Multilingual Generation for Translation in Speech-to-Speech Dialogues and its Realization in Verbmobil

Tilman Becker and **Anne Kilger** and **Patrice Lopez** and **Peter Poller** [1]

Abstract. This paper presents the generation module of the speech-to-speech dialogue translation system Verbmobil. Spontaneous speech, large multilingual vocabulary, difficulty of the translation task, robustness and real-time constraints make the design of such a module very challenging. In order to overcome these difficulties, we have developed a system based on a general kernel and the declarativity of the knowledge sources. This fully implemented system proves the practical relevance of several techniques such as constraint-solving for the microplanning task or HPSG-to-TAG compilation for syntactic realization. In addition to the successful deployment of our module into the Verbmobil system, the kernel has been adapted to other domains and tasks.

1 Speech–to–Speech Dialogue Translation

Speech-to-speech dialogue translation systems try to exploit the recent advances in speech and natural language processing to go beyond the language barrier. Their task is the multilingual translation of spontaneous speech input in order to allow several speakers to discuss freely in their own languages. In practice the dialogues are limited to restricted domains.

In contrast to a text translation system, the processing of spontaneous speech requires extended functionalities in almost every module because the system has to be able do deal with, e.g., ill–formed and disfluent (hesitations, repetitions, repairs) speech input. These additional functionalities have a strong impact on the generation task and raise to several specific requirements.

1.1 Requirements on Generation

While the macroplanning task is important and mandatory in text generation, it is limited in dialogue translation. Most of the related problems, for instance sentence segmentation and sentence mood choice, have been solved in the source language. On the contrary, the spontaneous speech translation task leads to specific requirements:

- **Robustness**: Incomplete and non-wellformed input for the generation module can result from spoken disfluencies, from faults inside previous processing level (speech recognition module for example) or from the translation task.
- **Time constraints**: The usability of a dialogue translator depends on its acceptability with respect to speed.
- **Portability**: The generator has to cover a large vocabulary and different languages.

[1] DFKI GmbH, Stuhlsatzenhausweg 3, D-66123 Saarbrücken, email: {becker|kilger|lopez|poller}@dfki.de

ECAI 2000. *14th European Conference on Artificial Intelligence*
Edited by Werner Horn
Published in 2000 by IOS Press

Although the generation system presented in this paper tackles these constraints without being domain-dependent, they have been integrated first in the real-world speech-to-speech dialogue translation system Verbmobil.

1.2 The Verbmobil system

The Verbmobil project is a long term research effort: Started in 1993 the system is reaching completion this year. The Verbmobil System [12] aims to translate in spontaneously-spoken dialogues robustly and bidirectionally for German/English and German/Japanese. The current Verbmobil system is speaker-independent with the ability to process large vocabularies (about 10,000 words) for German and English. Verbmobil works on specific domains (negotiations about scheduling appointments, travel planning and hotel reservations) and can support different voice media from face-to-face dialogue to cellular phones.

Due to the high complexity of this task, the system is subdivided into 24 separate subtasks (implemented modules). Moreover, Verbmobil combines different approaches to machine translation. A simplified overall architecture of the Verbmobil system is presented figure 1.

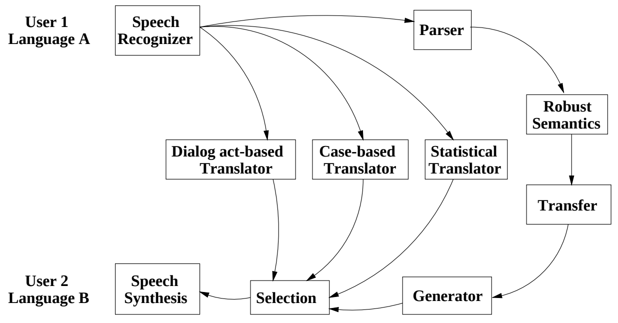


Figure 1. Simplified system architecture of the speech-to-speech translation system

The competitive translations resulting from the different parallel processing paths are partly associated with confidence values reflecting their quality and then sent to a special selection component that chooses the most appropriate one. Here the shallow translation paths serve as a fall-back in order to fulfill the strong necessity of a translation result as far as possible. In addition to the requirements presented in the previous section, our practical experience shows that there are cases in which the input to the generation component is impossible to process.

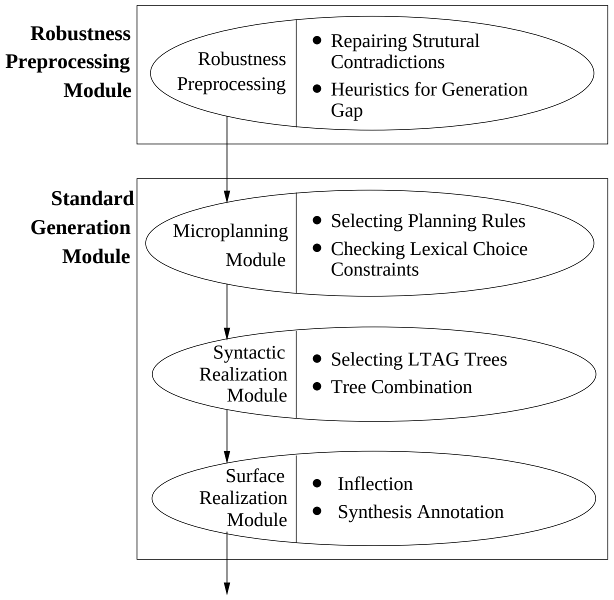
For an easy adaptation to other domains and languages, we have emphasized an organization based on a general language-independent kernel system and the declarativity of language-dependent knowledge sources [3].

2 The Kernel Architecture

2.1 The VM-GECO system

The deep processing in the Verbmobil system is based on a pipeline of modules which use a unique interface language that incorporates an interlingua semantic representation called VIT (Verbmobil Interface Term, see [5]). As a semantic representation, the VIT gathers information about *what is said / what to say*. It also includes information about *how it is said / how to say* (tense, aspect, prosody, sortal restrictions, morpho-syntax). Since discourse plays a central part for translation in dialogues, the model theoretic semantics is based on Kamp's Discourse Representation Theory (**DRT**, see [8]). Each individual indicated by some input utterance is formally represented by a **discourse referent**. Information about the individual is encoded within the **DRS–condition**, combining a predicate with the chosen discourse referent. Relations between descriptions of different discourse referents lead to a global hierarchical semantic structure.

The VerbMobil GEneration COmponent (VM-GECO) architecture consists of four main modules (see figure 2). The first one, the *robustness preprocessing module*, is dedicated solely to robustness in the specific speech-to-speech translation task and will be further discussed in the section 3.2. We present in the next sections the *standard generation modules* which aim to constitute a generic and portable kernel for real-time multilingual generation on large vocabularies.



Robustness Preprocessing Module
Robustness Preprocessing
Repairing Strutural Contradictions
Heuristics for Generation Gap
Standard Generation Module
Microplanning Module
Selecting Planning Rules
Checking Lexical Choice Constraints
Syntactic Realization Module
Selecting LTAG Trees
Tree Combination
Surface Realization Module
Inflection
Synthesis Annotation

Figure 2. The VM-GECO architecture

2.2 Microplanning

The input to the microplanner is the output of Verbmobil's semantic–based transfer. The microplanner has to map the relevant information onto a sentence plan that servers as input for the syntactic realization component. The VM–GECO sentence plan consists of lexical items (plus some syntactic information) and a specification of semantic roles connecting them. The example input to VM–GECO in Figure 3 shows a pretty–printed representation of parts of a VIT which represents the transfer result for the German input *Wann beginnt Ihr Urlaub ?* ("When does your vacation start ?"). Most relevant are the conditions slot containing the semantic predicates and the constraints slot defining the dependencies between the elements.

The microplanner's task is to plan the utterance on a *micro*– (i.e. a phrase– or sentence–) level (see, e.g. [6]). It has to decide about sentence type selection, clause conjunction and subordination into longer sentences, aggregation (elision) to remove redundancies, theme control, focus control, reference (anaphora) specification, lexical selection, etc. Obviously, neither of these decisions can be made locally because of their multidirectional dependencies. The choice of an interrogative sentence requires an (at least elliptical) verbal phrase as head of the sentence; nominalization or the choice of passive voice depend on the result of word choice, etc. Since we did not want to prefer one order of choices over others, we conceived and realized microplanning by a **constraint system** suited to represent undirected relations between variables.

The advantages of a constraint system do not only lie in the declarativity of the knowledge sources. Having defined a suitable representation of the problem to be solved, a constraint–based approach also establishes a testbed for examining the pros and cons of different evaluation methods, including backtracking, constraint propagation, heuristics for the order of the instantiation of variable values etc. Although some naive approaches towards solving constraint systems suffer from inefficiency, the current system has acceptable runtime and we expect that the accurate examination of the task will lead to the use or development of a special algorithm with even better performance.

```
vit(vitID(...),                              %Segment ID
    []),                                     %WHG-String
    index(l250,l234,i72),                    %Index
    [start_v(l248,i72),                      %Conditions
     arg1(l248,i72,i75),
     nop(l240,h85),
     quest(l249,h84),
     time(l238,i73),
     abstr_vacation(l247,i75),
     pron(l242,i74),
     poss(l244,i75,i74),
     temp_loc(l239,i72,i73),
     def(l245,i75,h87,h86),
     whq(l235,i73,h83,h82)],
    [in_g(l235,l237),                        %Constraints
     ...
     leq(l234,h85),
     ...],
    [s_class(l240,mp),                       %Sorts
     ...],
    [ana_ante(i74,[i75,i69,i67,i66]), %Discourse
     prontype(i74,third,std),
     ...],
    [gend(i75,masc), num(i75,sg)],           %Syntax
    [ta_mood(i72,ind),                       %Tense and Aspect
     ...],
    [...]                                    %Prosody
   )
```

Figure 3. An Example VIT as input to VM–GECO

In our approach the representation of the microplanning problem as a CSP (Constraint Solving Problem) defines the variables of the constraint net – simplified spoken – as the set of all **discourse referents** plus all **predicates** from the input VIT. The domains for each variables are computed online by applying the microplanning rules, which are represented as pattern–action pairs. A pattern is to be matched with part of the input, the action describes a bundle of syntactic features realizing the message part in an adequate way. According to the types of variables used, the microplanning rules are separated into the following classes:

Content rules define the mapping from (groups of) semantic predicates to syntactic features including their relations to complements (e.g. lexical choice, filling of verb frames).

Relation rules define the mapping of semantic relations (information concerning one discourse referent) to semantic/syntactic relations between lexical heads and modifiers.

Hole rules define the mapping of underspecified quantifier and operator scopes to unique syntactic relations between the respective elements.

A part of the microplanning content rules for the semantic predicate TIME is shown exemplarily in Figure 4. The first rule describes the 3:1–mapping of cooccurring predicates TIME, WHQ and TEMP_LOC to the lexical item *when* (WHEN1). The second describes a 1:1–mapping of the semantic predicate TIME to the lexical item *time* (TIME_N1) inheriting the number information from the syntax slot of the VIT. In the second case (which should be weighted as less adequate), the WHQ would be additionally mapped to *what*, the TEMP_LOC predicate to *at* and the global result would be "at what time". The microplanner's knowledge base contains about 4500 content rules and 160 relation rules for English, 7000 content rules and 170 relation rules for German.

The main constraint used for the CS is a matching algorithm which filters out combinations of variable instantiations with contradictory (syntactic or semantic) specifications. Since there may be dependencies over the whole set of variables it is defined as global constraint which unfortunately prevents us from using common approaches for binary CSPs. A second constraint filters out solutions which do not represent connected graphs.

```
;; ------- noun + whq entry
((TIME (L I) WHQ (L1 L H) TEMP_LOC (L2 I1 I))   ;pattern
 (TIME (CAT ADV) (HEAD WHEN1)))                 ;body
;; ------- noun entry
((TIME (L I))                                   ;pattern
 (TIME (CAT N) (HEAD TIME_N1) (NUM NUM(TIME)))) ;body
```

Figure 4. Example Microplanning Content Rules

Up to now, we implemented a simple backtracking mechanism for solving the CSP. Although this approach is rather primitive our system showed acceptable runtime (see Section 3.1). We expanded the algorithm by weights for the elements of the domains in order to support the microplanner in filtering out an appropriate sentence plan for a given message from all possible alternatives. All syntactic descriptions are weighted with respect to input, discourse context and dialog situation. Since there might be locally optimal mappings that lead to contradiction on a global level, the microplanner should generally use these weights to direct the process of backtracking or propagation.

There is a mechanism for constraint systems that helps meeting the requirement of robustness: the **constraint hierarchy**. Working with strengths of constraints enables the designer of the microplanning knowledge base to formulate which constraint must and which should be fulfilled for realizing an adequate utterance. Furthermore, variable levels of both *correctness* and *acceptability* can be encoded so that they can be adapted on–line to the needs of the given situation (e.g. reflecting time pressure).

Currently, we are testing several constraint propagation and backtracking mechanisms for their suitability with respect to the microplanning task. Although we have not yet completed the design of the microplanner, we have gained valuable experience with

- different representations of the problem and their advantages and disadvantages for the processes of mapping and constraint solving,
- possible influences of the weighting of alternatives on the instantiation of variables,
- the usage of constraint hierarchies for the sake of robustness, e.g. by allowing for *minor semantic and/or syntactic contradictions* in the output, and
- the suitability of incremental constraint solving techniques for the microplanning task,

which will fasten our progress in developing constraint–based microplanning.

2.3 Syntactic realization

The syntactic realization module generates a string (richly annotated for a concept-to-speech synthesis) based on the sentence plan that is generated by the microplanner. At this point, word choice is almost complete. However, the sentence plan does not include auxiliaries which are added in a first preprocessing phase.

2.3.1 Precompilation of HPSG resources

A lexicalized formalism is best suited to work on such structures and we chose to take lexicalization in the existing grammar and lexicon resources, namely HPSG grammars [10], one step further by compiling them into a tree-adjoining grammar (TAG), [7], extending the algorithm described in [9].

The main motivation for precompilation are:

- An HPSG grammar was already developed within the Verbmobil project.
- The sharing of resources is facilitated by the fact that HPSG and TAG are closely related formalisms, both characterized by strong lexicalization. In both formalisms, the grammar is separated from the lexicon. It is based on a finite set of *lexical types* and the lexicon maps entries to their lexical type, stem, semantic predicate, etc.
- The TAG formalism is better suited for generation: The compilation allows to identify larger substructures of the HPSG grammar that are not context-dependent. Using the extracted partial trees results in gains in efficiency compared to run-time construction of these trees.
- The syntactic realization only needs to check syntactic constraints, the HPSG grammar contains a lot of additional features which are not used for this task.

The size of the resulting TAG grammar (2,350 trees for 359 lexical types) appears to be a potential drawback of this approach. However, there are several stages of filtering during syntactic generation which reduce the search space drastically. The first step is the selection of the appropriate subgrammar for each sentence to be generated: Since the TAG/HPSG grammar is lexicalized, only those trees are selected that are anchored by one of the current lexical types. This reduces the subgrammar to an average of around 10 trees per word.

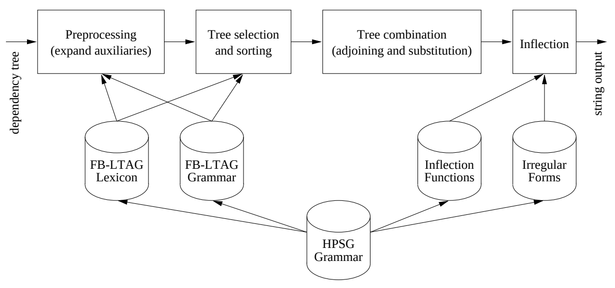


Figure 5. The main steps of syntactic realization and the knowledge soures used.

2.3.2 On-line processing

The main phases of the syntactic realizer are shown in figure 5.

As noted above, a preprocessing phase maps from the abstract sentence plan to the concrete syntactic dependencies as specified by the HPSG grammar and thus by the TAG grammar compiled from it.

The next phase, *tree selection* is the above-mentioned selection of a subgrammar. A lexicon lookup for every word in the sentence plan returns the set of all precompiled partial phrase structure trees for its lexical type (i.e., the elementary trees of the TAG grammar). The chosen words in the sentence plan carry additional information (such as the sentence type, person, number, and gender) which is unified into the elementary trees. This represents a second filter and further reduces the set of applicable elementary trees.

The core *combination phase* is implemented as a guided search. Since the elementary trees already represent maximal partial structures, this phase consists only of the combination (with the adjunction and substitution operations of TAG) of elementary trees. No HPSG schemata (corresponding to context-free rules) need to be applied at run-time. Further filtering takes place during tree combination, since some adjunctions and substitutions fail due to incompatible feature information.

A final *inflection phase* uses the stems and the feature structures for an inflectional morphology component. This phase is based completely on the structures and tools taken from the HPSG grammar. The resulting string is then annotated with the relevant syntactic information for the concept-to-speech synthesis module.

2.3.3 An Example

Figure 6 shows some of the trees used in the syntactic generation of the phrase *“Thursday the ninth of July”*.

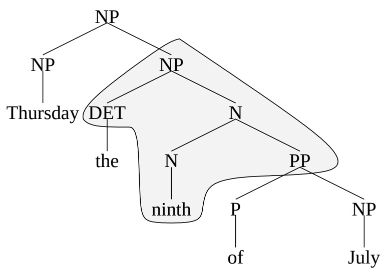


Figure 6. Resulting derived tree for *“Thursday the ninth of July”*

Some of the trees available for the word *“ninth”* are shown in figure 7.

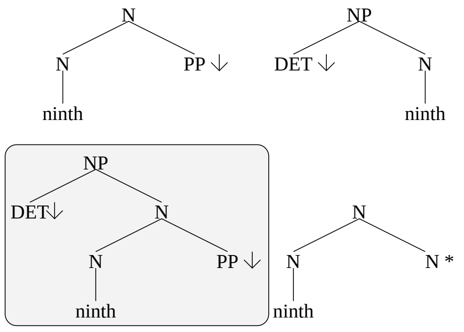


Figure 7. Available trees for the word *“ninth”*

The sentence plan is drawn as a tree in figure 8.

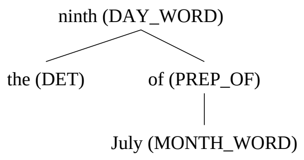


Figure 8. The sentence plan for the example sentence.

3 Implementation and Applications

In this section we switch to practical considerations concerning facts, specific problems and extended functionalities of the implementation of our generation module inside the Verbmobil system.

3.1 Results

In order to give an impression of the implementation, we present here some characteristic numbers about it. The vocabulary of Verbmobil consists of about 10,000 words for English and German as well as about 3,000 words for Japanese which are all covered appropriately by the knowledge sources of the three generators (lexicon, microplanning rules, lexicalization, inflection). Currently, the number of VIT’s that follow the current VIT definition is about 60,000 for all three languages.

The runtime of our generators is limited externally and indirectly by a real–time factor of 4 which is predefined for the overall system runtime. Dynamically depending on the remaining runtime that other modules left, the generation module fits very well into the given time limit for typical sentences (an average runtime of 0.5 sec for a testsuite of 1000 inputs with size between 1 and 45 semantic predicates). In order to avoid time consuming non–productive inevitable timeouts, we additionally implemented heuristics to predict the estimated generator runtime (depending on the input size) which are used to stop the generation of input structures whose predicted runtime is out of the remaining time limit. In such cases, the translation is automatically left to the flat translation modules (see Section 2.1).

3.2 Robustness

Due to the large number of 24 different modules closely cooperating inside the Verbmobil system, it turned out in practice that robustness is a very important constraint inside the individual modules themselves as well as in general to achieve robust speech–to–speech translation in Verbmobil.

To address this issue, we extended the generation module by a robustness preprocessing submodule [4] whose task is the identification (and correction, if possible) of problematic input according to known sets of task inherent and technical problems. Task inherent problems manifest themselves as faults wrt. the interface language definition while technical problems mostly concern mismatches between semantic expressions and the coverage of the generator grammar.

3.3 Dialogue Scripts and Result Summaries

The flexibility of our generator as well as the numerous different modules of Verbmobil have been used to implement an additional system functionality of Verbmobil itself. It is the automatic and multilingual generation of dialogue scripts [1] and result summaries [2] of the dialogues which could be relatively easily achieved by extending some of the system module's functionalities appropriately.

The "Dialogue Module"[2] is responsible for the extraction and collection of dialogue data that are relevant for the contents of dialogue scripts and result summaries, respectively. On the other hand, the generation module is responsible for distributing and enriching these data to semantic representations of meaningful sentences or paragraphs for the desired document. Then, the multilingual generation of these representations can easily be achieved by using the system's language–specific generators as well as the system's Transfer component.

3.4 UNL

The adaptability of our kernel generation component [3] to a different application that uses a very different semantic representation formalism was proven inside the multilingual UNL[3] project (http://unl.ias.unu.edu). UNL is a language–independent semantic representation of textual documents aimed to overcome the international language barrier. Currently, software tools (for generation and analysis) for 12 different languages are under development based on large corpora of, e.g., international law texts and soccer game reports. Our part in this project is the realization and implementation of the German generator.

Due to our language–independent core generator the adaptation of the generation component to the UNL formalism decreased to the adaptation of the language–independent parts, namely the structural and lexical knowledge bases of the microplanning component and appropriate domain–specific extensions on the lexicon of the syntactic generator. The average runtime per sentence on a SUN ULTRA–2 is less than 0.5 seconds based on a testsuite of about 500 sentences and an average sentence length of 15 words.

[2] The main task of the "Dialogue Module" is to model the user's dialogue behavior including the arrangement of a dialogue memory in order to support contextual translation problems like, e.g., anaphora resolution.

[3] UNL is the acronym for *Universal Networking Language* [11]. It is an interlingua–based approach to semantics.

4 Conclusions and Current Work

In this paper we have shown a portable multilingual generation component that is used for translation in speech–to–speech dialogues. The dialogue scenario in Verbmobil has the advantage that the generation task begins with microplanning while all macroplanning decisions are implicitly made by the dialogue partners themselves. On the other hand, spontaneously spoken speech input requires additional effort for all system modules to deal with ill–formed (e.g., ungrammatical or phrasal) input data. Additionally the numerous intermodular interfaces sometimes produce problematic data which cannot be processed further without corrections. Both of these problems are handled in our special robustness preprocessing module.

Our current work mainly consists merely in tuning our implementation for all three languages English, German and Japanese. The speech–to–speech translation system Verbmobil has already reached its full functionality. It has already been presented successfully; the presentations continuously include major AI conferences such as, e.g., COLING, ECAI, ACL.

REFERENCES

[1] J. Alexandersson and P. Poller, 'Towards multilingual protocol generation for spontaneous speech dialogues', in *Proceedings of INLG-98*, Niagara-On-The-Lake, Ontario, Canada, (1998).

[2] J. Alexandersson, P. Poller, M. Kipp, and R. Engel, 'Multilingual summary generation in a speech–to–speech translation system for multilingual dialogues', in *Proceedings of INLG-2000*, Mitzpe Ramon, Israel, (2000).

[3] T. Becker, W. Finkler, A. Kilger, and P. Poller, 'An efficient kernel for multilingual generation in speech–to–speech dialogue translation', in *Proceedings of COLING/ACL-98*, Montreal, Quebec, Canada, (1998).

[4] T. Becker, A. Kilger, P. Lopez, and P. Poller, 'An extended architecture for robust generation', in *Proceedings of INLG-2000*, Mitzpe Ramon, Israel, (2000).

[5] M. Dorna, 'The ADT–Package for the Verbmobil Interface Term', Verbmobil Report 104, IMS, Universität Stuttgart, Germany, (1996).

[6] E. Hovy, 'An overview of automated natural language generation', in *Proceedings of the International Symposium on Natural Language Generation and the Processing of the Chinese Language, INP(C)–96*, ed., X. Huang, pp. 15–31, Shanghai, China, (1996).

[7] Aravind K. Joshi, 'An introduction to Tree Adjoining Grammars', in *Mathematics of Language*, ed., A. Manaster-Ramer, John Benjamins, Amsterdam, (1987).

[8] H. Kamp and U. Reyle, *From Discourse to Logic*, volume 42 of *Studies in Linguistics and Philosophy*, Kluwer Academic Publishers, Dordrecht, 1993.

[9] R. Kasper, B. Kiefer, K. Netter, and K. Vijay-Shanker, 'Compilation of hpsg to tag', in *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pp. 92–99, Cambridge, Mass., (1995).

[10] Carl Pollard and Ivan A. Sag, *Head-Driven Phrase Structure Grammar*, Studies in Contemporary Linguistics, University of Chicago Press, Chicago, 1994.

[11] H. Uchida and M. Zhu, 'An interlingua for multiligual machine translation', Technical Report 89–NL–72–9, Information Processing Society of Japan, (1989).

[12] *Verbmobil: Foundations of Speech-to-Speech Translation*, ed., W. Wahlster, Springer, 2000.