Language Technology in a Predictive, Restricted On-screen Keyboard with Dynamic Layout for Severely Disabled People

Anders Sewerin Johansen, John Paulin Hansen, Dan Witzner Hansen
The IT University of Copenhagen
Glentevej 67,
2400 Copenhagen, Denmark
{dduck, paulin, witzner}@it-c.dk

Kenji Itoh, Satoru Mashino
Tokyo Inst. of Technology,
2-12-1 Oh-okayama Meguro-ku,
Tokyo 152-8552 Japan,
ken@ie.me.titech.ac.jp

Abstract

This paper describes the GazeTalk augmentative and alternative communications (AAC) system, and presents results from two user studies of initial typing rates among novice users. GazeTalk can be operated using eye tracking, mouse or other pointing devices. The system presents the user with a user interface that is based on 12 large on-screen buttons.
GazeTalk supports a wide range of configurations, including several variants of probabilistic or ambiguous/clustered keyboards. The language model used in GazeTalk is based on a corpus constructed on the basis of text extracted from Usenet discussion groups. The results from the user studies indicated that the prediction-based input system was less efficient than a static layout. However, user comments suggest that this was mainly caused by design related factors, which were not directly related to the basic design principles. In the next design iteration, we aim to improve the design by eliminating the problems and to increase the quality of the language model by using a significantly larger training corpus.

1 Introduction

We are currently developing an eye-typing communication tool – GazeTalk – for people who suffer from amyotrophic lateral sclerosis (ALS), also known as Lou Gehrig's disease, who have lost their voice and mobility and may only be able to move their eyes. To make the tool widely available, we use standard PCs and low-resolution, off-the-shelf digital cameras to track eye positions. As the precision of eye-tracking is very dependent on the quality of the cameras used, we decided on a basic layout, which consisted of few, large on-screen buttons.

In many cases eye-tracking systems have to deal with the "Midas Touch" problem: Everywhere you look, a command may be activated without your intention (Jacob, 1991). A dwell time activation principle has been the preferred solution to this problem since the very first eye-tracking based communication systems (Hutchinson, 1989; Majaranta and Räihä, 2002). The dwell time period is typically between 500 and 1500 milliseconds.

1.1 GazeTalk –goals and design

The goal of the GazeTalk project is to develop an eye-tracking based Augmented and Alternative Communication (AAC) system that supports several languages, facilitates fast text entry and is both sufficiently feature-complete to be deployed as the primary AAC tool for users, yet sufficiently flexible and technically advanced to be used for research purposes. The system is designed for several target languages, initially Danish, English

and Japanese. The system has a built-in voice output subsystem, that supports the use of either external (usually synthetic) voice, using the SAPI API or the Windows clipboard, or an internally developed digitized voice system. This digitized voice is currently only available in Danish, and has a vocabulary of approximately 33.000 words.

The motivation for the project was to explore and eventually deploy eye-tracking as an input modality for ALS patients, and to explore issues in text input in constrained user interfaces (UIs), such as mobile devices. We explore the relationship between text input in AAC systems and mobile devices extensively in (Johansen and Hansen, 2002).

The GazeTalk system consists of three elements:

- The actual GazeTalk program, which implements the basic UI elements and functionality (email, voice output, saving and loading documents etc.).
- A layout editor – StateEditor – which is used for designing and building the layouts used by GazeTalk.
- A library of layouts built with StateEditor, which are designed for various situations (deployment, research etc.).

All layouts in the library use the four by three matrix format shown in Fig. 1, which allows for a maximum of 12 active on-screen buttons.

GazeTalk and StateEditor support various text-input modes, which include direct selection of letters, letter- and word prediction/completion and several types of ambiguous/clustered keyboards.

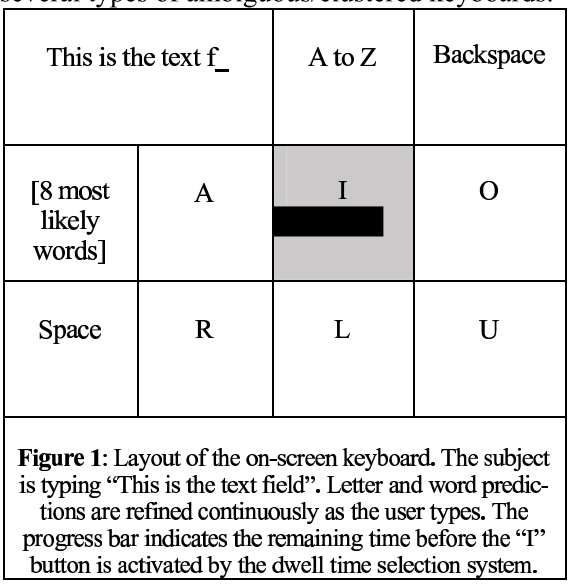

Figure 1: Layout of the on-screen keyboard. The subject is typing "This is the text field". Letter and word predictions are refined continuously as the user types. The progress bar indicates the remaining time before the "I" button is activated by the dwell time selection system.

GazeTalk contains a wide variety of reporting functions which allow an experimenter to measure variables such as average selection speed; word per minute (WPM) rate; pointer trails in terms of entry, exit and time spent per button; the type and specific content of buttons inspected and selected: as well as keeping track of the user's navigation within the system.

2 Language technology in GazeTalk

The decision to use affordable off the shelf cameras for the eye tracking subsystem rather than expensive, custom equipment imposed severe restriction on the number of UI elements available in the basic layout. Since we could not expect to attain a level of performance from the eye-tracking system that would allow us to use a full-size on-screen keyboard, we decided to investigate alternative approaches. Language technology emerged as a potential means to provide fast text input, despite the small number of on-screen buttons.

2.1 Language models

Measurements of the information content of written language indicate that, given a sufficiently efficient language model, only approximately one bit of information is needed per character (Shannon, 1951). Language models can be based on many different natural language processing (NLP) algorithms with varying results in terms of performance, capabilities and resource consumption. A simple language model (frequency count) achieves an entropy of 4.03 bits per character (Shannon, 1951), and an advanced state-of-the-art model (maximum entropy) achieves 1.2 bits per character (Rosenfeld, 1996). The traditional n-gram approach is relatively high performing at 1.5 bits per character (Tilbourg, 1998). David J. Ward tabulates the performance of most current approaches and discusses the construction of language models extensively in (Ward, 2002).

Most language models need to be primed in order to perform optimally. In the case of a trigram model, which predicts a word on the basis of the two preceding words, it needs to be primed with two words in order to supply any predictions at all. In other words, most (possibly all) language models will supply unreliable predictions for the first few words or characters of input. As a conse-

quence, when designing text input systems based on language models, one must allow for the possibility that the predictions are wrong.

One of the most common problems when using word-level language models is the dictionary problem, also known as the out of vocabulary (OOV) problem: What happens when the user wants to enter a word which is not in the dictionary? It is possible to estimate the probability of an unknown word, but it is obviously impossible to predict the actual (unknown) word. It is not just impractical to add all known words to the dictionary (Websters Ninth New Collegiate Dictionary boasts almost 160.000 entries!) – it is virtually impossible as new words are constantly added to the vocabulary of all living languages, and as the number of proper nouns (names of people, places, products, pets...) is virtually unlimited. This is a problem of great concern with regards to GazeTalk, as users of an AAC-system are very likely to develop slang and abbreviations that are well understood by caretakers in a daily communication context but incomprehensible to outsiders and not represented in any dictionary.

2.2 The GazeTalk language model

GazeTalk currently only supports the use of letter- and word prediction/completion when using European languages. Faced with the choice between a two-stage hierarchic and a probabilistic input configuration in the European versions, we decided on a six-key probabilistic on-screen keyboard as the basic input strategy. Given that the cost of a prediction hit is one selection, and the cost of a prediction miss is three selections – one to access the hierarchical input system, and two to select the desired letter – the probabilistic keyboard will be superior to the hierarchical, if the prediction algorithm can supply the desired letter as one of the top six predictions more than 75% of the time. Obviously this simple calculation ignores the additional cognitive load placed on the user by the need to search for the desired letter or word. However, as the advantage of the probabilistic keyboard is potentially very big (e.g., 1.2 selections per symbol compared to two selections on a two-level static, hierarchical system, given that the prediction algorithm includes the desired letter among the top six candidates 90% of the time), we decided to base our prototype on a probabilistic keyboard.

Both the word- and letter-level language models in the current version use a Katz-style back-off Markov model as described in (Katz, 1987). The actual letter predictions are generated by a back-off algorithm, which consults both the word- and letter-level models. As the word-level predictions are more likely to be correct, since they are based on a higher-level context than the letter based model, the letter prediction algorithm gives higher priority to the predictions from the word level model.

2.3 Corpus construction

As we did not at the time have access to either dictionaries or corpora, we decided to construct our own. The current Danish language model is based on a corpus collected from the Danish Usenet discussion groups, specifically the groups dk.snak (general conversation) and dk.helbred.handicap (special interest group for disabilities). The decision to extract a corpus from these sources was based on both the availability of the data, and the fact that the expected language exhibited by the users would be conversation-level Danish, which is the primary form of language used in the Usenet groups. The use of Usenet posting as a basis for a corpus is also attractive from a legal standpoint. It is generally agreed that by posting on Usenet, the author de facto grants everyone the right to store and otherwise process the material produced.

We collected an initial corpus of approximately 1.5 million words, and extracted and proofread a vocabulary of approximately 33.000 words from this, based on their frequency in the corpus. Unfortunately the initial corpus contained a lot of undesirable features, including (but not limited to) use of other languages than Danish (primarily English, Swedish and Norwegian), bad spelling, excessive or missing punctuation, slang and imitation of speech-level language (e.g., thought-sounds equivalent to “ah” and “uhm”, as seen in transcriptions of conversations). We therefore applied several heuristics to the corpus, using the proofread vocabulary, in order to filter out the unwanted features. These heuristics primarily consisted of discarding sentences containing “stop words”, mainly frequent non-Danish words; discarding sentences that were extremely short or extremely long, on the assumption that they were artifacts of missing or excessive punctuation; and discarding sentences that contained more than one or two out-of-

vocabulary words. This resulting corpus consists of approximately 467.000 words, a reduction of approximately 2/3 from the initial corpus.

2.4 Adaptive vocabulary

As a means to increase the performance of the language model in face of a small vocabulary and training corpus, we decided to implement automatic collection and integration of vocabulary and n-grams during daily use. This feature has been reported to increase user acceptance and text production rate significantly in (Carlberger, 1997) and (Darragh et al., 1990). Additionally, measurements of cross training performance indicate that an in-domain model performs significantly better than an out-of-domain model, even for very large training sets and very similar test sets (Rosenfeld, 1996). We confirmed this by performing perfect-user simulations, i.e., simulations where the number of selections needed to input a test corpus is computed on the assumption that the user will make no selection errors, and always use the most efficient input strategy in terms of the number of selections used.

The user n-grams are integrated in the combined word level model by giving priority to n-grams on the basis of length and source. Thus, e.g., the combined model will consider predictions based on user generated trigrams more precise than those based on trigrams from the base model, but predictions based on user generated bigrams are considered less precise than predictions based on trigrams from the base model.

2.5 Clustered/ambigous keyboards vs. dynamic/predictive keyboards

When faced with the task of designing a text input system on a reduced keyboard, there are three obvious alternatives: A traditional hierarchic system, as seen in the "Multi Tap" input method; a clustered or ambiguous system, such as the Tegic T9 system or Kühn and Jörn (2001); and a dynamic keyboard that reassigns the content of the buttons based on current likelihood. Each has advantages and disadvantages.

The hierarchical system is inherently inefficient, but the static nature of the system allows for eyes-free operation, rote learning and – perhaps more importantly – it allows for unambiguous letter input, which is the input mode that users expect from any other way of entering text (typewriters, longhand, word processing etc.).

An ambiguous input system has the potential to perform significantly better than a static, hierarchical system, and allows for eyes-free operation on a per-word basis. Given a sufficiently powerful language model (or a small dictionary, that the user is able to memorize) it does support full, or almost full eyes-free operation. It is however suffering from the OOV problem: Unless augmented by morphological algorithms as seen in Rau and Skiena (1995), it is necessary to allow for an alternative input method, for the user to be able to enter words not found in the vocabulary.

A dynamic keyboard also has a potential for high performance, but does not allow for eyes-free operation or rote learning. However, it does conform to the user's expectations that letter entry is unambiguous and final.

In conclusion, both ambiguous and dynamic keyboards are potential high-performing alternatives to the static, hierarchic layout, but they both require the user to accept and adapt to an input system that does not conform to their expectations. Whether the users are more likely to accept ambiguous input or dynamic letter placement is an open research issue, which we hope to explore further using the GazeTalk system. As a consequence, the GazeTalk system supports both static, hierarchic keyborad layouts, as well as dynamic and ambiguous keyboards.

2.6 Layout considerations

In this, the first series of user experiments, we decided to focus on the intial performance among users presented with a dynamic keyboard layout. The built-in context-sensitive letter prediction algorithm was used to supply the six most likely letters for the dynamic keyboard. They were then placed according to the workings of the parafoveal vision, i.e. with the most probable suggestion in the center position ('I' in Fig 1), and the other suggestions placed according to probability in a clockwise fashion around the center position ('O', 'U', 'L'. 'R' and 'A' in Fig. 1). This was done on the assumption that users would quickly learn to anticipate the placement of the desired letter and then – in case the letter prediction did not supply this as the primary candidate – be able to evaluate the

other candidates with a minimum of eye movement.

Furthermore this mode featured buttons for backspace and space, as well as buttons for access to word prediction/completion mode and an alphabetical letter entry mode ("A to Z"). The word prediction/completion mode presented the current eight most likely words (the actual words are shown on the "eight most likely words" button) in a four by two matrix and featured buttons for access to alphabetical letter entry mode and the primary letter entry mode. The system remains in word completion mode, until the user explicitly exits, in order to encourage the user to use the word predictions. This may not be the optimum strategy for a dictation task, but the goal of the GazeTalk project is to accelerate a text composition task. When engaged in a composition task, it's a reasonable assumption that the user will often accept a synonym for the desired word, rather than spending time explicitly requesting the desired word. This feature may lead to so-called "parrot speech", i.e., that the text composed with the system lacks the individual character that text produced by other means would exhibit, but as the primary concern is to aid a disabled user who struggles to keep up with normal spoken conversation (which is often conducted at rates in excess of 150 WPM) we consider this an acceptable design trade-off.

The alphabetical letter entry mode enabled the user to select the desired letter in a two-stage process, by first selecting a group of letters (e.g. "ABCDEFG") containing the desired letter, and then the letter.

3 Prototype tests of GazeTalk

During the design process, we conducted experiments and user evaluations to determine the initial effectiveness of various designs. Our focus on initial performance is motivated by our concern that early impressions of system effectiveness may have a major impact on the users' determination to use a new system. User motivation is of highest importance to AAC systems as they are often introduced in periods of life crisis, e.g., during recovery from an accident or during a serious progressive disease such as ALS.

In our first experiment we performed a between-subjects comparison of differences in performance between mouse-click typing and mouse-dwell typing on a Danish on-screen keyboard with letter- and word prediction among novice users (initial performance) (Hansen et al., 2003). Users of dwell-time activation showed longer selection times and larger overproduction rates than users of mouse click activation. The average productivity in terms of WPM was 5.51 and the overproduction rate (unnecessary clicks and error corrections) was 16.9% for click selections, while the group of users with dwell selections produced 4.79 WPM on average and had an overproduction rate of 26.2% (Hansen et al., 2003).

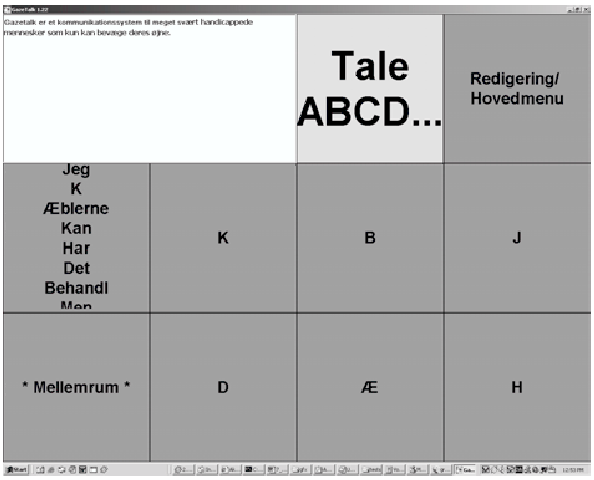

Figure 2: The Danish version of GazeTalk.

Our second experiment was a within-subjects study of an input system for Hiragana and Katakana ("Kana") characters, which used the same basic layout and UI-elements as the Danish system (cf. Fig. 2 and 3). Each subject made dwell-time selections using either a mouse or an eye-tracker. The Japanese system had no prediction functions. Instead, it displayed the characters in a conventional, static, two-level hierarchical manner. The purpose of the second experiment was to compare mouse dwell selections with eye dwell selections at an initial performance level. Selection by dwell-time activation using mouse and eye-tracking interaction was found to be almost equally fast, but using the mouse as pointing device was far more precise than eye-tracking. Consequently, the productivity in terms of characters per minute (CPM) was 33% higher when using the mouse, (22.1 CPM versus 16.6 CPM) (Hansen et al., 2003).

The two experiments suggests that users in general can be productive from the very first encounter

with a dwell-time based system, but the productivity depends on the familiarity with the input display structure and the input mode (i.e., hand or eye).

Based on a comparison of 20 similar sentences in Danish and Japanese, we estimated that it takes two Japanese characters to produce the equivalent of one Danish word. Using this ratio, WPM can be compared to CPM. There was a large difference between the Danish average production of 4.78 WPM and a Japanese average production of 22.5 CPM, which equals 11.3 WPM.

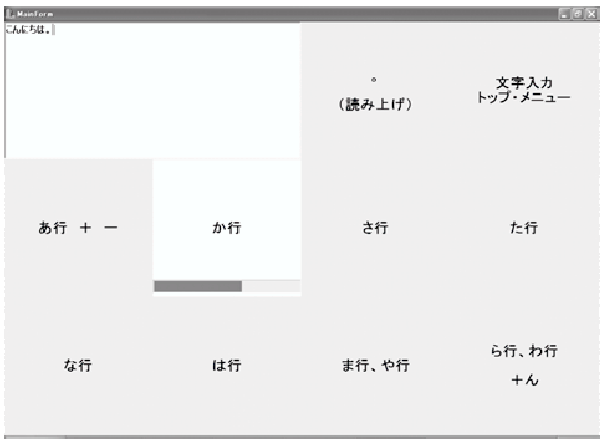

Figure 3: The Japanese version of GazeTalk.

The problems experienced by initial users of highly ambiguous displays, as indicated by the markedly lower productivity of Danish users who used the predictive layout, when compared to the Japanese users, who used a static layout, are most likely primarily caused by confusion with regard to the position of the desired letter. We assumed that the users would appreciate that the letters were positioned according to their current probability, but that turned out not to be the case. Apparently this placement strategy was counterintuitive, and prevented them from developing an efficient strategy for selecting letters, as they found themselves constantly scanning the on-screen keyboard in order to locate the desired letter, as evidenced by this post-experiment comment from one of the subjects:

"It's a bit confusing that the letters change position after each selection. You spot the 'T' in one position, and then you have to find it all over again after the next selection."

4 Future improvements

Obviously, scanning for a target letter or word among predicted candidates consumes time, and may give rise to frustration. The design implications are that reduced keyboards should strive to introduce regularities and invariants which conform to the expectations of the users in the dynamics whenever possible. As a consequence, for the next prototype we have decided to assign "home positions" for all letters. Thus, a "T" will occur on the same button whenever the prediction algorithm determines it's a likely candidate for the next letter, unless another letter that shares the same home position has a higher probability. The placement algorithm selects the distribution of positions that minimizes the number of home position conflicts. The home positions are assigned by sorting the letters according to their static probability, and then assigning them positions in a round-robin fashion. Thus, given that the most common seven letters according to frequency counts in the Danish corpus are "ERTNDAI", 'E' and 'I' share the same home position. The purpose of this home position assignment strategy is to minimize the number of placement conflicts.

The present WPM rates achieved by novice users of GazeTalk are compatible with the productivity found in e.g. SMS text-production (James and Reischel, 2001) and compares well to the typing speed of other advanced AAC-systems with a limited number of (large) buttons, e.g. (Kühn and Jörn, 2001). However, we believe that a considerable increase in text production rates may be achieved in two ways:

- By including a language model specific to the individual user.
- By increasing the size of the corpus in the general language model.

We have therefore developed a companion program for GazeTalk, which allows the user to train the user generated language model on any text material which can be transferred to the Windows clipboard or exported to ASCII or RTF format. As most computer users maintain a personal archive in the form of sent email and other documents, it is feasible to use this as a personal corpus, with the aid of out language model training tool.

Moreover, we will continue our work on expanding the corpus and vocabulary. We expect to use a much larger open-source vocabulary (Den

Store Danske Ordliste - http://da.speling.org/), which contains approximately 370.000 wordforms. Also, we have expanded the base corpus by including a much wider range of Usenet groups. We expect the resulting corpus to consist of approximately 40 million words after application of the previously mentioned heuristics. Expanding the vocabulary and corpus by nearly two orders of magnitude will also necessitate major revisions of the implementation of the language model.

Conversion from Kana to Kanji (Chinese) ideograms is a most immediate concern for the Japanese users and will thus be implemented in the next Japanese version of GazeTalk.

5 Conclusion

Performance among first time users of the prediction-based input system was not as high as we had hoped, but the experimental measurements and user comments have indicated several ways to improve the system. Although we did not succeed in realizing the inherent performance advantage of a prediction-based input system, it is encouraging that the users appeared to accept the dynamic input configuration without any major reservations.

As an initial study of alternative input methods, we consider the experiment a success, as it validated the usefulness of GazeTalk as a test bed for alternative input configurations, and identified several factors that impact novice user performance. It is specifically worth noticing that novice user performance appeared to be dominated by design related factors, rather than language technology related issues.

Acknowledgments

The GazeTalk project is supported by The Danish Ministry of Science,

The Nordic Academy for Advanced Study (NorFA) provides grants for the first author.

References

Carlberger, J (1997) ; Design and Implementation of a Probabilistic Word Prediction Program ; *NADA report TRITA-NA-E9751*, Swedish Royal Institute of Technology, dept. of Numerical Analysis and Computer Science (NADA), 1997.

Darragh, John J., Witten, Ian H., James, Mark J. (1990) "The Reactive Keyboard: A Predictive Typing Aid", *IEEE Computer, 23(11)*:41--49, November 1990

Hansen, John Paulin ; Johansen, Anders Sewerin; Hansen, Dan Witzner; Itoh, Kenji ; Mashino, Saturo (2003): Command Without a Click: User Studies of Dwell Time Typing by Mouse and Eye-Gaze Selections. Paper available at: http://www.it-c.dk/research/EyeGazeInteraction/Papers/Hansen_et_al_2003.pdf

Hutchinson, T. E., White, K. P., Martin, W. N., Reichert, K. C. & Frey, L. A. (1989) "Human-Computer Interaction Using Eye-Gaze Input" *IEEE Transactions on Systems, Man and Cybernetics, Vol. 19*, No. 6, November/December 1989, Pages 1527 – 1534.

Jacob, J. K. (1991) The Uses of Eye Movements in Human-Computer Interaction Techniques: What You Look At is What You Get, *ACM Transactions on Information Systems, Vol. 9*, No. 3, April 1991, Pages 152 – 169.

James, Christina L. ; Reischel, Kelly M. (2001) "Text Input for Mobile Devices: Comparing Model Prediction to Actual Performance", *SIGCHI'01 March 31-April 4, 2001*, pp. 365-371, Seattle, WA, USA

Johansen, Anders S. & Hansen, John P. (2002): Augmentative and alternative communication: The future of text on the move. *Proceedings of 7th ERCIM Workshop "User Interfaces for all"*, 23 - 25 October 2002, Paris (Chantilly), France. Pages 367 - 386

Katz, A (1987) Estimation of probabilities from Sparse Data for the Language Model Component of a Speech Recognizer" *IEEE Transactions on Acoustics, Speech and Signal Processing, VOL ASSP-35*, no. 3, March 1987

Kühn, Michael ; Garbe, Jörn (2001) Predictive and Highly Ambiguous Typing for a Severely Speech and Motion Impaired User, *Universal Access in Human-Computer Interaction.Proc. of UAHCI 2001*, New Orleans, August, 5-10. Mahwah (NJ): Lawrence Erlbaum Associates

Majaranta, P. & Räihä, K-J (2002) Twenty Years of Eye Typing: Systems and Design Issues, *Proceedings of the Symposium on ETRA 2002: Eye Tracking Research & Applications Symposium 2002*, New Orleans, Louisiana. Pages 15 – 22

Rau, H ; Skiena, S. S. (1996) Dialing for Documents: An Experiment in Information Theory, *Journal of Visual Languages and Computing 7(1)*, pp. 79-95

Rosenfeld, R (1996) A Maximum Entropy Approach to Adaptive Statistical Language Modelling, *Computer, Speech and Language*, 10, 1996, pp. 187-288

Shannon, C. E (1951) Prediction and Entropy of Printed English, *Bell Systems Technical Journal* 30, January 1951, pp. 50-64

Tilbourg, H (1998) An Introduction to Cryptology, 1988, Kluwer Academic Publishers

Ward, David J. (2002) Adaptive Computer Interfaces, Ph.D.-thesis, Inference Group, Cavendish Laboratory, University of Cambridge, November 2001, http://www.inference.phy.cam.ac.uk/djw30/papers/thesis.html, verified January 13 2003