Disfluency Insertion for Spontaneous TTS: Formalization and Proof of Concept

Raheel Qader, Gwénolé Lecorvé, Damien Lolive, Pascale Sébillot

Disfluency Insertion for Spontaneous TTS: Formalization and Proof of Concept*

Raheel Qader[1], Gwénolé Lecorvé[1], Damien Lolive[1], and Pascale Sébillot[2]

[1] Univ Rennes, CNRS, IRISA – 22300 Lannion, France
[2] Univ Rennes, Inria, CNRS, IRISA – 35000 Rennes, France
{firstname.lastname}@irisa.fr

Abstract. This paper presents an exploratory work to automatically insert disfluencies in text-to-speech (TTS) systems. The objective is to make TTS more spontaneous and expressive. To achieve this, we propose to focus on the linguistic level of speech through the insertion of pauses, repetitions and revisions. We formalize the problem as a theoretical process, where transformations are iteratively composed. This is a novel contribution since most of the previous work either focus on the detection or cleaning of linguistic disfluencies in speech transcripts, or solely concentrate on acoustic phenomena in TTS, especially pauses. We present a first implementation of the proposed process using conditional random fields and language models. The objective and perceptual evalation conducted on an English corpus of spontaneous speech show that our proposition is effective to generate disfluencies, and highlights perspectives for future improvements.

Keywords: Disfluencies · Spontaneous speech · Natural language generation.

1 Introduction

Speech disfluencies can be defined as a phenomenon which interrupts the flow of speech and does not add any propositional content [22]. Despite the lack of propositional content, disfluencies have several communicative values. They facilitate synchronization between addressees in conversations [6]. They also improve listening comprehension by creating delays in speech and signaling the upcoming message complexity [14, 23] (cited by [3]). Despite this, current Text-To-Speech (TTS) systems only partially integrate disfluencies. They are thus inadequate to express a spontaneous style, and prevent from high user acceptability in some human-machine interactions (e.g. personnal assistants, avatars). To tackle the issue, this paper investigates the automatic insertion of disfluencies.

This paper proposes a novel formalization of the disfluency generation mechanism. This formalization enables controlling the nature and proportion of the

* This study has been realized under the ANR (French National Research Agency) project SynPaFlex ANR-15-CE23-0015.

disfluencies to be generated. Our proposal is supported by a proof of concept through a first implementation trained on an English corpus. This implementation relies on conditional random fields (CRFs) and language models (LMs) to experimentally demonstrate the ability of our approach to produce plausible disfluent utterances. As exploratory work, no synthesis experiment is carried out because integrating disfluencies in a TTS system requires adaptations on many aspects (underlying speech corpus, prosody prediction, etc.). Thus, the current work conducts the textual validation of the generated disfluent utterances.

In the remainder, Section 2 reviews the domain and presents our motivations. Then, Section 3 introduces the formalization of the problem while its implementation is given in Section 4. Finally, the validation of our work is provided in Section 5 through objective and perceptual evaluations.

2 Review of the Domain and Motivations

According to Shriberg [16], disfluencies are characterized by 3 sections playing a specific role: the *reparandum* region (or RM) which is the sequence of erroneous words ; the *repair* region (RR), i.e. the sequence of corrected words for the RM region ; and finally the so-called *interregnum* section indicating the interruption in the speech stream. In this schema, the point between the reparandum and the interregnum is the *interruption point* (IP). Below is an example of disfluency:

$$I\ think\ \overbrace{she\ will}^{RM}\ \underset{IP}{\uparrow}\ \overbrace{I\ mean}^{IM}\ \overbrace{he\ will}^{RR}\ not\ come\ today. \qquad \text{(Example 1)}$$

Several studies suggest to categorize disfluencies into three main types: pauses, repetitions, and revisions [15, 24, 12]. Pauses are useful to keep the conversation on while the speaker searches for a phrase. Pauses can be silent, filled (e.g., "uh" or "um") or discourse markers ("you know", "well", etc.). Repetitions can be used to gain time and recover the flow of the speech, intensify the effect of an expression, or signal an upcoming problem in the speech [24]. Finally, revisions occur when the speaker slightly fixes his speech after an error. False starts are an extreme case of revisions in which the speaker completely abandons the interrupted speech and starts a fresh one. Hence, revisions help the speaker monitoring his speech.

Most studies on disfluencies are for automatic speech recognition [17, 18, 11, 10, 8] where the main objective improve language modeling and produce disfluency-free transcripts. On the contrary, disfluency generation is still poorly studied in TTS. According to [2], this is because, most of the time, speech databases for TTS systems do not contain any disfluencies, and linguistic processing pipelines in the front-end still badly integrate disfluent sentences, in spite of NLP progresses in the domain [9]. Among existing work, [19, 21] studied the automatic insertion of filled pauses (especially "*uh*", "*um*") using finite state acceptors or word lattices. Other studies like [1, 7, 4] have formalized the problem as searching for an IP using machine learning before selecting, among a set of

possibilities, the best words to be inserted according to probabilities given by an LM. This approach is also adopted in our work. Although this approach relies on the reductive hypothesis that disfluencies are predictable based on shallow (non-psychological) cues (raw words, parts of speech, etc.) [7], the resulting disfluencies have shown to feign personality traits [25]. Likewise, recent work has studied the acoustic aspects of lengthenings and filled pauses w.r.t. the perception of uncertainty [20].

Among limitations, most of these studies concentrate on one type of disfluencies (mostly filled pauses). Recently, [5] proposed to model several types of pauses. Following the same objective, we introduce a rich formalization, able to integrate repetitions and revisions in addition to pauses. Then, Shriberg's schema of disfluencies is useful to determine whether an utterance is disfluent or not, but it does not explain how to move from a fluent to a disfluent utterance, especially when disfluencies are interwinted. To solve this problem, we propose to decompose this schema such that it can be used to generate disfluencies in a deterministic way. Finally, it is worth noting that disfluency generation, as usually in natural language generation, is difficult to evaluate since several outputs are generally acceptable in these problems. This makes it particularly difficult to compute objective measures when data, as is the case in our work, contains only one reference to be compared with. This problem is discussed in Section 5.

3 Disfluency Generation Process

In this work, we propose a complete process for disfluency generation. The key idea is to compose disfluencies of elementary types. This section presents the whole process, each disfluency type, and the composition mechanism.

3.1 Main Principles

The proposed process considers a disfluency as the result of a transformation function on a fluent utterance. Hence, an utterance with multiple disfluencies results from successively composing transformation functions. In practice, one transformation function is defined for each disfluency type. That is, given a disfluency type T, the transformation function f_T reads a sequence of n words $\mathbf{w} \in V^n$, where V denotes the vocabulary, and returns a sequence of m words, $m > n$. In practice, each function f_T consists of two sub-functions: π_T, which determines the IP position, and ω_T which inserts the actual disfluent words using the result of π_T. Mathematically, these two functions can be defined as below:

$$\pi_T : \quad V^n \quad \rightarrow \quad [\![0, n]\!] \ , \tag{1}$$

$$\text{and} \quad \omega_T : V^n \times [\![0, n]\!] \rightarrow \quad V^m \ . \tag{2}$$

Thus, f_T is simply calculated as $\omega_T(\mathbf{w}, \pi_T(\mathbf{w}))$. Sub-functions have been chosen to be specific on disfluency types since IPs may not appear in the same context according to the type, and each type has its own structure, expressible through Shriberg's schema and described in the following.

3.2 Disfluency Functions

Pauses can syntactically be seen as a simple interruptions, without any RM nor RR, solely reduced to an IM. This IM can be instantiated by different pause tokens, in our work those present in the corpus used for the experiments: "*<silence>*", "*uh*", "*um*", "*you know*" "*I mean*" and "*well*". This list can obviously be extended in order to make the whole process richer. The following is an example of a pause transformation from a fluent utterance:

$$\mathbf{w} : \textit{once you get to a certain degree of frustration,}$$

$$f_{\text{pause}}(\mathbf{w}) \quad : \textit{once you get to a certain degree of } \underset{IP}{\uparrow} \overbrace{\textit{uh}}^{IM} \textit{ frustration.}$$

(Example 2)

To make the link with the sub-functions presented earlier, the IP here is determined by the π_{pause} function and the choice of the word(s) to be inserted is made by the ω_{pause} function. Repetitions are duplications of one or few words, i.e., their RM and RR regions are identical. Due to the proposed composition mechanism, no IM is considered, as follows: Thus, all repetition are treated as the following example:

$$\mathbf{w} : \textit{and I think this happens to a lot of people,}$$

$$f_{\text{repetition}}(\mathbf{w}) \quad : \textit{and } \overbrace{\textit{I think}}^{RM} \underset{IP}{\uparrow} \overbrace{\textit{I think}}^{RR} \textit{ this happens to a lot of people.}$$

(Example 3)

A repetition with a pause in the middle is considered as 2 disfluencies. The scope of the repetition, i.e., length of RM and RR, is determined by the sub-function $\omega_{\text{repetition}}$. In a similar fashion, revisions do not include any IM, and ω_{revision} determines the span of the RR region and generates the RM. As opposed to repetitions, the predicted RM differs from the RR region. An example of revision is given below:

$$\mathbf{w} : \textit{that is so that if whoever would get it,}$$

$$f_{\text{revision}}(\mathbf{w}) \quad : \textit{that is so that } \overbrace{\textit{if you}}^{RM} \underset{IP}{\uparrow} \overbrace{\textit{if whoever}}^{RR} \textit{ would get it.}$$

(Example 4)

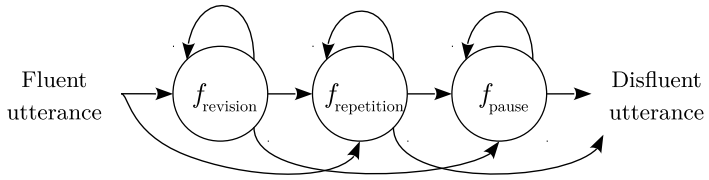

Fig. 1. Whole disfluency generation process.

3.3 Composition of Disfluency Functions

Composition is the only way to generate all disfluency regions and several disfluencies. For instance, an utterance containing a revision and a pause can be seen as the result of $f_{\text{revision}} \circ f_{\text{pause}}$. However, this may be also the result of $f_{\text{pause}} \circ f_{\text{revision}}$. To minimize such ambiguities and make the process deterministic, the following precedence order is defined:

$$\text{revision} \prec \text{repetition} \prec \text{pause} \ . \tag{3}$$

Thus, for the given example, the composition $f_{\text{pause}} \circ f_{\text{revision}}$ is forbidden. This order is justified by the fact that knowing where revisions and repetitions are can be useful to determine where to insert pauses. Technically, it is also easier to insert a pause in between repeated words than inserting repeated words around one or several pause tokens. Likewise, inserting revisions after repetitions would break repetitions, whereas repetitions applied on top of revisions may help strengthening revisions.

Following this precedence order, the generation process is as given in Figure 1. Starting from a fluent utterance, and in the respect of , each type of disfluency can be applied zero, one or several times. Below is an example of consecutive transformations:

$$\begin{aligned}
& \textit{I have to go.} \\
f_{\text{revision}} \quad & [\textbf{I want to } \textit{I have to}]_{\text{revision}}\, \textit{go.} \\
\circ\, f_{\text{repetition}} \quad & \Big[\textit{I want}\,[\textbf{to } \textit{to}]_{\text{repetition}}\, \textit{I have to}\Big]_{\text{revision}}\, \textit{go.} \\
\circ \quad f_{\text{pause}} \quad & \Big[\textit{I want}\,[\textit{to to}]_{\text{repetition}}\, [\textbf{uh}]_{\text{pause}}\, \textit{I have to}\Big]_{\text{revision}}\, \textit{go.} \\
\circ \quad f_{\text{pause}} \quad & \Big[\textit{I want}\,[\textit{to to}]_{\text{repetition}}\, [\textit{uh}]_{\text{pause}}\, [\textbf{I mean}]_{\text{pause}}\, \textit{I have to}\Big]_{\text{revision}}\, \textit{go.}
\end{aligned} \tag{Example 5}$$

These hierarchized iterative compositions can easily be formulated as an actual algorithm and implemented, as described in the next section.

4 Implementation

This section presents one way to implement the disfluency generation process. The objective of this implementation is to validate the approach before studying

input : `OriginalUtt`: a fluent utterance
output: input utterance with added disfluencies

1 **data:**
2 `Types`: list of disfluency types
3 `OutputUtt`: sequence of words
4 `IP`: integer
5 `Types` $\leftarrow$ [*repetition*, *pause*]
6 `OutputUtt` $\leftarrow$ `OriginalUtt`
7 **for each** `T` $\in$ `Types` **do**
8 `IP` $\leftarrow \pi_T$(`OutputUtt`)
9 **while** $\neg$ *StoppingCriterion(*`T`, `OutputUtt`, `IP`*)* **do**
10 `OutputUtt` $\leftarrow \omega_T$(`OutputUtt`, `IP`)
11 `IP` $\leftarrow \pi_T$(`OutputUtt`)
12 **return** `OutputUtt`

Algorithm 1. Main algorithm for disfluency generation.

richer and more efficient implementations in the future. For this reason, we limit this study to pauses and repetitions, and set aside the more difficult case of revisions. This configuration is minimal but functionnal since it enables testing the composition mechanism.

For each disfluency type T, the IP prediction function π_T is treated as a labeling task achieved using a CRF, while the word insertion function ω_T is the selection of the best phrase among a set of automatically built candidates, the selection criterion relying on an LM. In short, the whole process is built on 2 CRFs and 2 LMs. We describe the main algorithm, then these models.

4.1 Main Algorithm

Algorithm 1 presents how to transform an input utterance to a disfluent one. Each type T is examined in a same manner, following the precedence order. The algorithm tries to determine a potential IP (line 8). If this IP is accepted according to a stopping criterion (l. 9), a new disfluency of type T is added to the current version of the utterance being transformed (l. 10). Then, a next IP proposal is computed (l. 11). As soon as an IP is rejected by the stopping criterion, the algorithm moves to the next disfluency type (l. 7) or, if none anymore, returns the transformed utterance (l. 12). The stopping criterion stops insertions as soon as, for the current type T, the proportion of disfluencies of this type in the transformed utterance reaches a maximum threshold fixed by the user. In practice, these thresholds have been set to 1 % and 12 %, respectively for repetitions and pauses, as observed on average in the training corpus.

4.2 IP Prediction

IP prediction is carried out by a CRF on an input (fluent or disfluent) sequence of words and potentially associated features. This CRF is trained to categorize

successive words under two labels: words that are followed by an IP, and the others. At runtime, the CRF produces a list of IPs which are examined in turn until finding out one which has not been exploited yet. This requirement for fresh IPs at each iteration prevents the method from indefinitely adding disfluencies at the sole best place deemed by the CRF. If no new IP is found, the main algorithm moves to the next disfluency type. The examined IPs are those returned for each labelling hypothesis in the N-best list of the CRF. They are sorted by descending posterior probability. Falling back on N-best lists ensures a very large choice of IPs, delegating the termination decision to the stopping criterion.

4.3 Insertion of New Words

Given a chosen IP, the word insertion step seeks to produce a disfluency that best integrates into the utterance. The proposed implementation of ω_T constructs a set of possible word sequences, centered on the IP, and then determines the most probable w.r.t. type T. For repetitions, candidates are RM/RR pairs of various lengths. As for pauses, 6 candidates are proposed, one for each considered pause tokens able to fill the IM. Candidate sequences are descriminated by comparing their probability within their local contexts (± 3 words around the IP). The probability for type T is computed by an n-gram LM trained on T-specific disfluent data. For an IP at position i in a sequence of words $\mathbf{w} = [w_1 \cdots w_N]$, let a disfluent section under examination $\mathbf{d} = [d_1 \cdots d_D]$, and the left/right surrounding words $\mathbf{w}^{(\ell)} = [w_{i-W+1} \cdots w_i]$ and $\mathbf{w}^{(r)} = [w_{i+1} \cdots w_{i+W}]$ respectively. The proper integration of $\mathbf{d}$ within $\mathbf{w}$ can be measured through either the average probability per word or the global probability conditionned on $\mathbf{d}$, respectively defined as:

$$\frac{\Pr(\mathbf{w}^{(\ell)}\mathbf{d}\mathbf{w}^{(r)})}{2W + D} \tag{4}$$

$$\text{and } \Pr(\mathbf{w}^{(\ell)}\mathbf{d}\mathbf{w}^{(r)}|\mathbf{d}) = \frac{\Pr(\mathbf{w}^{(\ell)}\mathbf{d}\mathbf{w}^{(r)})}{\Pr(\mathbf{d})} \tag{5}$$

Since utterances are processed independently, the first measure favors over-insertion of the most frequent tokens from the training corpus. On the contrary, the second disregards the prior probability of the disfluent tokens and solely focuses on how the final word sequence flows well. As a consequence, it leads to over-generating rare tokens. In this paper, a linear interpolation with equal weights associated to each measure is chosen.

5 Experimental Validation

The proposed implementation has been tested on 20 h from the Buckeye corpus [13], an American English conversational speech corpus made of individual interviews with 20 speakers. Manual transcripts (150K words) are annotated with $2,714$ repetitions and $20,264$ pauses. For each type of disfluencies, a dedicated version of the corpus is derived where utterances with no disfluency of that

type were filtered out. To be consistent with precedence order, all pauses were removed from the repetition-specific corpus. An entirely cleaned (fluent) version of the corpus was also built. Data is divided into 3 sets: one to train the models (*train*, 60% of the utterances), another to tune hyper-parameters (*development*, 20%), and a set for evaluation (*test*, 20%). CRFs were trained using Wapiti[3] and LMs are trigrams trained with SRILM[4]. The remainder presents the different evaluations conducted to validate the proposed approach.

5.1 Objective Evaluation

IP predictions are examined through precision, recall and F1-score compared to the reference from our corpus, i.e., a predicted IP is a true positive if it is placed at the exact same position as an IP of the reference utterance. In the absence of multiple references, these measurements are difficult to interpret. Thus, we also propose to introduce the Interruption Rate Ratio (IRR) between the predictions and the reference, i.e., the scale factor between the average number of IPs per sentence in our hypotheses and in the reference. For example, IRR with value 1 indicates an equal proportion of IPs, 0.6 means an under-prediction of 40 %, and 2.2 an over-prediction of 120 %. Word insertion is evaluated by the LM perplexity given to the generated sequences. Since LMs are also used to select disfluency candidates, this measure is biased but it is primary used to understand the general behavior of the proposition. Disfluent sentences are expected to get lower perplexities than fluent sentences.

Different CRF training settings were studied in preliminary experiments on the development set for IP prediction. Two factors have been studied and adjusted: the optimal set of features and the size of contextual information for each word, i.e., the size of the observed neighborhood window. As a result, it turns out that our best results are obtained with very few attributes, namely raw words and part of speech (POS). As for the neighborhood, a window of a few words (2 in the final experiments) around the word being examined is beneficial.

On the test set, the compared configurations are: the cleaned utterances (*cl.*), their disfluent reference (*ref.*), and utterances produced by our models with the previously exposed features. Regarding pause insertion, an extra feature is introduced to tell the CRF whether a word under study comes from the original fluent sentence or has been added along iterations. This intents to integrate dependencies across transformations, as for instance desired to insert a pause in a repetition. We remind that the reference for repetitions do not contain any pause, whereas the cleaned version for pauses can include repetitions.

Tables 1 and 2 show the results obtained for the repetitions (R) and pauses (P). First, the results are globally low, especially for repetitions. These results can be explained by the relatively small amount of learning data and the uniqueness of our reference. IRRs show that generated utterances have always fewer disfluencies than the reference, because of the adopted stopping criterion. To

[3] `http://wapiti.limsi.fr/`
[4] `http://www.speech.sri.com/projects/srilm/`

Table 1. Objective evaluation of repetitions on the test set.

	Features	Window?	Recall	Prec.	F1	IRR	PPL
($R_{cl.}$)	Fluent (cleaned) utterance					0.0	241
($R_{ref.}$)	Disfluent reference utterances					1.0	236
(R_A)	Words	no	0.8%	3.8%	1.3	0.1	236
(R_B)	+ POS	yes	**6.2%**	**17.1%**	**9.2**	0.4	**231**

Table 2. Objective evaluation of pauses on the test set.

	Features	Window?	Recall	Prec.	F1	IRR	PPL
($P_{cl.}$)	Fluent (cleaned) utterances					0.0	242
($P_{ref.}$)	Disfluent reference utterances					1.0	**172**
(P_A)	Words	no	8.2%	29.4%	12.8	0.5	209
(P_B)	+ POS	yes	17.9%	33.6%	23.3	0.7	191
(P_C)	+ prev. disfl.	yes	**19.8%**	**34.5%**	**25.1**	0.7	188

our knowledge, no comparative work exists for repetitions, and considering a wide range of pause tokens (not only "uh" and "um") is rather difficult [21]. Hence, these results are acceptable for a first implementation. In terms of perplexity, the generated disfluencies are rather close to the reference. Moreover, perplexities and IRRs on pauses highlight, as expected, that a high proportion of pauses brings a low perplexity. Finally, information about previous iterations of the algorithm, i.e., knowing which words are inserted (disfluent) words, seems beneficial, as shown in particular by the increase of about 2 percentage points of the F1-score (P_B *vs.* P_C).

5.2 Perceptual Tests

Two series of perceptual tests were conducted on 24 participants. The first series separately studies the effects of repetitions and pauses, while the second seeks to measure their combined effects. Based on a displayed fluent text, testers had to imagine how it could be uttered during a spontaneous conversation, and gave their opinion on several proposals, ranging from 0 (impossible utterance) to 10 (perfectly possible). A same set of 40 utterances from the test set is used for all experiments (4-25 words, all selected so that their disfluent reference contains a mixture of repetitions and pauses, not necessarily interleaved).

Mean opinion scores (MOSs, confidence interval $\alpha = 0.05$) are reported in Figure 2 for the first series of tests. System labels are the same as in Tables 1 and 2. First, these results are close from one configuration to another, and differences are generally insignificant, even between the cleaned and reference utterances. This seems to show that the perception of disfluencies is a difficult task, at least when presented in a textual form. On repetitions, it appears that configurations with no or few repetitions ($R_{cl.}$ and R_A) are preferred to those containing more ($R_{ref.}$ and R_B). This can be explained by the absence of pauses in the middle of the presented repetitions. Then, the results on pauses seem

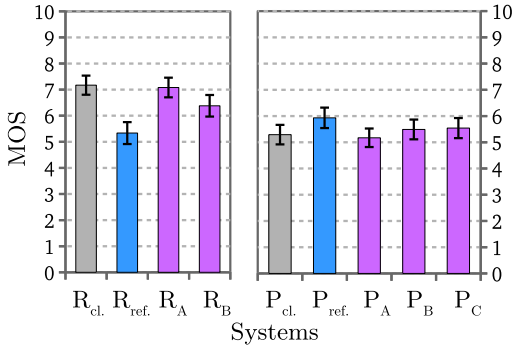

Fig. 2. MOS on repetitions (left) and pauses (right). System labels are as in Table 1.

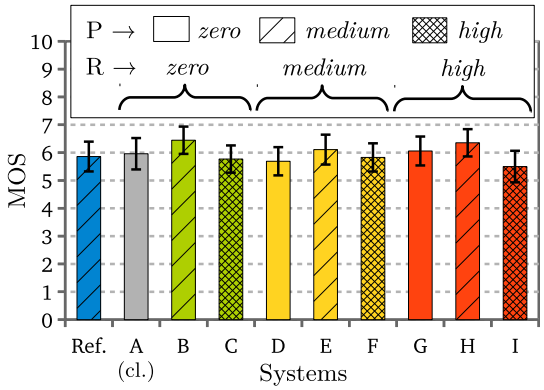

Fig. 3. MOS for mixed repetitions (R) and pauses (P). System labels are as in Table 2.

correlated with the proportion of pauses (see IRRs in Table 2), although significance is not proven. On the whole, these two tests show that the automatically generated utterances do not denote w.r.t. the reference.

The incidences of the disfluency proportions, and of combining repetitions and pauses are studied in the second series of experiments. For each type, three insertion levels are considered by modulating the stopping criterion threshold (see Algorithm 1): *zero* means no disfluency of the considered type ; *medium*, a proportion consistent with the training set ; or *high*, 3 times more disfluencies than in the corpus. MOSs of these tests are in Figure 3. Again, the results are all very similar. Nevertheless, the absence of low scores means that the disfluency composition mechanism produces plausible utterances, which is the first motivation of these tests. Then, two trends emerge: first, the absence of pause or their strong presence are badly perceived compared to the intermediate setting (B, E, H), then the most disfluent utterances (I) get the lowest MOS.

As a conclusion, the perceptual tests show that the utterances produced by our method are acceptable in comparison to clean ones and to disfluent ones as uttered in real situations. This tends to validate the proof-of-concept im-

plementation and the underlying proposed formalization. The small differences between configurations however encourage one to improve this implementation and to think about more discriminating ways to conduct perceptual tests.

6 Conclusion

In this paper, we have presented an innovative formalization for the automatic insertion of disfluencies in texts. The ultimate goal of this work is to make synthetic speech signals more spontaneous, and thus more acceptable in some human-machine interactions. We have introduced a theoretical process of disfluency composition and provided a first implementation based on CRFs and LMs. The experiments conducted on this implementation show that the proposed process is functional, although perfectible.

A first perspective is now the extension to revisions. Since the validation in this paper, this work has been achieved. The word insertion part, which is the difficult part, has been implemented by altering words from the RR with linguistically similar ones, i.e., words with the same POS and geometrically close in a lexical embedding space. Evaluation will be conducted in the near future. Among other perspectives, more complex models could be tested, for instance to enable including broader, non-lexical, considerations (phonetic confusion, speaker intention, etc.). However, collecting training data is an obstacle here. Finally, evaluation is a challenge. The best improvement track on this point seems to us to provide natural realizations of all the tested utterances. This would avoid bypass the unsuitability of current TTS systems but it requires recording people.

References

1. Adell, J., Bonafonte, A., Escudero, D.: Filled pauses in speech synthesis: towards conversational speech. In: Proceedings of Text, Speech and Dialogue (TSD) (2007)
2. Adell, J., Bonafonte, A., Mancebo, D.E.: On the generation of synthetic disfluent speech: local prosodic modifications caused by the insertion of editing terms. In: Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech) (2008)
3. Adell, J., Escudero, D., Bonafonte, A.: Production of filled pauses in concatenative speech synthesis based on the underlying fluent sentence. Speech Communication **54** (2012)
4. Andersson, S., Georgila, K., Traum, D., Aylett, M., Clark, R.A.: Prediction and realisation of conversational characteristics by utilising spontaneous speech for unit selection. In: Proceedings of Speech Prosody (2010)
5. Betz, S., Wagner, P., Schlangen, D.: Micro-structure of disfluencies: Basics for conversational speech synthesis. Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech) (2015)
6. Clark, H.H.: Speaking in time. Speech Communication **36** (2002)
7. Dall, R., Tomalin, M., Wester, M., Byrne, W.J., King, S.: Investigating automatic & human filled pause insertion for speech synthesis. In: Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech) (2014)

8. Hassan, H., Schwartz, L., Hakkani-Tür, D., Tür, G.: Segmentation and disfluency removal for conversational speech translation. In: Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech) (2014)
9. Honnibal, M., Johnson, M.: Joint incremental disfluency detection and dependency parsing. Transactions of the Association for Computational Linguistics **2** (2014)
10. Kaushik, M., Trinkle, M., Hashemi-Sakhtsari, A.: Automatic detection and removal of disfluencies from spontaneous speech. In: Proceedings of the Australasian International Conference on Speech Science and Technology (SST) (2010)
11. Liu, Y., Shriberg, E., Stolcke, A., Hillard, D., Ostendorf, M., Harper, M.: Enriching speech recognition with automatic detection of sentence boundaries and disfluencies. IEEE Transactions on Audio, Speech, and Language Processing **14** (2006)
12. Boula de Mareüil, P., Habert, B., Bénard, F., Adda-Decker, M., Barras, C., Adda, G., Paroubek, P.: A quantitative study of disfluencies in french broadcast interviews. In: Proceedings of Disfluency in Spontaneous Speech Workshop (2005)
13. Pitt, M.A., Johnson, K., Hume, E., Kiesling, S., Raymond, W.: The Buckeye corpus of conversational speech: labeling conventions and a test of transcriber reliability. Speech Communication **45** (2005)
14. Rose, R.L.: The communicative value of filled pauses in spontaneous speech. Ph.D. thesis, University of Birmingham (1998)
15. Shriberg, E.E.: Phonetic consequences of speech disfluency. Tech. rep., DTIC Document (1999)
16. Shriberg, E.E.: Preliminaries to a theory of speech disfluencies. Ph.D. thesis, University of California (1994)
17. Stolcke, A., Shriberg, E.: Statistical language modeling for speech disfluencies. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (1996)
18. Stolcke, A., Shriberg, E., Bates, R.A., Ostendorf, M., Hakkani, D., Plauche, M., Tür, G., Lu, Y.: Automatic detection of sentence boundaries and disfluencies based on recognized words. In: Proceedings of the International Conference on Spoken Language Processing (ICSLP) (1998)
19. Sundaram, S., Narayanan, S.: An empirical text transformation method for spontaneous speech synthesizers. In: Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech) (2003)
20. Székely, E., Mendelson, J., Gustafson, J.: Synthesising uncertainty: the interplay of vocal effort and hesitation disfluencies. Proceedings of the Annual Conference of the Internationa Speech Communication Association (Interspeech) (2017)
21. Tomalin, M., Wester, M., Dall, R., Byrne, W., King, S.: A lattice-based approach to automatic filled pause insertion. In: Proceedinds of the Workshop on Disfluency in Spontaneous Speech (2015)
22. Tree, J.E.F.: The effects of false starts and repetitions on the processing of subsequent words in spontaneous speech. Journal of Memory and Language **34** (1995)
23. Tree, J.E.F.: Listeners' uses ofum anduh in speech comprehension. Memory & cognition **29** (2001)
24. Tseng, S.C.: Grammar, prosody and speech disfluencies in spoken dialogues. Unpublished doctoral dissertation. University of Bielefeld (1999)
25. Wester, M., Aylett, M.P., Tomalin, M., Dall, R.: Artificial personality and disfluency. In: Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech) (2015)