TITAN: A Cross-linguistic Search Engine for the WWW

Yoshihiko Hayashi, Gen'ichiro Kikui, Seiji Susaki
NTT Information and Communication Systems Laboratories
1-1 Hikari-no-Oka Yokosuka, 239 Japan
{hayashi, kikui, suzaki}@nttnly.isl.ntt.co.jp

Abstract

Although various search services are available on the World-Wide Web, there are only a few services that can help people access web pages in a foreign language. In order to solve this problem we have developed a cross-linguistic WWW search engine, TITAN[1], which provides users with an interface to query in their native languages (e.g.,Japanese) while performing multilingual searches on the WWW. This paper outlines TITAN, and details two modules in the system, which are essential in cross-linguistic search of web pages; a module for identifying languages of the pages, and a simple MT module, which performs bi-directional translation between Japanese and English, employed by the navigation interface of the system.

Introduction

Major search engines currently available on the WWW, such as Yahoo, Lycos, and Alta Vista, are vital tools in navigating the global information space, and widely used. These engines locate web resources containing or relating to user's topic of interest.

However these major search engines are primarily for English-speaking people. That is, many of them are monolingual (English-based), while others are 'language indifferent', in the sense that they search for documents by string matching. Therefore users whose native language is not English must put much effort into translating their queries into English in which target information might be written. Also they must choose the desired documents from the listing of titles given in English, even if the search is successful.

Above is just an example with English-based search engines and non-English-speaking users. As (Oard & Dorr 1996) surveys, cross-linguistic search facilities are becoming more important and demanding even for English-speaking people, if they want information in other languages.

In summary, to solve/ease these problems, search systems should have the following cross-linguistic features:

F0 The system should properly handle (analyze, index, retrieve) a web page as long as they are written in one of the system's target languages.

F1 The system should be able to form a cross-linguistic query from a user query given in a language of his/her choice.

F2 The system should present an outline of the page in the user's native (or selected) language.

However the list does not end here. For example, if a user wants to retrieve documents only in English and his query word is 'internet', then the results might include French or Japanese documents as long as they contain the string 'internet'. This is a problem with the 'language indifference' systems, and the problem becomes serious, if the search space is global and unrestricted. We therefore add the following to the list.

F3 The system should provide users with an option to restrict their searches by specifying document languages.

The next section overviews our novel cross-linguistic search system, TITAN, and the following sections detail modules of the system related to the cross-linguistic issues.

Overview of TITAN

TITAN accepts query phrases in the native language of the user (for now, English and Japanese), and searches for web pages that contain given query phrase or their translation (into English and Japanese). The user also can restrict his/her search by specifying the server domains and/or page languages.

The search results presented to the user with a query is a list of URLs (Uniform Resource Locators) of the searched pages with annotations. Translations of the page titles are also given in order to help user select a relevant page from the list.

The system is a typical 'robot-based search engine' (Cheong 1996) augmented by two particular functions for the cross-linguistic retrieval. The overall architecture of the system is shown in Figure 1.

The system can be divided into three parts: a catalog database builder, a navigation interface, and a

[1] http://isserv.tas.ntt.co.jp/chisho/titan-e.html

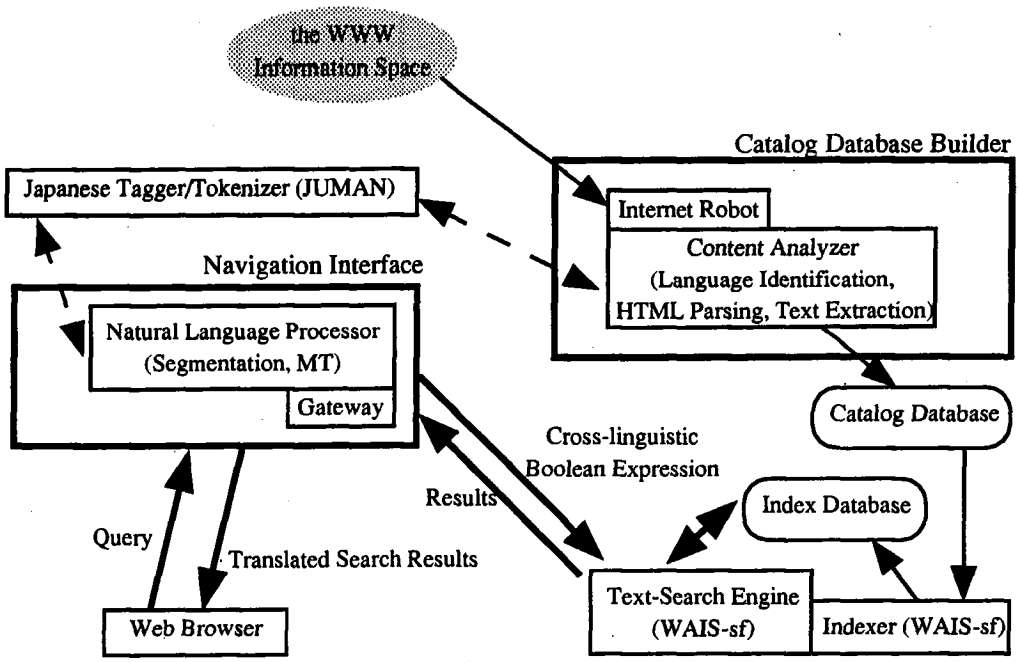


Figure 1: Overall architecture of TITAN.

text-search engine. As the text-search engine, including the indexer, utilized by TITAN is WAIS-sf [2], a popular text-search engine, the following subsections outline the former two parts.

Catalog Database Builder

The catalog database builder is responsible for creating a database called a "catalog." Conceptually, the database is a compressed and annotated representation of the target search space, while it is a set of files, called content files, actually.

The catalog database builder consists of an internet robot and a content analyzer. The former automatically traverses the WWW space, which is a network of WWW documents (e.g., HTML files), starting from a given list of URLs basically in breadth-first order. It fetches the web page of every visited URL and sends it to the content analyzer.

The content analyzer creates content files in the following steps.

1. Identifies the coding system and the language of the given web page.
2. Parses the page structure if marked up by HTML, and extracts text segments along with their surrounding HTML tags.
3. Tokenizes the text segments if it includes a Japanese string.
4. Eliminates (Japanese) stop words.

The first step will be detailed in the next section. The third and the fourth steps handle only Japanese portions of the page, because paragraphs in English (and other Western-European languages) are processed by the indexer of WAIS-sf.

For some Eastern-Asian languages, including Chinese, Japanese, and Korean, it is not easy to segment a text into words, because a text has no explicit word delimiters (e.g., spaces in European languages). The system carries out the third step by using JUMAN (Matsumoto et al. 1994), a morphological analyzer for Japanese texts.

Normally, IR systems remove a set of stop words, useless words for information retrieval, from the index terms (for example, (Fox 1992)). The problem is how to define "useless words." TITAN uses a simple definition relying on syntactic categories assigned by the morphological analyzer. The system regards nouns and the 'unknown category' as "useful words." The remaining categories (including functional words and some content words) are regarded as stop words and are discarded.

In addition to the above mainstream process, the content analyzer classifies web pages with two measures. One measure designates to what extent a web page contains information in itself. This is done with a heuristic which refers to the text length and number of paragraphs. The other measure is associated with number of hyper-links which the page provides. As described later, this additional information is presented

[2] ftp://ls6-www.informatik.unidortmund.de/pub/wais/FAQ

to users in order to help they choose a relevant page from the result list.

The resulted content files are paired with their URLs, and converted to index file (inverted file). This process is managed by WAIS-sf indexer, which is partially modified for East-Asian languages, specifically Japanese.

Navigation Interface

The navigation interface is responsible for providing an interface between the user and the catalog database. It is invoked from WWW clients via CGI (Common Gateway Interface). Figure 2 displays the input form of the system.

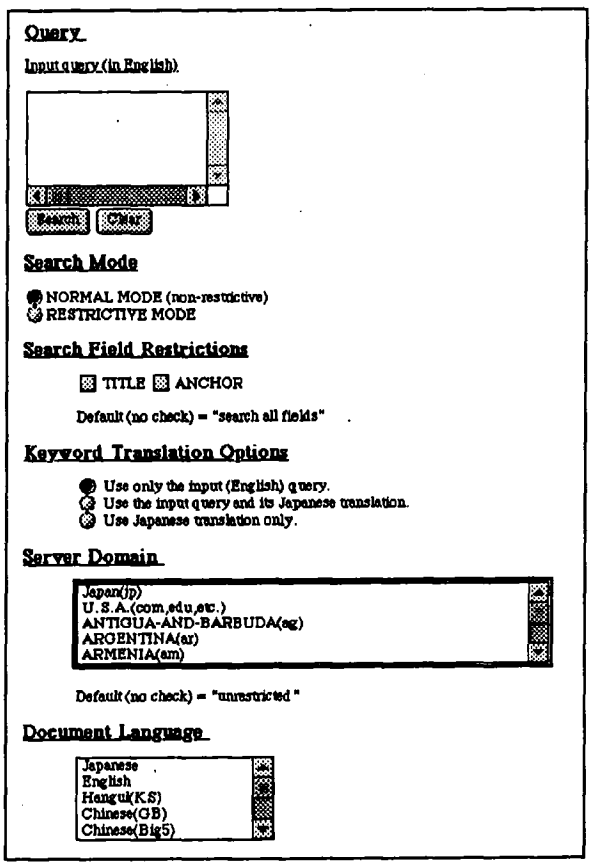


Figure 2: Query input form.

The interface interprets user queries, then converts them into Boolean expressions consisting of words in the original query and its translation. The text-search engine (WAIS-sf) searches for content files that match the converted expression. If matching files exist, a search result page is presented.

Converting User's Query: The system accepts a phrase in English or Japanese. First, if the phrase is in Japanese, it is segmented into words. Then, the resulting words are translated. If the original phrase is in Japanese, it is translated into English. If the original phrase is in English, it is translated into Japanese. Finally, the original and translated words are combined with conjunctive (AND) or disjunctive (OR) operations.

Creating Output Page: The search results are formatted into a page which consists of the query section and the result section. The query section contains the input query and its Boolean form. The result section shows web page information, with the line entries displayed in the order of the highest to lowest score reflecting how closely they match the Boolean expression converted from the user's query. As shown in Figure 3, each line contains the original title of the page, its Japanese translation, and some icons that correspond to the information analyzed by the content analyzer, such as the server domain, media (MIME) type, page type, and document language. These annotations are helpful for a user to choose a page from the result list. For example, it is easy for a user to choose a page submitted from the U.S.A. with many hyper-links.

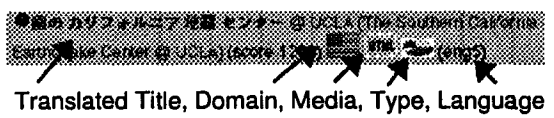


Figure 3: An example of the searched item.

Cross-linguistic Modules

It may be good to summarize the roles of the two cross-linguistic modules of the system here.

- Language Identification (LI) module (embedded in the catalog builder).

  It is inevitably necessary to correctly apply language dependent processes (e.g., segmentation, tagging, and stemming) in the text extraction phase. This is essential for the required feature F0. Also, the information about what language each document is written in enables the system to realize the search option based on document languages, which is the required feature F3.

- Machine Translation (MT) module (embedded in the navigation interface).

  The MT module is bi-directional; one way is used for converting a user's query into a cross-linguistic Boolean form. This achieves the required feature F1. The other way is used for translating titles of the searched web pages. This partially satisfies the required feature F2.

The Language Identifier

Automatic language identification has been discussed in the field of document processing. Several statistic

models have been tried including using the N-gram of characters (Cavner & Trenkle 1994), diacritics, special characters (Beesley 1988), and using the word unigram with heuristics (Henrich 1989). Among these methods, the result of N-gram statistics (Cavner & Trenkle 1994) shows the best accuracy level, over 95%.

These methods, however, are not sufficient for documents on the WWW, mainly for two reasons. First, they presuppose that the target document is correctly decoded. This is hard to expect for the unrestricted web pages coming from all over the world. Also, it is sometimes difficult to properly identify the coding system of a page without knowing the language in which it is written. Second, they cannot handle East-Asian languages, because they presuppose that the target document is easily segmented into words, which does not hold true for these languages.

To deal with the first problem, our algorithm (Kikui 1996) uses a statistic language model to select the correctly decoded string as well as to determine the language. To solve the second problem, our algorithm combines two statistic language identification modules: one is for Western European languages, and the other is for East-Asian languages.

The Overall Algorithm

As there are web pages in which more than one languages are appeared, the overall algorithm is designed for handling texts that contain at most one East-Asian language and one West-European language.

The algorithm first extracts East-Asian language parts in the input document if they exist, and identifies their language. The algorithm then identifies the rest part as one of West-European languages.

In both steps, a process responsible for decoding the input code sequence into text string, and identifying its language is employed. We call the process 'identification process'.

Extracting East-Asian Language Parts: It is easy to identify East-Asian characters in a code string if it contains escape sequences defined in ISO 2022. This is because these escape sequences explicitly indicate the character set (e.g., Korean characters or the West-European alphabet) of every part in the code string. For code string without escape sequences[3], this step is further divided into the following two sub-steps. First, a tentative set of East-Asian substrings are estimated for each East-Asian coding system by applying regular expression patterns to the input string. For example, if a sub-sequence consists of an even number of bytes greater than 0xA1[4] and smaller than 0xFE, this sub-sequence is estimated as Japanese Kanji characters encoded with EUC-JIS, or Korean characters encoded with EUC-KS etc. The identification process for East-Asian languages is then applied to each tentative set of code substrings. If the identification process fails for any of the tentative set, the algorithm judges that there are no East-Asian parts in the document, and the identification process for West-European languages is invoked.

The Identification Process: Figure 4 exemplifies an identification process. It first decodes the given code string for every possible coding system. For example, the code string shown in Figure 4(a) is decoded into the strings in Figure 4(b). Then, the statistic-based language identifier calculates the most likely language and its likelihood score for each decoded string by using statistic-based language models described in the next subsection. For example, for each string in Figure 4(b), the algorithm outputs the language and its score as shown in Figure 4(c). Finally, the algorithm chooses the decoded string with the highest likelihood score. Since the highest score in Figure 4(c) is -6.9 for String-1 (i.e., the result of Decoder1), string-1 is the most likely decoding and its language is estimated as Japanese. Note that an identification failure can be occurred, when every invoked statistic-based language identifier returns "fail."

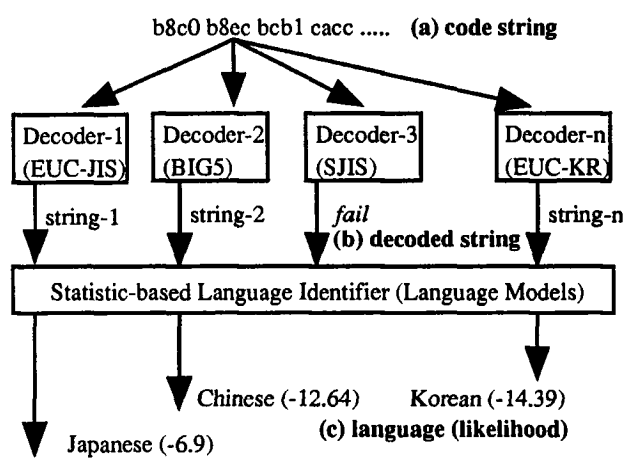


Figure 4: An example of the identification process.

This process is implemented in two separate modules: one is only for West-European languages and the other is only for East-Asian languages. These modules employ decoders and statistic language identifier associated with their target languages, while they share the same architecture.

Statistic-based Language Identifier

Each statistic-based language identifier regards a text as a list of tokens where a token is a word in European languages, or a character in East-Asian languages. The identifier first calculates the likelihood of a text (=

[3]Some coding systems, such as EUC-JIS or BIG5, indicates East-Asian characters by setting eighth bits of some character codes.

[4]0x stands for the hexadecimal sign.

a list of tokens) with regards to each language, then chooses the language with the highest likelihood. If the highest likelihood does not exceed a predetermined threshold, it returns "fail."

The likelihood of a list of tokens with regard to a language is the product of unigram probabilities for the class of every token in the list. The class of a token in a European language is the token itself when it consists of less than n characters, where n is empirically determined, otherwise, the last n characters following "X-". For example, if n is set to 4, classes of "the" and "searching" are "the" and "X-hing," respectively. For East-Asian languages the class is defined as the token (i.e., character) itself [5].

Formally, let $TEXT$ be the set of tokens in a text, then the likelihood of $TEXT$ with regard to language l is given as follows.

$$P(TEXT, l) = \frac{1}{N} \sum_{t \in TEXT} log(P_l(t))$$

Here, $P_l(t)$ is the unigram probability of token class t in language l, which is estimated from the text corpus (called the 'training corpus') in language l, and N is the number of tokens in $TEXT$.

Experiments on Language Identification

The Corpora: We collected about 100,000 web pages using our internet robot. From these pages, sample pages were extracted and correct languages were assigned to them in the following way.

First, the collected pages are classified into an ASCII set and a non-ASCII set. The former consists of all the files that contain only ASCII characters excluding the escape character (0x0B). The latter consists of the remaining files. Then, a tentative language name was assigned to each file. The tentative name is determined by the domain name of the URL.

Next, the tentative language labels are corrected by humans. The resulting data are 1400 pages in the following languages.

1. European languages
   Danish, German, English, Spanish, Finnish, French, Italian, Dutch, Norwegian, Portuguese, Swedish.
2. Asian languages
   Chinese (traditional, simplified), Japanese, Korean.

These texts are encoded with the following coding systems.

ISO-8859-1 (ISO-latin-1), EUC-JIS (Japanese), EUC-KS (Korean), EUC-GB (Chinese), BIG5 (Chinese), SJIS (Japanese), ISO-2022-JP (Japanese), ISO-2022-KR (Korean), ISO-2022-INT.

These data are then randomly divided into two non-overlapping sets: a training set and a test set.

[5] This is reasonable; for example, a Kanji character carries its own meaning.

Results: We trained the statistic model of each language with 200 kB of web pages in the training set. Then, it is used to estimate languages of 640 web pages taken from the test set.

Table 1 shows the confusion matrix for the Western-European language results. The columns correspond to the outputs from the system, and the rows correspond to the correct answers. The value of n, the length of characters for determining classes, is set to 4, which gives the fewest errors for the training set.

The average error rate is 4.8%. Error occurs when the document is not a normal text (e.g., computer programs, a list of proper names or network addresses). The result shows that our method achieved the level of correctness equivalent to the previous methods that presuppose correctly decoded character strings.

We experimented the algorithm with different sizes of training data, from 50 kB to 400 kB. The accuracy saturates at the size of 100 kB. This results suggests that 100 kB is sufficient for a training corpus of one language. Since the average size of a WWW page is around 8 kB, 100 kB is equal to 120 WWW pages on average. This is small enough and easy to collect.

Table 2: Confusion Matrix for East-Asian(CJK) Languages

jpn=Japanese, kor=Korean, zho=Chinese

	jpn	kor	zho	ELSE
jpn	123	0	0	4
kor	1	7	0	0
zho	2	0	47	0
ELSE	1	0	2	8

Table 2 shows the confusion matrix for East-Asian languages. It displays that the system performs fairly well also for East-Asian languages. Most errors occur when the document includes only a few East-Asian characters. There are many pages which are written mostly in English but only proper names are written in Asian characters.

The Simple MT Module

Again, the role of the MT module is to help users access pages on the WWW by translating input queries and titles of retrieved web pages. Since queries and titles are short and are relatively simple, complete MT systems for full texts are not necessary or are sometimes wasteful.

Cross-linguistic Query Formation

The system converts a user query into a Boolean expression consisting of words in the original query and its translation. It supports bi-directional translation between Japanese, our native language, and English, the dominant language on the Internet.

Table 1: Confusion Matrix for European Languages

dan=Danish, deu=German, eng=English, els=Spanish, fin=Finnish, fra=French, ita=Italian, nld=Dutch, nor=Norwegean, por=Portuguese, swe=Swedish

	dan	deu	eng	esl	fin	fra	ita	nld	nor	por	swe	ELSE
dan	8		1									
deu		47										1
eng		1	96			1					1	4
esl				25								1
fin					29							
fra						48						1
ita							29					1
nld								49				1
nor	1	1	1						29			1
por										8	1	
swe											26	
ELSE			1								1	42

The conversion process proceeds as follows. First, the system tokenizes the user query into words. If the query is in Japanese, JUMAN is used to split it into words.

Second, it consults bilingual dictionaries. If the query is in Japanese, then the Japanese-to-English dictionary for ALT-J/E (Ikehara et al. 1991), a Japanese-to-English translation system, is consulted. If the query is in English, it consults the English-to-Japanese dictionary converted from the former dictionary. The system searches for a dictionary entry corresponding to the longest sequence of words from left to right of the query. For example, if the input string is "information retrieval system," and if the dictionary contains "information" and "information retrieval," then the latter entry is chosen for the leftmost part of the query. Then, it searches for the entry with "system." If there are two or more translations for one dictionary entry, the module chooses the most frequently used words (or phrases) in a text corpus collected from the WWW.

Query: インターネットの情報検索

Tokens: インターネット／の／情報／検索

Translation: internet information retrieval

Boolean Expression:
(OR (AND インターネット 情報 検索)
(AND internet information retrieval))

Figure 5: An example of query formation.

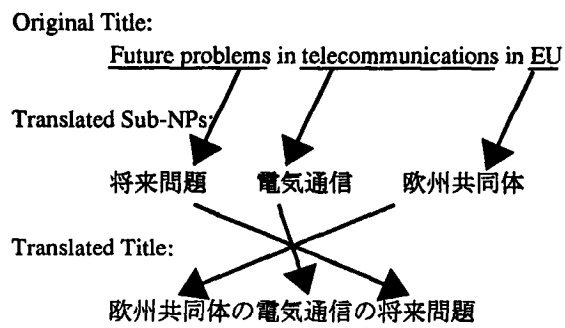
Finally, the words in the original query and the words in the translation are separately combined using "AND" operators. Then, the resulting two conjunctions are combined using an "OR" operator.

Figure 5 outlines an example of query formation.

Translating WWW Page Titles

The title translation mode is more complicated because outputs should be understandable and possibly natural for humans. Currently, only English-to-Japanese translation is supported. We regard titles as a simple noun phrase, where no relative clauses are included.

Figure 6 exemplifies a title translation process.



(の: post-positional particle in Japanese)

Figure 6: An example of title translation.

The module first consults a dictionary in the same way as the query translation, then sets a phrase boundary marker before each preposition (i.e., prepositional phrase), and finally applies transformation operations to convert head-first structures in English into head-last structures in Japanese.

Although the quality of the translation may not be very high, Japanese users can quickly judge whether or not retrieved web pages are relevant to their needs. This feature is proven useful as a navigation aid as well as other annotations attached to the search results.

Discussion

Although TITAN achieves a cross-linguistic search function between Japanese and English, it is still primitive.

First, the accuracy of the searches might not be sufficient. To improve the accuracy, the query translation process must be further improved. Although we have implemented a special routine to handle Japanese compound nouns, other techniques to reduce the translation ambiguities should be introduced. As contextual information is required for the disambiguation, application of cross-linguistic relevance feedback techniques (Oard & Dorr 1996) seems promising.

Second, the system currently only supports cross-linguistic searches between Japanese and English, while it can distinguish web pages written in other languages. To expand the set of languages which are searchable, some common representation (an *interlingua*) should be incorporated. However, as discussions in the machine translation arena show, it would be difficult to signify a fine-grained concept given in a specific language with some interlingual representation (assumed it can be developed). To solve the problem, an *iterative deepening* approach seems effective. With this approach, a user first roughly specifies topic of interests with an interlingual support provided by the system. After some set of relevant pages is retrieved, the user precisely describes his/her information need, and the system performs language-pair-dependent cross-linguistic searches.

A general issue particularly crucial in a multi-lingual cross-linguistic search system is document ranking algorithm. Currently used ranking algorithms (Harman 1992) are developed for mono-lingual contexts. We thus may need to expand the algorithms in order to handle a set of documents written in several languages. Use of interlingual concept symbols instead of words in natural languages may be a straight expansion, while its effectiveness has not been proved so far.

The final issue we'd like to mention here, particularly for the searches on the WWW, is a cross-linguistic navigation interface. It is very important to facilitates the iterative process which employs the cross-linguistic relevance feedback and/or the iterative deepening of query concepts. Cross-linguistic text summarization will be beneficial for users to screen irrelevant items out from the result lists. Use of a type of hyper-links, which we call 'cross-linguistic link', will provide a good clue in selecting a page with equivalent content in a different language. Those links are usually marked by particular linguistic expressions (e.g., "Click for French page"), and easy to extract for the internet robot.

Concluding Remarks

This paper detailed two cross-linguistic modules implemented in a working system, TITAN. These modules serves users, especially those who stick to Japanese, to explore the WWW in their native language. The system has been in experimental service since December 1995, and appreciated as one of the representative engines in Japan (Lawson 1996).

We are also experimenting on an interactive/visual navigation interface (Susaki, Hayashi, & Kikui 1996). In the trial version, a user can adjust his/her query with a GUI, by which he/her can directly manipulate document sets classified by index terms with so-called Venn-diagram. Furthermore, the items in a result list are visualized in a 3D space, in which the axes are normalized relevance score from WAIS-sf, the amount of text information, and the number of hyper-links in the pages.

References

Beesley, K. R. 1988. Language Identifier: A Computer Program for Automatic Natural Language Identification of On-line Text. In *Proceedings of the 29th Annual Conference of the American Translators Association.*

Cavnar, W.B., and Trenkle, J. M. 1994. N-gram Based Text Categorization. In *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval.*

Cheong, F. 1996. *Internet Agents: Spiders, Wanderers, Brokers, and Bots.* New Riders Publishing.

Fox, C. 1992. Lexical Analysis and Stoplists. In *Information Retrieval: data structure and algorithms.*, edited by Frakes, W.B. and Baeza-Yates, R.: Prentice-Hall.

Harman, D. 1992. Ranking Algorithms. In *Information Retrieval: data structure and algorithms.*, edited by Frakes, W.B. and Baeza-Yates, R.: Prentice-Hall.

Henrich, P. 1989. Language Identification for the Automatic Grapheme-to-Phoneme Conversion of Foreign Words in a German Text-to-Speech System. In *Proceedings of EUROSPEECH-89.*

Ikehara, S., et al. 1991. Toward and MT System without Pre-Editing-Effects of New Methods in ALT-J/E. In *Proceedings of MT Summit III.*

Kikui, G. 1996. Identifying the Coding System and Language of On-line Documents on the Internet. In *Proceedings of COLING-96.*

Lawson, S. 1996. An Introduction to Japanese Internet Search Engines. *Computing Japan*, May 1996.

Matsumoto, Y., et al. 1994. NAIST Natural Language Tools. Technical Report, NAIST-IS-TR94024. Nara Institute of Science and Technology.

Oard, D.W., and Bonnie, J.D. 1996. A Survey of Multilingual Text Retrieval. Technical Report, UMIACS-TR-96-19 CS-TR-3615. University of Maryland, College Park.

Susaki, S., Hayashi, Y., and Kikui, G. 1996. Navigation Interface in Cross-lingual WWW Search Engine, TITAN. In *Proceedings of AUUG96 and Asia Pacific World Wide Web.*