Achieving Robustness through the Integration of Language Production in Comprehension

Paul Van Eecke (vaneecke@csl.sony.fr)
Sony Computer Science Laboratory Paris, 6, Rue Amyot
75005 Paris, France

Abstract

Language comprehension in humans is very robust. With apparent ease, language users can grasp the meaning of utterances made in very bad acoustic conditions or containing grammatical errors. A key mechanism in achieving this robustness is the ability of the listener to predict what the speaker is likely to say. According to recent work in psycholinguistics, these predictions are made by covertly simulating the speaker while he is speaking and thus integrating production in comprehension. This article presents a model which makes use of this idea in order to enhance the robustness of language processing in intelligent systems, in particular to recover the intended meaning of utterances containing grammatical errors. The model is implemented in Fluid Construction Grammar and applied to a case study for the complex Dutch Verb Phrase.

Keywords: Language Processing; Comprehension; Production; Robustness; Construction Grammar

Introduction

Language production and comprehension have traditionally been studied as two separate, independent cognitive processes (Clifton, Meyer, Wurm, & Treiman, 2012). Following the broader movement in cognitive science to study the interaction between perception and action processes (for example in eye-hand coordination), researchers have more recently started to focus on the interplay between comprehension (as a form of perception) and production (as a form of action). Pickering and Garrod (2007) indicate that the trend in cognitive science of integrating perception and action was at first asymmetrical. The hypothesis that perception influences the guiding of action has been widely accepted for a long time (Lee, 1976), whereas the hypothesis that action contributes to perception has been much less studied. Evidence in favour of this claim has only been provided quite recently, for example by Wilson and Knoblich (2005), who argue that imitative motor activation is used in the perception of conspecifics. More specifically, they state that perceived human movement triggers ‘emulation’, an internal simulation of the perceived motion. This emulation generates expectations and predictions about the unfolding action and in this manner contributes to perception.

The same asymmetry as in cognitive science can be observed in psycholinguistics. The hypothesis that comprehension is important in monitoring production has been studied and confirmed by many researchers (Levelt, 1983; Levelt, Roelofs, & Meyer, 1999; Postma, 2000), but the hypothesis that production is used to assist comprehension has only recently gained attention. Pickering and Garrod (2007) make a strong case for the use of production in comprehension and Pickering and Garrod (2013) present an integrated theory of language production and comprehension, in which they assign, like Wilson and Knoblich (2005), a crucial role to expectations and predictions. In this theory, listeners continuously emulate the speakers. Based on their own potential utterances, they use forward models to predict what the speakers are likely to say.

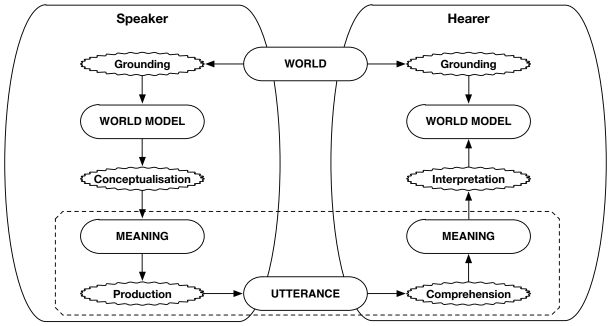


Figure 1: The semiotic cycle

In this article, we employ the idea of using production to facilitate comprehension in order to enhance the robustness of language processing in intelligent systems. More specifically, we present a model in which the listener makes use of production to make hypotheses about the intended meaning of the speaker, if he cannot comprehend the speaker’s utterance directly. In particular, we focus on how the listener can recover the intended meaning of utterances containing grammatical errors. The fully operational model is implemented in Fluid Construction Grammar (Steels, 2011) and applied to a case study for the complex Dutch Verb Phrase.

Comprehension and production in FCG

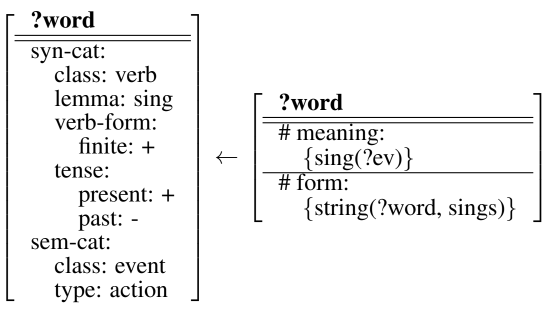
In order to communicate, intelligent systems need to go through the different stages of the *semiotic cycle*, shown in figure 1 (Steels & van Trijp, 2011). These stages include *grounding* (the perceptual linking of the real world to the internal world model), *interpretation and conceptualisation* (the linking of the internal world model to what is said or to what needs to be said) and *comprehension and production* (the linking of an utterance to its meaning). Although the semiotic cycle presents the different steps that are involved in communication in a modular way, they are tightly interwoven. It is clear, for example, that the comprehension process receives continuous feedback from the interpretation and grounding processes.

Fluid Construction Grammar (FCG) was originally developed to fulfill the task of the comprehension and production components of the semiotic cycle (marked by the dashed box in figure 1) in language evolution experiments. Using a grammar, FCG can map an utterance to its meaning representation (comprehension) and map a meaning representation to an utterance (production). Before explaining how we implemented the integration of production in comprehension, we need to have a closer look at how FCG comprehends and produces utterances, as well as what a typical FCG grammar looks like.

The Basic Mechanisms

An FCG grammar is composed of *constructions*, which are couplings of meaning and form. An FCG construction, as illustrated in (1), consists of a left-hand side, the *contributing part*, and a right-hand side, the *conditional part*. The contributing and conditional part are connected by an arrow. The conditional part, in its turn, consists of a production lock and a comprehension lock, divided by a horizontal line. In comprehension, the construction looks whether it can find the requirements stated in the comprehension lock in the *transient structure* (i.e. the input, potentially enhanced by units and features added through the previous application of constructions). If so, it adds the features from the production lock and the contributing part to the transient structure. Then, other constructions can apply. In production, the construction searches for the requirements stated in the production lock and adds the features from the comprehension lock and contributing part to the transient structure. For example, the construction presented in (1) searches in comprehension for a string ‘sings’. If this is found, it makes a unit around this string and adds the meaning ‘sing(?ev)’ to this unit, as well as its syntactic and semantic properties stated in the contributing part. In production, this construction looks for the meaning ‘sing(?ev)’, builds a unit around it and adds the string and its semantic and syntactic properties to this unit.

(1)



FCG is an entirely bidirectional grammar formalism in the sense that it uses the same grammar and processing mechanisms for both comprehension and production. The only difference resides in which units and features are included in the comprehension and production locks or in other terms, in which units and features should be present for the construction to apply.

A Grammar for the Dutch Verb Phrase

As a case study for the model presented in this paper, we use a grammar for the Dutch Verb Phrase (VP). The Dutch VP forms a very interesting case study because of its high complexity. In the next paragraphs, we will go through those aspects of the grammar that are needed to explain our model. A more elaborate discussion of the grammar itself can be found in (Van Eecke, to appear).

Complexity of the Dutch VP The Dutch VP is a very complex structure. Its complexity is mainly due to three of its syntactic properties. First, Dutch VPs allow *modal stacking*, i.e. the use of more than one modal auxiliary inside the same VP. A simple example of modal stacking is shown in (2).

(2) Hij moet kunnen komen .
`He must can come .`
He must be able to come.

In this utterance, the lexical verb ‘komen’ (*to come*) is modified by a sequence of two modal auxiliaries, ‘moet’ (*must*) and ‘kunnen’ (*can*). Each auxiliary applies its modal meaning to the part of the VP that it precedes. In this case, ‘kunnen’ expresses the ability of the subject to come and ‘moet’ expresses the necessity of this ability. The whole utterance expresses that it is necessary that the subject is able to come. Modal stacking is quite common in Dutch and is not limited to a maximal number of auxiliaries.

The second source of syntactic complexity for the Dutch VP is related to the expression of the perfect aspect. In Dutch, the perfect aspect is expressed by adding a perfect auxiliary (either ‘zijn’ or ‘hebben’, depending on the lexical verb) to the VP. The perfect aspect can not only be expressed on the main verb, but also on any constituent introduced by a modal auxiliary. This allows the speaker to carefully set the scope of the perfect. This is illustrated in (3a) to (3c).

(3) a. Hij zal moeten hebben gesprongen.
`He will must have jumped .`
He will have to have jumped.

b. Hij zal hebben moeten springen .
`He will have must jump .`
He will have had to jump.

c. Hij heeft moeten kunnen antwoorden.
`He has must can answer .`
He has had to be able to answer.

In (3a), the perfect aspect is expressed on the verb ‘gesprongen’. Because this is a lexical verb, it is realised as a past participle under the influence of the perfect auxiliary. In (3b) and (3c), the perfect aspect is expressed on the constituents ‘moeten springen’ and ‘moeten kunnen antwoorden’ respectively. As these constituents are introduced by a modal auxiliary, all verb forms are realised as infinitives.

The third syntactic property that makes the Dutch VP difficult to model is the possible variation in word order for perfect VPs. Within a set of constraints, the perfect auxiliary, the modal auxiliaries and the lexical verb can appear at different positions in the utterance. For the sentence introduced in (3a) for example, all word orders presented in (4) are acceptable, affecting the meaning only very slightly.

(4) a. Hij zal moeten hebben gesprongen.
b. Hij zal moeten gesprongen hebben.
c. Hij zal gesprongen moeten hebben.

Meaning Representation A very important aspect of every FCG grammar is the way it represents meaning. FCG does not impose any kind of meaning representation, so its design is completely up to the grammar designer. The semantic representation used in our grammar for the Dutch VP consists of a set of meaning predicates. Each predicate expresses very specific semantic information and at least one of its arguments is a variable. By linking the variables of different meaning predicates to each other, the grammar builds semantic networks which can express the meaning of a complex VP.

The core meaning component of a VP is the *lexical meaning* of the main verb. Lexical meanings are represented using a single predicate consisting of two elements: the predicate name itself, which is an English form expressing the lexical meaning, and a referent for which the predicate holds. An example of the meaning representation for the verb 'zingen' (*to sing*) is shown in (5). It can be read as 'there is a singing event'. Conventionally, variables are preceded by a question mark and a prefix notation is used for predicates.

(5) `(sing ?ev)`

A second important component of the meaning of VPs is the expression of a specific *modality*. The kind of meaning predicate that we use to represent modality is slightly different compared to the one for lexical meaning. A modal meaning predicate takes the event provided as third argument as input, adds the modal interpretation provided as fourth element, and returns the event with its modal interpretation as the second argument. The input argument can be bound to any event, whether it is introduced by a meaning predicate for a lexical verb, for another modal expression or for a perfect expression. As an illustration, the meaning predicates for volitive and permissive modality are shown in (6).

(6) a. `(modality ?super-ev ?ev volitivity)`
b. `(modality ?super-ev ?ev permission)`

The third meaning component covered in our grammar is the *perfect aspect*. The structure of the predicate representing this meaning component is very similar to the one for modality. The predicate takes the event provided as third element as input, adds the semantic properties of the perfect aspect and returns the result as the second element. This predicate is shown in (7).

(7) `(perfect ?super-ev ?ev)`

The fourth kind of semantic information conveyed by the VP is *tense*, the location of an event in time. In our grammar, we distinguish between two temporal categories: past and present. This binary distinction (although frequently referred to as past vs. non-past) is very common for Dutch (De Jonghe & De Geest, 1990), as well as for English (Huddleston, 1995) and many other Indo-European languages (Comrie, 1985). Future in Dutch is not expressed as a tense, but rather as a modal distinction. Formally, we represent past and present tense as a precedence or overlapping relation with reference to the deictic time point. Two predicates are used for this: one to introduce the deictic time point itself, and another to define the relation between this deictic time point and the event. These predicates are shown in (8a) for the present and in (8b) for the past.

(8) a. `(deictic-time-point ?origo)`
`(overlaps ?ev ?origo)`
b. `(deictic-time-point ?origo)`
`(before ?ev ?origo)`

The last meaning distinction made by our grammar is related to the meaning components that are profiled in an utterance. In perfect VPs, a speaker can focus on the event itself or rather on the modal interpretation of the event. Syntactically, this semantic difference is realised as a word order difference. Focusing on an event causes the past participle of the main verb to be positioned immediately after the finite verb, instead of later in the verb phrase. Formally, this part of the meaning is represented by adding one of the meaning predicates given in (9).

(9) a. `(action-focus ?ev +)`
b. `(action-focus ?ev -)`

The meaning predicates introduced above can be freely combined and integrated into a semantic network by linking their variables. As an example, figure 2 presents the semantic network for the VP 'moet hebben kunnen zingen' (*must have been able to sing*). The network clearly shows how the different meaning components are interlinked and compose the complex meaning of the utterance.

Grammar The task of the grammar is to map between Dutch VPs and their corresponding semantic networks. The constructions of which the grammar is composed can be divided into two groups, depending on their level of abstraction. The first group consists of *morphological and lexical constructions*. Morphological constructions map between strings and their morpho-syntactic properties. In comprehension, they look for strings in the input, build units around these strings and add morpho-syntactic features to these units. In production, they look for units containing certain morpho-syntactic features and add the corresponding strings to the output. Lexical constructions perform a similar task for meaning predicates. In comprehension, they look for units with specific syntactic properties and add the corresponding

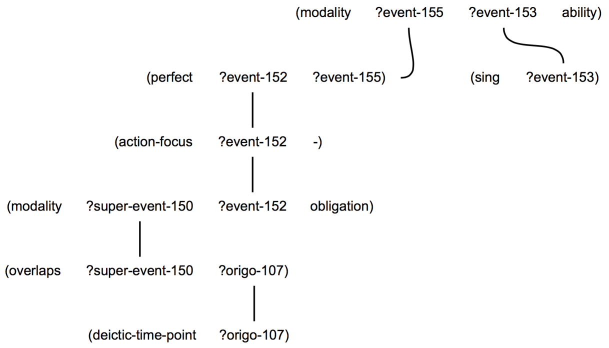

(modality ?event-155 ?event-153 ability)
(perfect ?event-152 ?event-155)
(sing ?event-153)
(action-focus ?event-152 -)
(modality ?super-event-150 ?event-152 obligation)
(overlaps ?super-event-150 ?origo-107)
(deictic-time-point ?origo-107)

Figure 2: Semantic network for the VP 'moet hebben kunnen zingen' (*must have been able to sing*).

meaning predicates to the output. In production, they look for meaning predicates in the input, build units around them, and add their semantic and syntactic properties.

The second group of constructions contains the *grammatical constructions*. These constructions look for units containing specific syntactic and semantic features. They add new features to these units, create new units and build hierarchical structures. In comprehension, these constructions use the variable bindings from the input semantic network for producing an utterance that does not only contain the correct strings, but also expresses the correct meaning. In production, these constructions ultimately bind the variables in the predicates that were introduced by the lexical constructions in a correct, meaningful way.

Integrating Production in Comprehension

The FCG-grammar described in the previous section allows to comprehend and produce Dutch VPs that adhere to the set of principles that it models. But this is often not sufficient in real-world settings, as language users constantly encounter utterances which deviate from a fixed set of rules and principles, for example due to language evolution or learner errors. Therefore, we want to make our grammar robust enough to deal with erroneous input, in particular grammatical errors. As the set of possible errors is open-ended, it is impossible to model them in the grammar itself. So we developed a method which uses the integration of production in comprehension to achieve this robustness.

The use of comprehension during production is a form of *re-entrance* (Steels, 2003), a concept that has played an important role in many experiments on language learning (for examples, see (Steels, 2012)). In these experiments, the learner internally reproduces the production process of the speaker and then adapts his construction inventory to make it coherent with what was produced by the speaker. Our robust comprehension method implements this concept for enhancing robustness.

The method proceeds as follows. First, the system tries to comprehend the input sentence using FCG's standard comprehension algorithm. Then, it evaluates the resulting structure. If the resulting structure is found to be satisfactory according to a selected set of criteria (e.g. a connected hierarchical structure or a semantic network in accordance with the listener's world model), the resulting semantic network is returned. If a satisfactory solution could not be found, the system proceeds to parse as many meaning predicates as possible from the input, regardless of variable bindings. It does this by applying the morphological and lexical constructions to all strings found in the input. This way, all predicates for lexical verbs, perfect aspect and modality are collected. Then, the system tries to determine the tense of the input utterance. In order to do this, it scans from left to right through the input strings, applying the morphological constructions to them. If a finite form is found, it determines its tense and collects the corresponding meaning predicates. If the erroneous VP contains more than one finite form, the tense of the leftmost form is adopted. If no finite form is found, the present tense is adopted by default. At this point, all relevant meaning predicates have been recovered. However, as their variables are not yet correctly bound, they are not yet organised in a meaningful semantic network.

Then, the integration of production in parsing comes into play. First, all variables in the collected meaning predicates are uniquely renamed, which ensures that there are no unwanted bindings due to duplicate variable names. Then, the model emulates the speaker. It uses the set of recovered meaning predicates as a basis to produce utterances. During this process, the variables are correctly bound by the grammatical constructions and a number of possible utterances are returned, together with their corresponding semantic networks. These hypotheses are then ranked according to a chosen metric and the system returns both the form and semantic network of the highest ranked reconstruction.

Many different strategies for ranking hypotheses can be implemented and their success rate highly depends on the goals. In a setting with autonomous robots for example, it seems a good strategy to compare the relevance of the different reconstructed meanings within the context of the perceived situation. In cases where no contextual information is available, the ranking strategies may focus on the differences and similarities between the reconstructed form and the erroneous form.

Algorithm 1 shows the robust comprehension algorithm in a more general way. The *Comprehend(utterance,grammar)* function stands for FCG's standard comprehension algorithm and returns a structure containing a semantic network. The *Satisfactory(solution,evaluation)* function checks whether this solution is valid with respect to the specified evaluation criteria. The process of collecting the different meaning predicates is performed by de *CollectPred(utterance,grammar)* function. This function is very specific to each (type of) grammar. Then, all structures that contain these predicates and are allowed by the grammar are produced by FCG's *ProduceAll(predicates,grammar)* function. Finally these hypotheses are ranked according to the chosen strategy and the

```
input: utterance                          // An utterance
       grammar                            // A grammar
       strategy                           // A ranking strategy
       evaluation                         // Evaluation criteria
solution $\leftarrow$ Comprehend(utterance,grammar);
if Satisfactory(solution,evaluation) then
    return solution;
else
    predicates $\leftarrow$ CollectPred(utterance,grammar);
    predicates $\leftarrow$ RenameVars(predicates,grammar);
    hypotheses $\leftarrow$ ProduceAll(predicates,grammar);
    hypotheses $\leftarrow$ Rank(hypotheses,strategy);
    return HighestRanked(hypotheses);
end
```

Algorithm 1: Robust comprehension algorithm, using the integration of production in comprehension

highest ranked hypothesis is returned as a solution.

A First Evaluation

Although an extensive evaluation of the robust comprehension method on real-world data is outside the scope of this paper, we have conducted a pilot study which confirms that the method based on integrating production in comprehension can indeed recover the intended meaning of many erroneous forms. As a test set, we compiled a corpus of 20 semantic networks that can be expressed by the grammar. The corpus contains semantic networks of different complexity, ranging from 4 to 9 meaning predicates. For each semantic network, 4 VPs were generated: one correct VP according to the grammar, one VP containing a morphological error (e.g. a finite form or infinitive instead of a past participle), one VP containing a word order error (e.g. the finite verb in final position) and one VP containing both a morphological error and a word order error. All errors were introduced in a random fashion. The only constraint on the errors was that the word order of infinitives would not change, if this leads to a different valid parse result. This is because it is systematically impossible, even for humans, to recover from these errors without reference to the context. Note that for certain other errors which do appear in the corpus, it is very difficult, but not systematically impossible, to recover from for humans as well, such as forms not containing any finite verb.

For our purpose, we implemented three different strategies for ranking hypotheses about the intended meaning. As we do not have access to any contextual information, all three hypotheses are based on formal cues. The first strategy is based on *word order*. It assumes that the erroneous utterance contains the smallest number of word order errors possible. This strategy is expected to be especially well suited to comprehend VPs containing morphological errors. The second strategy is based on the *number of erroneous forms* in the input. It assumes that the erroneous utterance contains the smallest number of errors possible. This strategy is expected to perform well on utterances containing word order errors. The third strategy minimises the *Levenshtein distance* between the hypothesis and the perceived form. It calculates the minimal number of word-level edits that separate the hypothesis and the perceived form, according to equation (10). In this equation, i and j correspond to the number of words in the VPs a and b respectively and $1_{(a_i \neq b_j)}$ is an indicator function equal to 1 if $a_i \neq b_j$ and 0 otherwise. As the Levenshtein distance is influenced by both word order and erroneous forms, this strategy is expected to yield good results on VPs containing both kinds of errors.

(10)

$$D_{a,b}(i,j) = \begin{cases} max(i,j) & \text{if } min(i,j) = 0, \\ min \begin{cases} D_{a,b}(i-1,j)+1 \\ D_{a,b}(i,j-1)+1 \\ D_{a,b}(i-1,j-1)+1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases}$$

All 80 VPs were analysed 4 times, once with the standard comprehension method of FCG, and three times with our robust comprehension method using the three different strategies. Then, we verified whether the reconstructed semantic networks were equivalent to the original semantic networks. The results are shown in table 1. *Correct* stands for the correct VPs, *Morph* and *WO* for those containing morphological and word-order errors respectively and *Morph + WO* for those containing both kinds of errors.

The results show that with all four comprehension methods and strategies, all VPs that did not contain any errors could be correctly comprehended. While performing perfectly on correct VPs, FCG's standard comprehension method could not recover the meaning of any of the erroneous VPs. The robust parsing method using the strategy based on word order performs, as predicted, well on VPs containing morphological errors, with a score of 16 out of 20 recovered meanings. Moreover, it could recover the meaning of a small portion of VPs containing word order errors (6) and both morphological and word order errors (5). The number-of-errors strategy performs better than the word-order strategy on VPs containing word-order errors (11) and performs worse on VPs containing morphological errors (7). Its better score on the VPs containing both kinds of errors (8) makes it perform comparably overall (46 vs. 47). Finally, the strategy based on Levenshtein distance performs as well as the word-order strategy on morphological errors (16), slightly better than the number-of-errors strategy on word-order errors (13) and considerably better on the VPs containing both kinds of errors (13). Consequently, its overall result of 62 is considerably better than the overall result of the other strategies (47 and 46) and improves greatly upon the baseline of 20.

Further Research and Applications

The model described in this paper opens many perspectives for further research. A first step would be an extensive validation of our robust parsing method on real-world data, including an elaborate error analysis and comparison to human performance. Real-world evaluation can be performed in intelligent cognitive systems (for example in robotic settings) in which meaning-based ranking strategies can be used.

Table 1: Parsing results for the different kinds of errors and the different strategies.

	Correct	Morph	WO	Morph + WO	Overall
Standard	20	0	0	0	20
Robust - Word Order	20	16	6	5	47
Robust - Number of Errors	20	7	11	8	46
Robust - Levenshtein	20	16	13	13	62

Besides, it would be interesting to test how the robust comprehension method performs on different kinds of grammars. The method described above works especially well for grammars which introduce most meaning predicates through lexical constructions and bind variables in the other constructions. For non-lexical constructions that introduce meaning predicates, additional mechanisms have to be implemented to decide whether these predicates should be added in production. In our example, this is the case only for the tense predicates. Finally, we would like to investigate new ways to recruit meaning predicates for production. This could be done by determining during production which predicates need to be added in order to build a semantic network which is in accordance with the agent' s world model.

The presented model and its methods can be used in various projects, ranging from enhancing comprehension robustness in human-computer interfaces to automatically correcting learner's errors in intelligent tutoring systems. The richer and more complex the environment is, the more information can be used for making predictions and ranking hypotheses and the more the use of production will improve the quality of comprehension.

Conclusion

Recent studies in psycholinguistics and cognitive science have put forward that the integration of production in comprehension is an important mechanism contributing to the robustness of human language processing. In this article, we have presented a fully operational model which makes use of this concept in order to improve the robustness of comprehension in artificial cognitive systems. The model can comprehend utterances that are not covered by its grammar by using production to make hypotheses about their intended meaning. These hypotheses are then ranked according to semantic or syntactic criteria. We implemented the model in Fluid Construction grammar and applied it to a grammar for the Dutch VP. A first evaluation confirms that the model integrating production in comprehension greatly improves upon FCG's standard comprehension algorithm and can indeed in many cases recover the intended meaning of utterances containing grammatical errors.

Acknowledgments

This research was conducted at Sony CSL Paris and funded by the Marie Curie Initial Training Network Essence (grant agreement no. 607062). I especially thank Remi van Trijp, Luc Steels and Miquel Cornudella for their valuable comments and support.

References

Clifton, C., Meyer, A., Wurm, L., & Treiman, R. (2012). Language comprehension and production. In *Comprehensive handbook of psychology, volume 4: Experimental psychology, 2nd edition.* New York: John Wiley & Sons.

Comrie, B. (1985). *Tense.* Cambridge University Press.

De Jonghe, H., & De Geest, W. (1990). *Nederlands, je taal.* Van In.

Huddleston, R. (1995). The case against a future tense in English. *Studies in Language*, *19*(2).

Lee, D. (1976). A theory of visual control of braking based on information about time-to-collision. *Perception*, *5*(4), 437–459.

Levelt, W. J. (1983). Monitoring and self-repair in speech. *Cognition*, *14*(1), 41–104.

Levelt, W. J., Roelofs, A., & Meyer, A. S. (1999). A theory of lexical access in speech production. *Behavioral and brain sciences*, *22*(01), 1–38.

Pickering, M., & Garrod, S. (2007). Do people use language production to make predictions during comprehension? *Trends in cognitive sciences*, *11*(3), 105–110.

Pickering, M., & Garrod, S. (2013). An integrated theory of language production and comprehension. *Behavioral and Brain Sciences*, *36*(04), 329–347.

Postma, A. (2000). Detection of errors during speech production: A review of speech monitoring models. *Cognition*, *77*(2), 97–132.

Steels, L. (2003). Language-reentrance and the 'inner voice'. *Journal of Consciousness Studies*, *10*(4-5), 173-185.

Steels, L. (2011). *Design patterns in fluid construction grammar.* John Benjamins Publishing.

Steels, L. (2012). *Experiments in cultural language evolution.* Amsterdam: John Benjamins Publishing.

Steels, L., & van Trijp, R. (2011). How to make construction grammars fluid and robust. In L. Steels (Ed.), *Design patterns in fluid construction grammar* (pp. 301–330). Amsterdam: John Benjamins.

Van Eecke, P. (to appear). Robust processing of Dutch verb phrases.

Wilson, M., & Knoblich, G. (2005). The case for motor involvement in perceiving conspecifics. *Psychological bulletin*, *131*(3), 460.