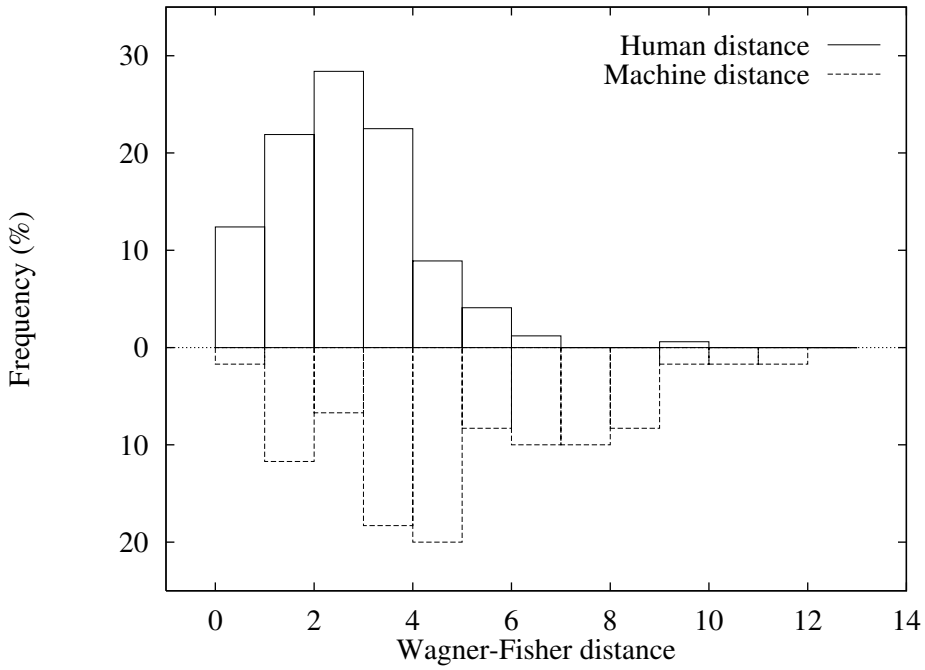


Figure 4.2: *Histograms of the Wagner-Fisher distances for incorrect human and computer readings. The top half of the figure shows the distances for the human answers, the bottom half the distances for the computer answers.*

4.6 Human results: non-words and impossible trigrams

We have examined the incorrect readings from the humans, to see if they often come up with a non-word, or produce, just like the computer, an existing word. The human answers were compared with a large list of Dutch words (2169872 words) and a list of English words (386117 words). It was found that a rather large percentage of the human wrong answers consists of words which are not in one of those dictionaries.

We also wanted to see if humans produce only trigrams which are 'possible' in a dictionary. In order to accomplish this, trigrams of every wrong answer were made. All these trigrams were compared with the trigrams from the two dictionaries. The percentage 'wrong' trigrams is 18.3% of the total number of trigrams in wrong answers. Note that the number of the wrong trigrams is influenced by the fact that humans often put a dot if they did not knew a letter. The results of these analyses are put in table 4.4.

Table 4.4: *Numbers and percentages of words and trigrams in the human wrong answers which are not in a Dutch or English dictionary. The total number of incorrect readings of humans is 555, the total number of trigrams that can be made out of these incorrect readings is 4019.*

Dictionary	*Not in dictionary*	
	words (555)	*trigrams (4019)*
Dutch (2169872 words)	439 (79.0%)	388 (09.7%)
English dict (386117 words)	444 (80.0%)	408 (10.2%)
Dutch & English	480 (86.5%)	735 (18.3%)

4.7 Discussion

In examining the differences between computer and human incorrect readings, it was found that there is not much agreement about which word are hard to read and which are not. This is an indication that the computer is still using different methods than the humans in reading. The computer is not acting 'human-like'.

To measure string distances, the Wagner-Fisher method has been used. This is a very pure method. The costs for substitution, insertion and deletion are equal. There are no extra costs for doing worse in the beginning than in the end of a word. The cost for a substitution is always the same, even though one substitution could be more severe than the other. Still, the method showed a clear difference between human and machine answers. The Wagner-Fisher distances for human answers are small and hardly ever are larger than 7. The distances for the computer answers are larger and more distributed between 0 and 12.

We also have examined the incorrect computer readings, to see if humans often come up with other existing words, non-words ore unpronounceable letter strings. We checked if the trigrams in the wrong human answers were existing trigrams or not. We found that almost 20 percent of the trigrams in wrong human answers were trigrams which did not exist in a large Dutch/English dictionary. This percentage is higher than expected. If we read the wrong answers, most non-words seemed pronounceable. The experiment showed that humans did not follow the conventions as much as we thought. Most of the wrong answers were non-words.

In only 13.5% of the wrong answers, humans produced an existing word, a behaviour which is more close to that of the computer.

4.8 Conclusions

The following conclusions can be drawn from examining the data from Appendix C:

1. A large amount of the incorrect readings of humans consists of non-words, while the computer always is forced to take a word out of its dictionary.
2. About 80% of the trigrams which humans produce in incorrect readings are trigrams which are also present in existing words, which indicates that many incorrect readings are pronounceable.
3. The average Wagner-Fisher distance from the incorrect readings to the actual words is larger for the machine than for the human answers. This is also due to the fact that the computer is forced to present an existing word.

Wrong computer results can be explained, but are often far from intuitive from the human point of view. A solution is not easy. What seems appropriate is to make an extra post-processing methods which looks at the beginning and end of the word (just like the subjects in the reading experiment did). If they are both correct, this reduces the dictionary enormously, and also gives more intuitive results.

The next and final chapter of this thesis gives a summary of the research described in this thesis by shortly answering the three research questions and by summing up the conclusions.

Chapter 5

Conclusion

5.1 Answering the three research questions

This section gives a summary of the research that has been done by answering each of the three research question that have been posed in the first chapter.

5.1.1 Question 1: What combination of machine reading methods has the best results?

Several combinations of four post-processing methods (**hard, trigram fuzzy** and **ascordesc** (trained or guessed) have been tried to find an optimal combination. We made use of three different combination methods: rank-sort combination, rank-sort with weighting and stand-in. The first method gives scores to the candidate words, based on their position in the list of possible solutions. The second method adds a weighting to these scores, candidates from a 'good' post-processing method get a higher weight than the candidates from a 'poor' post-processing method. The stand-in method uses post-processing method B if A does not have a solution.

It turned out that combining post-processing methods especially leads to better top-five recognitions. The results are almost 7% higher than in the standars left-to-right machine reading method. The top word recognition improves with about 3% which is a significant result.

The best top-five recognition result (86.6%) is found with the rank-sort combination of a) the rank-sort ranking of **hard** and **ascordesc-trained** and b) the rank-sort plus weighting of **hard**, **trigram** and **fuzzy**.

The best top word recognition result (75.2%) is found with **ascordesc**-guessed as stand-in for **hard**.

Combining different post-processing methods thus leads to significantly better results, especially for the top-five recognition. These findings can be used to try to improve the results of other handwriting recognizers, and maybe also for example speech recognizers. It might seem trivial that combining leads to better results, but the fact that it has this big of an impact was not expected. We haven't tried all possible combinations. From the results so far, no valid prediction can be made whether other combinations would come up with even better results.

If we would have had a post-processing method which would recognize many words which another does nor recognize, a good combination of the two would give very high scores. Unfortunately, the post-processing methods which were available did not have much surplus

value compared to each other. 76.4% was the highest possible top word recognition for **hard**, **trigram** and **fuzzy**. This result cannot be reached by only using combination methods, since a combination method does not know when to choose for the result from what method. It should be noted however, that all post-processing methods share the same information source: i.e. the stroke-based generation of letter hypotheses. Combining methods which have different origins probably leads to better results.

5.1.2 Question 2: Which features do humans use/need to recognize a word?

In an experiment handwritten words were presented to the subject in a hardly readable condition. By clicking with the mouse on a location on the screen, the subjects could make this part more clear. The subjects had to try to recognize the words as quick as possible with as less clicks as possible. It is presumed that the parts of the words were the clicks are, are most important for recognition. It was found that the first and last letter of the word are very important, that so called singularities [32] are important and that subjects use ascender and descender information to recognize the word.

The results can be used to implement a new post-processing method, which focuses on the beginning and end of a word. They can also be used to make other improvements to a handwriting recognizer if it has to become more human-like. With these results we now have a better idea of what is important for the human reader. It would be very interesting to try to find a confirmation for these results by making use of an eye-tracer in a similar experiment.

5.1.3 Question 3: Do the errors which the computer makes relate to the errors humans make?

The results of the human recognition of a word list of 210 words written cursively by ten different writers have been compared to the machine results of the best combination method. It turned out that the results of the top-five recognition of the computer are almost equal to the results of the humans when they read the words in clear condition (where no clicking was necessary because the pen trace was good visible). The words which the computer does not recognize are not the same as the words the humans do not recognize. This is a reason why the computer behaves not natural from the human point of view. Another major difference between the human and computer results is that the computer always has to come up with a word from its lexicon, while humans can think of non-words which are often closer to the actually written word than the word the computer had to make up.

Ways have to be found to make the computer act more human-like in order to become more accepted. Somehow humans expect a 100% recognition rate from the computer. This percentage is probably out of reach. Humans themselves are not able to recognize 100%. It would only be possible if humans would learn to write with 0% faults.

The best way probably to make handwriting recognition by machines more accepted by humans is to make sure that people have a good mental model of the system. This way, they understand why some errors are made and what they can do to make sure the computer does understand their handwriting. After all, we completely adapt to the computer if we use a keyboard, so why not adapt a little bit when using a handwriting recognizer?

5.2 Summary of conclusions

The general conclusions that can be drawn from the research done for this practical assignment are the following:

1. It was found that combining different post-processing methods leads to significantly better recognition percentages of handwritings the recognizer is trained on.
2. The top-five recognition even increases with almost 7% to a recognition rate which resembles normal human recognition of isolated words written by different writers in cursive script.
3. In an experiment on human reading it was found that humans do tend to focus on features such as down-strokes, obtuse or blunt endings in down-strokes, crossings etc.
4. In the same experiment another experiment has been confirmed in which was stated that readers make use of ascender and descender information in handwriting.
5. The experiment gave also evidence for the fact that the first and last letter of the word are important for recognition.
6. If the computer does not read a word correctly, the Wagner-Fisher distance between its answer and the actual word is generally larger than the distance between a wrong answer from a human and the actual answer.
7. A major difference between human and computer recognition of written words is that the computer always comes up with a word from its dictionary, while a human reader often gives up and invents a non-word. The computer has a finite list of answers, humans do not.

These conclusions could have several consequences. First of all, the results show that a pen interface with a pop up menu may be user friendly, since the top-five recognition rate is almost equal to the human recognition rate. Secondly, since humans seem to use letter-features in reading, this could help to make the machine reading better and also it confirms implementations which have already been made (like focussing on t-bar crossings for example). In the third place, a new post-processing method needs to be developed which focuses on the first and last letter of the word, in order for the recognizer to become more human-like. Finally, if the user has more insight in the way the recognizer works, (s)he can also have a better understanding of the mistakes that are made, so they are better accepted.

Appendix A

Experiment on human reading

A.1 Word list

abdomen	calcium	exuberant	larynx	showman
abstinent	charisma	fascist	lincoln	shuttle
adherent	checklist	feedback	lunchroom	sightseeing
adjunct	chevron	finland	luxe	sleep
advocate	chloride	fjord	macbeth	snob
afghanistan	cockpit	flipflop	magtape	society
album	cocktail	frankfurt	major	software
aldehyde	colonnade	fuchsia	masker	squaw
algebra	comfort	genre	maxwell	stanza
alluvium	concubine	gladiator	mazurka	stewards
alp	conjunct	god	megahertz	stockholm
amanuensis	copywriter	guyana	mysteries	stopwatch
analyst	cornwall	gymnast	native	strychnine
anecdote	corps	halfback	newton	studio
angst	cowboy	halve	nihilist	stuttgart
antecedent	crawl	hamster	object	sweatshirt
aorta	croquet	hoffman	ohm	symposium
appendix	cycle	hotdog	onyx	tableau
aqua	czerny	hulk	optimum	teamwork
arcsin	darwin	huxley	oxford	tokyo
auschwitz	dashboard	hyena	paperback	tomahawk
backup	deadline	hypotheses	papyrus	tonic
badminton	debugger	immigrant	partner	transfer
bangkok	dejeuner	inconvenient	persistent	trapezium
batik	delhi	inexact	pigment	trekking
bauhaus	delinquent	informant	pneumococcus	triplet
bazaar	deodorant	inhumane	poet	turf
bhagwan	diagnose	input	popcorn	turquoise
bijouterie	disjunct	interviews	portfolio	update
bladder	dixieland	israeli	potpourri	upgrade
bobby	dizzy	istanbul	potsdam	vacuum
bodyguard	dozen	jacques	projector	virgin
bolster	drink	jitter	prospectus	voltmeter
borax	edelweiss	jujube	quota	walrus
bouquet	entertainment	kafka	reflex	wonderland
boutique	equilibrium	kamchatka	rembrandt	workshop
bradford	equipment	keyboard	revue	wyoming
breakdown	essay	kidnapping	rhesus	xylophone
brisbane	excellent	kiwi	samovar	yoga
budget	exodus	knowhow	sandwich	yucca
buffet	export	kremlin	scherzo	zigzag
byte	extract	landcode	sheriffs	zwei

A.2 Instruction

INSTRUCTIE

Welkom bij dit experiment!

We willen bij dit experiment meer te weten komen over hoe mensen handgeschreven woorden lezen.

Tijdens het experiment verschijnt er telkens een handgeschreven woord in beeld (aan elkaar geschreven, dus niet in blokletters). Dit woord is echter nauwelijks leesbaar omdat het bijna dezelfde kleur heeft als de achtergrond. Door met de linkerknop van de muis te klikken kun je met elke klik een stukje van het woord helder maken. Als je genoeg helder hebt gemaakt om te weten welk woord er staat moet je dit opschrijven en kun je door naar het volgende woord. Het is de bedoeling dat je met zo weinig mogelijk klikken het woord herkent.

De woorden die op het scherm verschijnen zijn engels/nederlands, het zijn woorden die zowel iemand uit Engeland als iemand uit Nederland begrijpt. Er zitten ook eigennamen en plaatsnamen tussen, maar die zijn niet met een hoofdletter geschreven. De woorden zijn niet allemaal door dezelfde schrijver geschreven. Er zijn 10 verschillende handschriften. Misschien zitten er woorden bij die je niet kent, ook al zijn het allemaal bestaande woorden. Als dat zo is, is het moeilijker om zo'n woord te lezen. Als je dus hebt opgeschreven welk woord er staat, moet je ook aangeven of je het woord kent of niet. Je hoeft de betekenis niet te weten, maar het moet je wel bekend voorkomen.

Tenslotte is er nog één ding wat je moet doen. Telkens als er een woord op het scherm verschijnt, moet je eerst daar klikken waar je ogen naar toe getrokken worden. Je ziet het woord op het scherm verschijnen en direct gaan je ogen naar een bepaald punt van het woord. Dat moet je eerst aanklikken, daarna kun je je aandacht richten op het lezen van het woord.

Voor elk woord heb je een bepaalde maximum tijd tot je beschikking, afhankelijk van de lengte van het woord. Als deze tijd om is, verdwijnt het woord uit beeld en kun je opschrijven wat je herkend hebt. Als je al eerder weet wat er staat kun je op een knop drukken en het woord opschrijven. Dit opschrijven is niet aan tijd gebonden. Als je klaar bent met schrijven kun je met een druk op een knop het volgende woord in beeld krijgen. Als het nieuwe woord verschijnt hoor je een piepje.

Vóór het echte experiment begint doen we eerst een paar oefenwoorden. Het experiment zal ongeveer drie kwartier duren.

Als je nog vragen hebt kun je die nu stellen. Succes.

A.3 Fill-in paper

ANTWOORDVEL Naam: _____

De herkende woorden duidelijk (liefst in blokletters) opschrijven en aankruisen of je het woord wel of niet kent.

	Woord	Ken je het woord?		Woord	Ken je het woord?
	begin oefenen		6.	_____	O ja O nee
1.	_____	O ja O nee	7.	_____	O ja O nee
2.	_____	O ja O nee	8.	_____	O ja O nee
3.	_____	O ja O nee	9.	_____	O ja O nee
4.	_____	O ja O nee	10.	_____	O ja O nee
5.	_____	O ja O nee	11.	_____	O ja O nee
	eind oefenen		12.	_____	O ja O nee
	begin experiment		13.	_____	O ja O nee
1.	_____	O ja O nee	14.	_____	O ja O nee
2.	_____	O ja O nee	15.	_____	O ja O nee
3.	_____	O ja O nee	16.	_____	O ja O nee
4.	_____	O ja O nee	17.	_____	O ja O nee
5.	_____	O ja O nee	18.	_____	O ja O nee

	Woord	Ken je het woord?		Woord	Ken je het woord?
19.	_____	O ja O nee	32.	_____	O ja O nee
20.	_____	O ja O nee	33.	_____	O ja O nee
21.	_____	O ja O nee	34.	_____	O ja O nee
22.	_____	O ja O nee	35.	_____	O ja O nee
23.	_____	O ja O nee	36.	_____	O ja O nee
24.	_____	O ja O nee	37.	_____	O ja O nee
25.	_____	O ja O nee	38.	_____	O ja O nee
26.	_____	O ja O nee	39.	_____	O ja O nee
27.	_____	O ja O nee	40.	_____	O ja O nee
28.	_____	O ja O nee	41.	_____	O ja O nee
29.	_____	O ja O nee	42.	_____	O ja O nee
30.	_____	O ja O nee			
31.	_____	O ja O nee		eind experiment	

Bedankt voor het meedoen!!

Appendix B

Results experiment on human reading

B.1 Allographs

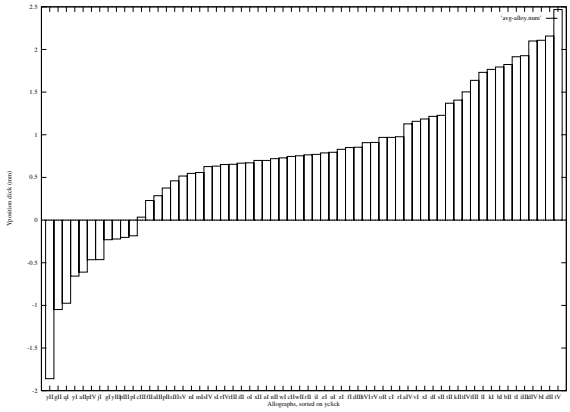


Figure B.1: *Figure 3.6 reprinted for allographs. Same results.*

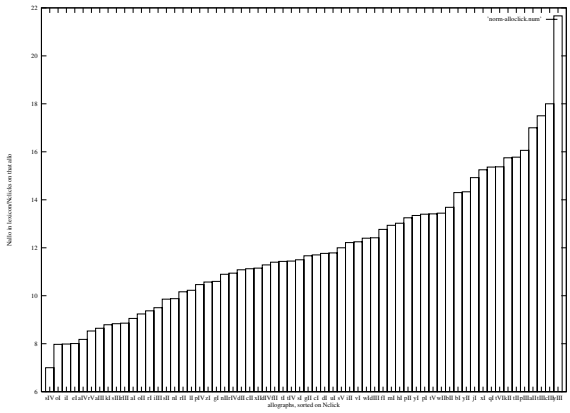
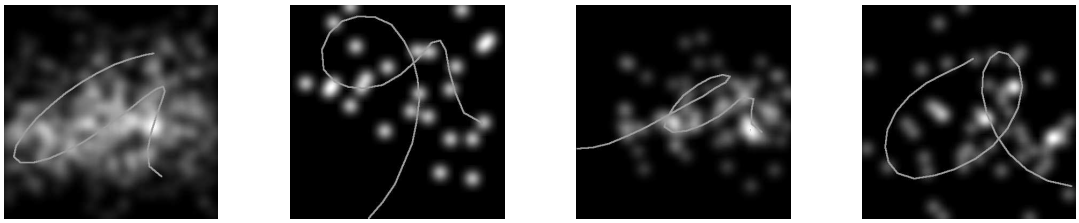


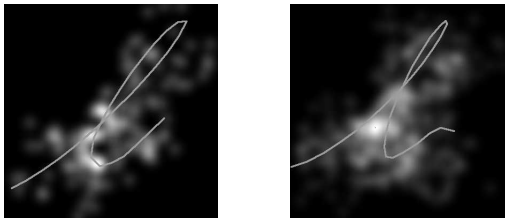
Figure B.2: *Figure 3.7 reprinted for allographs. Same results.*

B.2 Click-density in allographs



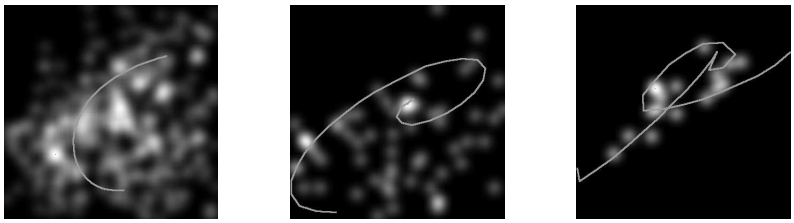
(a) *aI (112, 1276)* (b) *aII (3, 172)* (c) *aIII (14, 121)* (d) *aIV (11, 90)*

Figure B.3: Allographs *a*, with frequency and number of clicks.



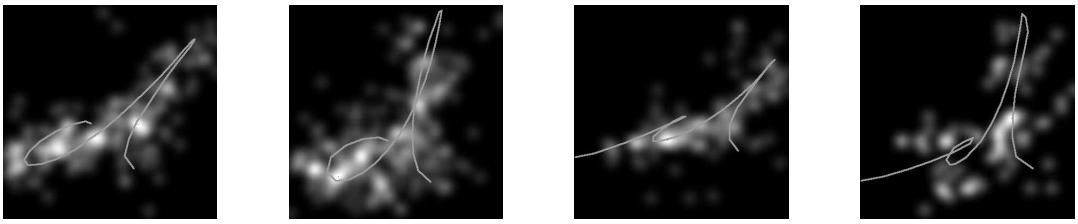
(a) *bI (10, 622)* (b) *bII (35, 479)*

Figure B.4: Allographs *b*, with frequency and number of clicks.



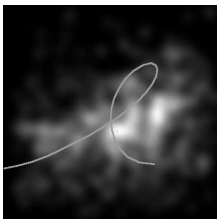
(a) *cI (57, 774)* (b) *cII (8, 107)* (c) *cIII (1, 18)*

Figure B.5: Allographs *c*, with frequency and number of clicks.



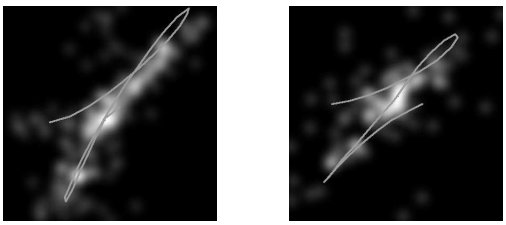
(a) *dI (7, 694)* (b) *dII (24, 415)* (c) *dIII (12, 149)* (d) *dIV (7, 79)*

Figure B.6: Allographs *d*, with frequency and number of clicks.



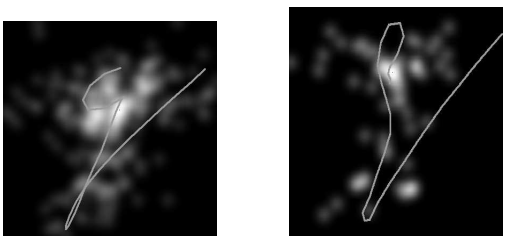
(a) *eI (143, 1145)*

Figure B.7: Allographs *e*, with frequency and number of clicks.



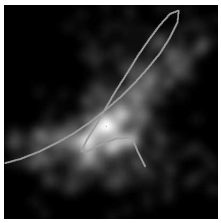
(a) *fI (17, 331)* (b) *fII (10, 114)*

Figure B.8: Allographs *f*, with frequency and number of clicks.



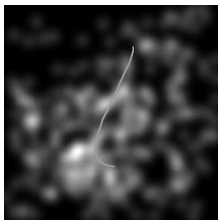
(a) *gI (25, 335)* (b) *gII (26, 70)*

Figure B.9: Allographs *g*, with frequency and number of clicks.

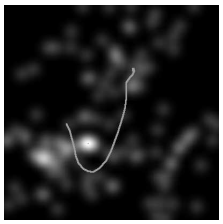


(a) *hI (45, 586)*

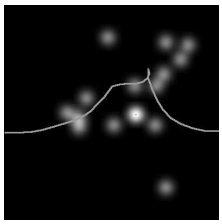
Figure B.10: Allographs *h*, with frequency and number of clicks.



(a) *iI (45, 789)*

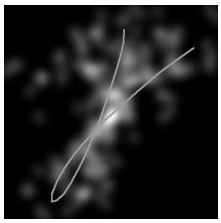


(b) *iII (14, 190)*



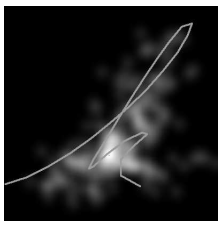
(c) *iIII (2, 19)*

Figure B.11: Allographs *i*, with frequency and number of clicks.

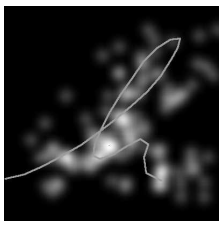


(a) *jI (13, 194)*

Figure B.12: Allographs *j*, with frequency and number of clicks.

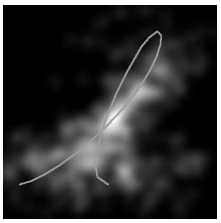


(a) *kI (24, 337)*



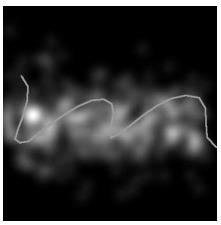
(b) *kII (8, 126)*

Figure B.13: Allographs *k*, with frequency and number of clicks.



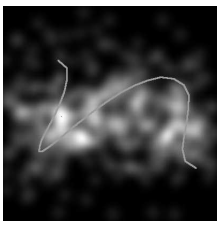
(a) *lI (57, 583)*

Figure B.14: Allographs *l*, with frequency and number of clicks.

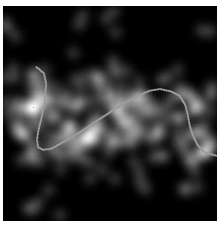


(a) *mI (45, 608)*

Figure B.15: Allographs *m*, with frequency and number of clicks.

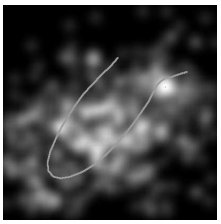


(a) *nI (66, 1066)*

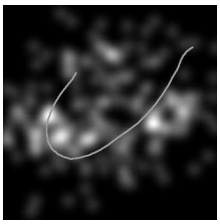


(b) *nII (38, 414)*

Figure B.16: Allographs *n*, with frequency and number of clicks.

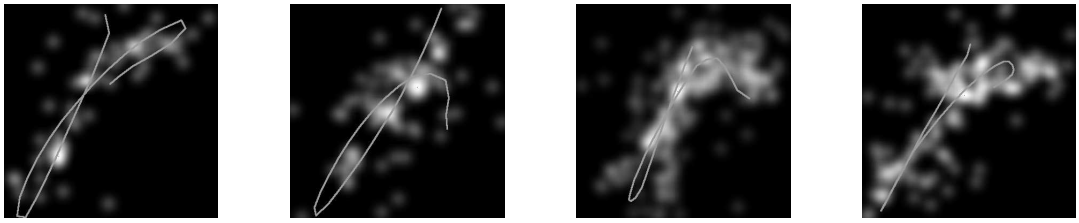


(a) *oI (74, 858)*



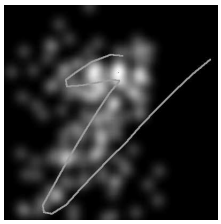
(b) *oII (29, 268)*

Figure B.17: Allographs *o*, with frequency and number of clicks.



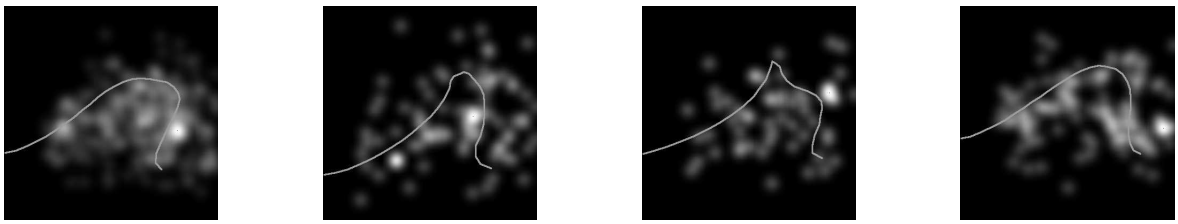
(a) *pI (5, 587)* (b) *pII (8, 363)* (c) *pIII (16, 257)* (d) *pIV (15, 157)*

Figure B.18: Allographs *p*, with frequency and number of clicks.

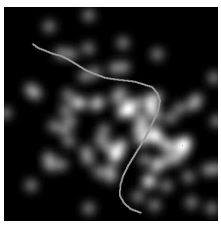


(a) *qI (11, 169)*

Figure B.19: Allographs *q*, with frequency and number of clicks.

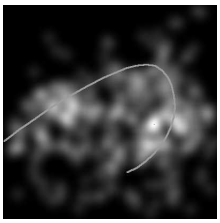


(a) *rI (43, 773)* (b) *rII (12, 184)* (c) *rIII (7, 62)* (d) *rIV (17, 186)*

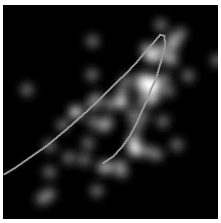


(e) *rV (7, 145)*

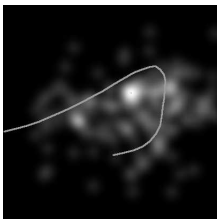
Figure B.20: Allographs *r*, with frequency and number of clicks.



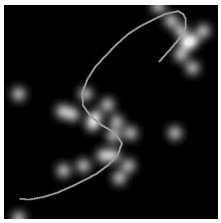
(a) *sI (40, 769)*



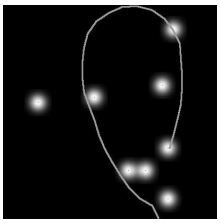
(b) *sII (7, 281)*



(c) *sIII (24, 212)*

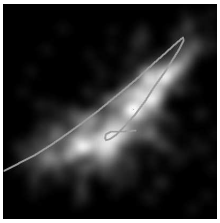


(d) *sIV (4, 28)*

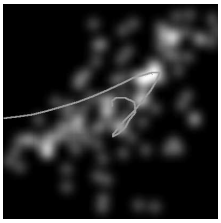


(e) *sV (1, 12)*

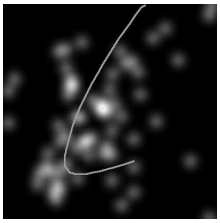
Figure B.21: Allographs *s*, with frequency and number of clicks.



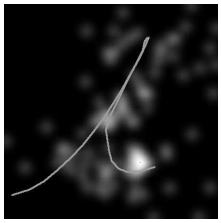
(a) *tI (65. 1058)*



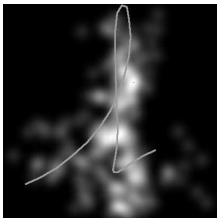
(b) *tII (9, 212)*



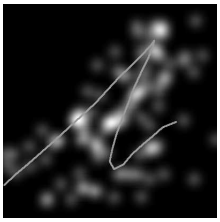
(c) *tIII (4, 70)*



(d) *tIV (9, 103)*

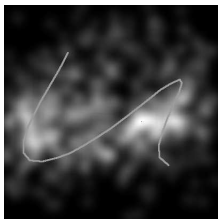


(e) *tV (17, 351)*



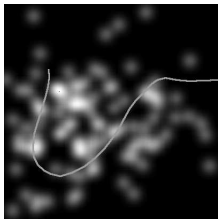
(f) *tVI (8, 123)*

Figure B.22: Allographs *t*, with frequency and number of clicks.



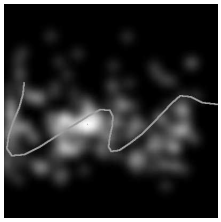
(a) *uI (70, 825)*

Figure B.23: Allographs *u*, with frequency and number of clicks.

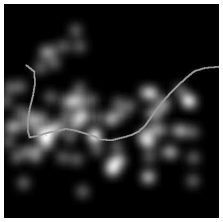


(a) *vI (12, 147)*

Figure B.24: Allographs *v*, with frequency and number of clicks.

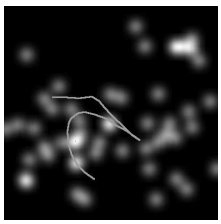


(a) *wI (20, 369)*

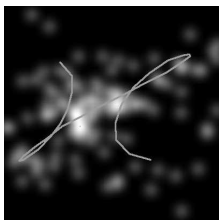


(b) *wII (9, 121)*

Figure B.25: Allographs *w*, with frequency and number of clicks.

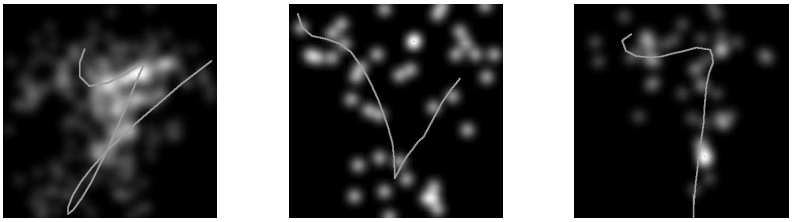


(a) *xI (4, 206)*



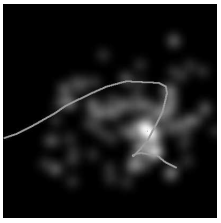
(b) *xII (13, 145)*

Figure B.26: Allographs *x*, with frequency and number of clicks.



(a) *yI (23, 415)* (b) *yII (3, 108)* (c) *yIII (3, 65)*

Figure B.27: Allographs *y*, with frequency and number of clicks.



(a) *zI (14, 148)*

Figure B.28: Allographs *z*, with frequency and number of clicks.

Appendix C

Machine and human answers

Table C.1: *Recognition of the wordlist (column 1) by the computer (column 2) and 7 humans (columns 3-9). For the computer results: if the correct word is not in the top position but is in the top 30, the position is put between braces behind the top word. An asterix (*) behind a human answer means that they could not read the word in clear condition.*

Word	comp	pp1	pp2	pp3	pp4	pp5	pp6	pp7
abdomen	OK	abahnen	abobome	abodrmen	ab...	balomen*	OK	abclonner
abstinent	interested	gestrinent*	gl.stinent*	glstinent*	OK	glo.tinent	glswnent	gl..ment*
adherent	sufficient(4)	aelcherent	adherkent	OK	OK	aelkerent	aelherent	OK
adjunct	OK	aalganet	adfanit*	adg.	adg..st	adgant	adjunit	OK
advocate	OK	advoeante	adwoadte	OK	advoeute	achvoerte	OK	OK
afghanistan	OK	hampton	OK	OK	OK	OK	OK	OK
album	OK	OK	alhum*	alumi	OK	alhum	OK	OK
aldehyde	OK	OK	OK	aldehijde	OK	OK	aldehijde	OK
algebra	OK	OK	OK	OK	OK	OK	l.lra	OK
alluvium	afternoon(15)	OK	OK	alluvi...	alluminium	alluvuum	alluminium	aluvium
alp	OK	OK	OK	OK	OK	OK	alpen	OK
amanuensis	OK	OK	aminnen*	OK	amantien*	amomeren*	OK	amanuensis
analyst	OK	OK	OK	analyse*	analyse	OK	OK	gynalise
anecdote	OK	OK	anekdote	OK	OK	anekdote	OK	OK
angst	OK	OK	argyd	OK	OK	OK	augit	angel
antecedent	OK	OK	arfeciden	fecedenf	feiedent	anfeilen	aufuedent	anfeadant
aorta	delete	aarfra	aorfre*	OK	aarfie*	aarta	aorfa*	OK
appendix	OK	OK	OK	OK	OK	append	OK	OK
aqua	OK	aguu*	OK	OK	OK	OK	OK	OK
arcsin	deden(2)	ausin*	ausin*	...*	ausin*	ausin*	cousin*	ausin*
auschwitz	OK	OK	verschuifd	OK	OK	ovenshuit	sch.uts	aller...
backup	OK	turkey*	lakuyn	OK	OK	OK	OK	bruckey
badminton	OK	badmentor	OK	OK	OK	OK	OK	OK
bangkok	strangers(14)	banakot	OK	OK	OK	OK	OK	OK
batik	takes(14)	OK*	batin	OK	balek	OK	OK	OK
bauhaus	OK	OK	OK	OK	OK	OK	OK	OK
bazaar	bronnen(3)	OK	OK	OK	OK	OK	OK	OK
bhagwan	OK	OK	rhagban	OK	OK	OK	baghwan	OK
bijouterie	OK	OK	bijjouterie	OK	OK	bijsuterie*	OK	byouterie
bladder	OK	OK	OK	OK	OK	OK	OK	OK
bobby	billy(14)	OK	OK	lobby	OK	OK	OK	OK
bodyguard	OK	OK	OK	bodygard	OK	OK	OK	OK
bolster	folder(4)	OK	OK	OK	bolsten*	bolsten*	OK	OK
borax	OK	OK	OK	OK	OK	OK	OK*	OK
bouquet	OK	bewust	bouget	OK	bougiet	bougeut*	briquet	bouguet
boutique	landscape	OK	coubrane*	houtzage*	houtbique	OK	boutrgne	hautique
bradford	OK	OK	OK	OK	OK	OK	braadforel*	OK
breakdown	OK	breardown	OK	OK	OK	OK	OK	OK
brisbane	OK	brisbreine	brislreme*	brisbene	OK	OK	OK	OK
budget	OK	OK	OK	buaget	budaget	budget	OK	buaget
buffet	OK	vaffel	OK	OK	beiffet	OK	OK	OK
byte	lips(4)	bye	OK	OK	bj...	OK	OK	OK

calcium	column(3)	ialuum*	OK	caluum*	OK	OK	caliu.m	caluum*
charisma	OK	chaam*	OK	karisma	char...	OK	OK	OK
checklist	deselecteert	OK	OK	OK	OK	OK	OK	OK
chevron	OK	OK	OK	OK	OK	OK	OK	OK
chloride	OK	OK	OK	OK	OK	OK	OK	OK
cockpit	OK	coeprit*	coeghiz	coehpit	OK	eoulque*	OK	OK
cocktail	OK	OK	OK	OK	OK	OK	OK	OK
colonnade	interested	colonne*	colonne	colo.nade*	coloro..de*	colomnade*	colorzam..made*	colorlande*
comfort	OK	OK	OK	OK	OK	OK	OK	OK
concubine	OK	conuchrine*	OK	jmachine*	emulsie*	coneedr..	unuilrine*	...m.rise*
conjunct	original	ongerust	conjennet	congruent	conj..ent	conjerent	contract*	OK
copywriter	OK	OK	OK	OK	OK	OK	copywrite	copywrite
cornwall	OK	comeback	comercient*	OK	comevalle	OK	contract*	comenall*
corps	OK	OK	oyeops	OK	OK	esps*	OK	OK
cowboy	country(2)	OK	OK	OK	OK	OK	OK	OK
crawl	OK	OK	vrawl	OK	OK	ovaal	c..aul	viawl*
croquet	OK	OK	OK	eroquet*	OK	OK	erognet	OK
cycle	apt(2)	cujle	agile	OK	cyile	OK	eyile	OK
czerny	among(2)	crusmg *	ezexmy*	czesny*	cz...*	czexmy*	caexmy*	czesny*
darwin	OK	dawm	OK	OK	OK	OK	OK	OK
dashboard	OK	OK	deskboard	OK	OK	OK	OK	OK
deadline	OK	OK	OK	OK	OK	OK	OK	OK
debugger	struggle	debug	OK	OK	OK	OK	OK	OK
dejeuner	OK	dyeumer	OK	OK	OK	OK	OK	OK
delhi	altar(11)	dek	OK	.elzi*	OK	delzin*	delzi*	getto
delinquent	OK	delenquent	telequent*	delenquent	delenquent	OK	OK	delequent
deodorant	OK	OK	OK	OK	OK	deodarant	OK	OK
diagnose	OK	OK	OK	OK	OK	OK	OK	OK
disjunct	OK	OK	disment	OK	dujunet	dujunet	OK	OK
dixieland	deselecteerden	OK	aixleland	OK	OK	OK	OK	dixiland
dizzy	diary(3)	OK	OK	OK	OK	OK	OK	OK
dozen	closer(2)	d.ren*	OK	lozen	OK	OK	OK	OK
drink	OK	OK	OK	OK	OK	OK	OK	OK
edelweiss	OK	edelweis	edelweis	OK	edelweis	OK	edelweis	OK
entertainment	OK	OK	OK	OK	OK	OK	OK	OK
equilibrium	OK	equolitrus*	OK	eq...m*	OK	equolab	equaliseren	egnoliolrum
equipment	OK	OK	OK	OK	OK	OK	OK	OK
essay	OK	OK	OK	OK	OK	OK	OK	OK
excellent	OK	OK	OK	OK	OK	OK	OK	OK
exodus	leaders(2)	OK	exvd	OK	OK	OK	OK	OK
export	OK	experiment	OK	expert	expert	OK	OK	expert
extract	OK	OK	OK*	OK	extact	extact*	extact*	OK
exuberant	OK	esenberant	OK	OK	OK	e.berant*	OK	OK
fascist	OK	OK	OK	fixu	OK	fesieyt	OK	frasisch
feedback	taaltech(2)	OK	OK	OK	OK	OK	OK	OK
finland	OK	finlaml	OK	OK	OK	OK	OK	OK
fjord	OK	OK	OK	afrit	OK	OK	OK	OK
flipflop	OK	OK	OK	OK	OK	OK	hiphop*	OK
frankfurt	OK	OK	OK	OK	frankrijk	OK	OK	OK
fuchsia	OK	OK	OK	fucksia	OK	OK	OK	OK
genre	OK	OK	OK	OK	gense	OK	geewze	genise
gladiator	OK	OK	gladeator	OK	OK	OK	OK	gladheid*
god	goal(2)	OK	OK	OK	OK	OK	OK	OK
guyana	OK	guinea*	gurguma*	gangzena*	gajena	gurpens	gurg.na	guyena
gymnast	opponent(7)	OK	OK	OK	gymenarc	OK	OK	OK
halfback	OK	OK	halbbrust	OK	OK	OK	OK	OK
halve	estate(2)	OK	OK	OK	OK	OK	OK	OK
hamster	OK	OK	hewster	OK	OK	OK	kosher	homster
hoffman	OK	OK	hofman	OK	hinlfman	hoofdman	OK	OK
hotdog	OK	brotlog	hobby	h...g*	hobby	notwg	bu.t.l.g*	intiogy
hulk	hullo(2)	OK	OK	hall	OK	OK	OK	OK
huxley	OK	huseley*	OK	huseley*	OK	OK	huseley	OK
hyena	OK	OK	OK	OK	OK	h..pent	OK	OK
hypotheses	OK	OK	hypfleres	hypothese	OK	hypothees	OK	hypothese
immigrant	countryside(3)	OK	kruispunt	OK	OK	ummipunt*	..pan.*	limimgrant
inconvenient	OK	mzonnevenne	convenient	momvenient	mo...ent	monument	meonvement	convenient
inexact	OK	inexait	OK	OK	OK	OK	inexait	OK
informant	OK	formant	fomem	OK	timmerman	moermansk	OK	morfismen
inhumane	OK	inhuman	inhimman	OK	inhssssmne*	inhumeren	OK	OK
input	triplet(2)	OK	impact	import	OK	OK	OK	OK
interviews	prospectus(4)	OK	OK	interview	interview	intenneus	OK	interview
israeli	OK	israel	israel	OK	OK	israel	israel	OK
istanbul	OK	OK	OK	OK	OK	OK	OK	OK
jacques	OK	OK	jacqi	OK	graagries*	geigers	jagues	jackies
jitter	pieter(2)	gitter	OK	j.tter*	OK	OK	jitten	OK
jujube	OK	OK	jiyad	OK	OK	jugule	OK	jungle
kafka	OK	OK	OK	OK	OK	OK	OK	OK
kamchatka	entertainment(22)	kamerhalter*	namazuthra*	hamerkatern*	kamerhen*	hamerhat..en	OK	kam.huti..*
keyboard	OK	iagboard	OK	OK	OK	hegboard	OK	OK
kidnapping	OK	gridmapping	OK	OK	OK	OK	OK	OK
kiwi	haar	ruui	kuur	OK	OK	OK	klur	OK
knowhow	OK	lenowho	OK	OK	OK	OK	OK	OK
kremlin	OK	OK	OK	OK	OK	OK	OK	OK
landcode	OK	londiode	OK	hardcode	OK	OK	loudcode	landiode
larynx	OK	OK	OK	OK	OK	OK	OK	OK
lincoln	OK	linium	OK	OK	lino...	OK	OK	finuolm
lunchroom	lieutenant(3)	OK	OK	OK	OK	lumkroon	OK	OK
luxe	lane(3)	OK	OK	binxe	OK	OK	OK	OK

macbeth	prullebak(5)	OK	OK	OK	maybeh	mubbett	OK	OK
magtape	OK	maatape	OK	OK	OK	OK	OK	OK
major	OK	OK	OK	OK	mayo	OK	OK	OK
masker	master(3)	masher	master	OK	master*	master	masher	OK
maxwell	OK	maewille*	matwell	maxwill	maewill*	OK	maewill*	maewell
mazurka	OK	mazarka	OK	OK	marinka	OK	OK	OK
megahertz	OK	OK	megaherz	OK	OK	OK	OK	OK
mysteries	OK	wepkries*	zwepsterries*	.*	wy...ies*	mepferies	webferies*	merferves*
native	nature(2)	nalive	nature	nature	nature	nature	makir	nalure
newton	OK	henden*	neutron*	hewbon	newborn	newborn	nenobon*	random
nihilist	OK	OK	OK	OK	OK	OK	OK	OK
object	cheque	OK	OK	OK	OK	objeel	OK	olyech
ohm	alone(2)	OK	OK	OK	OK	OK	OK	OK
onyx	cape(5)	algen*	OK	.	enzym	OK	onyp	mijn*
optimum	agreement	OK	zoimmen	optimun.	optimere	opklimmen	OK	optimism
oxford	OK	OK	OK	OK	OK	OK	OK	OK
paperback	OK	OK	OK	OK	OK	OK	paperhack	OK
papyrus	OK	papejeneis	OK	pope.rus	papiereus	paqvers*	paryrus	OK
partner	OK	OK	OK	OK	parts	arbeien	OK	barbner
persistent	OK	OK	OK	OK	OK	OK	OK	OK
pigment	OK	OK	OK	OK	OK	OK	OK	OK
pneumococcus	OK	pneumococeus*	pneumococeus	OK	OK	OK	pneumozoceus*	OK
poet	OK	OK	OK	poef*	OK	OK*	OK	poef*
popcorn	OK	poeorm	p.heorm	poptor..*	fopiorw	OK	popeom	fop...n
portfolio	OK	OK	rolgen	nortfalio*	porsifola	houtajo	OK	OK
potpourri	OK	potpouri	potpourie	potpouri	OK	OK	OK	OK
potsdam	feedback(4)	potselam	OK	OK	polselam	polselein	OK	potselam
projector	program	OK	project	OK	OK	proje.do.	properlor	OK
prospectus	OK	prospect	prospect*	peispectris	prospetus	OK	prwsteets	nevospechers
quota	goats(6)	OK	quote*	gnota	quote	OK	OK	OK
reflex	OK	pijlen	OK	OK	OK	OK	OK	OK
rembrandt	OK	OK	remelwondt	OK	OK	menilrandt	remlandt	OK
revue	severe(2)	OK	OK	OK	OK	OK	OK	OK
rhesus	OK	OK	OK	OK	OK	OK	OK	OK
samovar	OK	famouas*	tanraam*	fam...*	samurai*	tam.aar*	lambert*	tamoeran*
sandwich	OK	OK	jonvarch	OK	OK	OK	vandurch	OK
scherzo	OK	seherzo	sehezzo*	schezzo*	schezzo	seherso	schezzo	schezzo*
sheriffs	OK	sheriff	sherriffs	OK	sheriff	heriff	heriffs	sheriff
showman	OK	OK	howman	shoarma	OK	howman	hournan	OK
shuttle	OK	OK	ch.tl..*	OK	rha..ble	OK	chuttle	OK
sightseeing	OK	shidseemig	shightseeing	OK	corgheen	ghtseeing*	OK	OK
sleep	step(2)	OK	OK	OK	OK	OK	OK	OK
snob	trial	OK	OK	OK	OK	OK	tral	OK
society	OK	OK	violet	soc.et	soilet	OK	sodety	jodety
software	OK	OK	OK	OK	OK	softwane	OK	OK
squaw	eigenaar(14)	OK	sauaw	OK	OK	sajuart	OK	OK
stanza	OK	OK	OK	OK	OK	OK	OK	OK
stewards	OK	OK	steward	steward	steward	OK	OK	OK
stockholm	OK	OK	OK	OK	OK	OK	OK	OK
stopwatch	OK	OK	OK	OK	OK	OK	OK	OK
strychnine	OK	sochi...*	schrijvelzinne*	brychinue	bryelnine	brychnnne*	so.g...*	sorrychine*
studio	OK	stredino*	stroding*	stud.o*	stuctly*	studies*	strud.r*	struden*
stuttgart	OK	hitogant*	OK	OK	shittaart	OK	strittgart*	shittgart
sweatshirt	investment(2)	OK	iserthird	swaatshirt	OK	snoeklicht	OK	OK
symposium	OK	OK	OK	OK	syngoriun	OK	OK	OK
tableau	OK	OK	OK	OK	OK	h.ean*	OK	OK
teamwork	OK	OK	teamaro...	OK	OK	OK	OK	OK
tokyo	OK	OK	OK	tokuv	OK	OK	OK	OK
tomahawk	immediate(2)	tomehawk	tomathan	OK	vindhaark	gom.hout*	commauvel	tommahawk
tonic	trade(3)	trontz*	annie*	...*	trunk	annie*	a..r*	browse*
transfer	OK	inarfen	OK	OK	eramfer	OK	OK	OK
trapezium	OK	hageerium*	haperium	..um*	haje...num	harpezium	hapermim	hapersum
trekking	OK	OK	OK	OK	OK	strekking	OK	OK
triplet	OK	trifilet	OK	OK	OK	OK	OK	arikel
turf	fort	OK	f.rm	finish	OK	surf*	OK	OK
turquoise	OK	OK	OK	OK	OK	OK	OK	OK
update	OK	rijnstate	zydate	ripdate*	OK	OK	OK	ripdate
upgrade	OK	ifarde	OK	OK	OK	OK	OK	OK
vacuum	OK	OK	OK	bruum	vluurks	van...	OK	van.ism
virgin	weapon(15)	OK	OK	OK	OK	OK	vo.vgin	OK
voltmeter	OK	OK	OK	OK	OK	OK	OK	OK
walrus	OK	OK	walvis*	walvis*	OK	zalaas	OK	OK
wonderland	OK	OK	OK	OK	roonderland	OK	OK	OK
workshop	OK	OK	OK	OK	OK	nordshop	wordship*	OK
wyoming	OK	OK	wiommig	wjommig	OK	OK	OK	OK
xylophone	furthermore(5)	OK	tegelshow*	teglophene	ty.phone*	telephone	telephone*	....phone*
yoga	OK	OK	OK	OK	OK	OK	OK	OK
yucca	green(2)	yuour*	yucea*	yucra*	queen*	yuerre*	yucrw*	anges*
zigzag	OK	OK	OK	OK	OK	OK	OK	OK
zwei	inner	zone	ziner*	zinei*	zin..ei*	zinei*	zinei*	.*

Bibliography

[1] Braekel, van M. & Daenens, P. (1984). The infrared determination of writing sequence of ball-point pen strokes. *10th Meeting for the International Association of Forensic Sciences, volume 24/4* (450).

[2] Brassé, B. (1991). *De invloed van de globale woordcontour tijdens visuele woordherkenning: een vergelijking tussen drukschrift en handschrift. (The influence of global word contour during visual recognition: a comparison between machine print and script.)* Report 91-NICI-04, Nijmegen University, The Netherlands.

[3] Corcoran, D.W.J. & Rouse, R.O. (1970). An aspect of perceptual organization involved in reading typed and handwritten words. *Quarterly Journal of Experimental Psychology, 22* (526-530).

[4] Dengel, A., Pleyer, A. & Hoch, R. (1992). *Fragmentary string matching by selective access to hybrid tries.* ICPR (149-153).

[5] Eikvil, L. (1993). *OCR, Optical Character Recognition.* Rport No. 876, BILD, Norwegian Computer Center (http://www.nr.no/home/BILD/DocOnline.html).

[6] Fleming, K.K. (1993). Phonologically mediated priming in spoken and printed word recognition. *Journal of experimental psychology: learning, memory and cognition, 19* (272-284).

[7] Ford, B. & Banks, W.P. (1977). Perceptual differences between reading handwritten and typed words. *Memory and cognition, 5 (6)*, (630-635).

[8] Frankish, C., Hull, R. & Morgan, P. (1995) Recognition accuracy and user acceptance of pen interfaces. *Conference on Human Factors in Computing Systems: Mosaic of Creativity. CHI'95 Proceedings.*

[9] Guyon, I. & Pereira, F. (1995). Design of a linguistic postprocessor using variable memory length Markov models. *Proceedings of the ICDAR '95*, Montreal (454-457).

[10] Haber, L.R., Haber, R.N. & Furlin, K.R. (1983) Word length and word shape as sources of information in reading. *Reading Research Quarterly, 18* (165-189).

[11] Helsper, E. & Schomaker, L.R.B. (1993). Off-line and on-line handwriting recognition: the role of pen movement in machine reading. Optical Character Recognition in the Historical Discipline: *Proceedings of an International Workshop organized by NDHA and NICI.* Goettingen: Max-Planck Institut fuer Geschichte (39-51).

[12] Huang Y.S. & Suen C.Y. (1993). An optimal method of combining multiple classifiers for unconstrained handwritten numeral recognition. *Pre-Proceedings of the IWFHR-III*, CEDAR, Buffalo, USA.

[13] Hull, J.J., Commike, A. & Ho, T.K. (1990) Multiple algorithms for handwriting recognition. In C.H. Suen (Eds.), *International Workshop on frontiers in handwriting recognition*, Montreal: CENPARMI.

[14] Higgins, C.A. & Bramall, P.E. (1994). *Cursive script recognition based on a combination of AI techniques.* ISSN 0963-3308. Colloquium on 'Handwriting an pen-based input.' IEE, Savoy Place, London, UK, pp. 7.1-7.7.

[15] Jaarsveld van, H.J. (1983). *On reading handwriting.* Ph.D. thesis, Nijmegen University, The Netherlands.

[16] Kohonen, T. (1987) Adaptive, associative, and self-organizing functions in neural computing. *Applied Optics, 26* (4910-4917).

[17] McClelland, J.L. & Rumelhart D.E. (1981). An interactive activation model of context effects in letter perception: Part 1. An account of basic findings. *Psychological review, 88* (375-405).

[18] Meyer, A. (1995). Pen Computing: a technology overview and a vision. *ACM SIGCHI Bulletin.*

[19] Monk, A.F. & Hulme, C. (1983). Errors in proofreading: Evidence for the use of word shape in word recognition. *Memory & Cognition, 11* (16-23).

[20] Morton, J. (1969). Interaction of information in word recognition. *Psychological review, 76* (165-178).

[21] Neisser, U. (1967) *Cognitive Psychology.* New York.

[22] Paap, K.R., Newsome, S.L., McDonald, J.E. & Schaneveldt, R.W. (1982). An activation-verification model for letter and word recognition: the word-superiority effect. *Psychological review, 89* (573-594).

[23] Paap, K.R., Newsome, S.L. & Noel, R.W. (1984). Wordshape's in poor shape for the race to the lexicon. *Journal of Experimental Psychology: Human perception and performance, 10* (413-428).

[24] Powalka, R.K., Sherkat, N. & Whitrow, R.J. (1995). Recognizer characterisation for combining handwriting recognition results at word level. *Third Int. Conf. on Document Analysis and Recognition. ICDAR '95*, Montreal, Canada (68-73).

[25] Rumelhart D.E. & McClelland, J.L. (1981). An interactive activation model of context effects in letter perception: Part 2. The contextual enhancement effect and some tests and extensions of the model. *Psychological review, 89* (60-94).

[26] Santos, P.J., Baltzer A.J., Badre, A.N., Henneman, R.L. & Miller, M.S. (1993). On handwriting recognition performance: Some experimental results. *Proceedings of the Human Factors Society 36th Annual Meeting* (283-287).

[27] Schomaker, L.R.B. (1994) The UNIPEN/NICI/HP data collection of Summer/Autumn 1993. *Technical Report of the NICI/HP Collaboration projects on variability in handwriting.*

[28] Schomaker, L.R.B. (1994). *User-interface aspects in recognizing connected-cursive handwriting.* IEE, Savoy Place London, UK, ISSN 0963-3308. Colloquium on 'Handwriting an pen-based input', pp. 8.1-8.3.

[29] Schomaker, L.R.B. & Teulings, H-L. (1990). A handwriting recognition system based on properties of the human motor system. *Proceedings of the 1st international workshop on frontiers in handwriting recognition* (195-211).

[30] Selen, S.C.J. (1994). *The influence of text stimulus modality and kind of unit on writing quality and automatic recognition of cursive script.* Master's Thesis in Cognitive Science, Nijmegen University, The Netherlands.

[31] Senior, A.W. (1992). Off-line handwriting recognition: a review and experiments. *Technical Report CUED/F-INFENG/TR 105* Cambridge University, England.

[32] Simon, J.C. & Baret, O. (1992) Cursive words recognition. In S.Impedovo & J.C.Simon (Eds.), *From pixels to features III: Frontiers in Handwriting recognition* (241-260) Amsterdam: North-Holland.

[33] Souya, T., Fukushima, H., Takahashi, N. & Nakagawa, M. (1991). Handwriting interface for a large character set. *Fifth Handwriting Conference of the International Graphonomics Society* (166-168).

[34] Srihari, S.N. & Srihari, R.K. (1994) Written language recognition. *Technical Report, CEDAR-TR-94-1.*

[35] Stephen, G.A. (1992) String Search. *Technical Report TR-92-gas-01*, School of Electronic Engineering Science, University College of North Wales.

[36] Suen, C.Y. (1983) Handwriting, generation, perception and recognition. *Acta Psychologica 54* (295-312) North-Holland.

[37] Suen, C.Y., Guo, J. & Li, Z.C. (1992). Computer and human recognition of handprinted characters by parts. In S.Impedovo & J.C.Simon (Eds.), *From pixels to features III: Frontiers in Handwriting recognition* (223-236) Amsterdam: North-Holland.

[38] Wells, C.J., Whitly, P.E. & Whitrow, R.J. (1995). Fast dictionary look-up for contextual word recognition. *Pattern recognition, 23* (501-508).

[39] Whitrow, R.J. & Higgins, C.A. (1987). The application of n-grams for script recognition. *Proceedings of the 3rd International Symposium on Handwriting and Computer Applications* (92-94).

[40] Zuniga, C.M. de, Humphreys, G.W., Evett, L.J. (1991). Additive and interactive effects of repetition, degradation, and word frequency in the reading of handwriting. In D.Besner and G.Humphreys (eds.), *Basic Processes in Reading: Visual Word Recognition..*