A Robust Speech Recognition System for Service-Robotics Applications

Masrur Doostdar, Stefan Schiffer, Gerhard Lakemeyer

Knowledge-based Systems Group
Department of Computer Science 5
RWTH Aachen University, Germany
doostdar@kbsg.rwth-aachen.de, {schiffer,gerhard}@cs.rwth-aachen.de

Abstract. Mobile service robots in human environments need to have versatile abilities to perceive and to interact with their environment. Spoken language is a natural way to interact with a robot, in general, and to instruct it, in particular. However, most existing speech recognition systems often suffer from high environmental noise present in the target domain and they require in-depth knowledge of the underlying theory in case of necessary adaptation to reach the desired accuracy. We propose and evaluate an architecture for a robust speaker independent speech recognition system using off-the-shelf technology and simple additional methods. We first use close speech detection to segment closed utterances which alleviates the recognition process. By further utilizing a combination of an FSG based and an N-gram based speech decoder we reduce false positive recognitions while achieving high accuracy.

1 Introduction

Speech recognition is a crucial ability for mobile service robots to communicate with humans. Spoken language is a natural and convenient means to instruct a robot if it is processed reliably. Modern speech recognition systems can achieve high recognition rates, but their accuracy often decreases dramatically in noisy and crowded environments. This is usually dealt with by either requiring an almost noise-free environment or by placing the microphone very close to the speaker's mouth. Although we already assume the latter, all requirements for a sufficiently high accuracy cannot always be met in realistic scenarios.

Our target application is a service-robotics domain, in particular, the ROBOCUP-@HOME league [1], where robots should assist humans with their everyday activities in a home-like environment. Any interaction with a robot has to be done in a natural fashion. That is to say, instructions issued to the robot may only be given by means of gestures or natural spoken language. An important property of the domain, especially at a competition, is the high amount of non-stationary background noise and background speech. A successful speech recognition system in ROBOCUP@HOME must be able to provide robust speaker-independent recognition of mostly command-like sentences. For one, it is important that commands given to the robot are recognized robustly. For another, spoken language not directed to the robot must not be matched to an instruction for the robot. This is a non-trivial task in an environment with a high amount of background noise. That is why most teams use a head mounted microphone for their speech recognition. Still, it is not easy to determine which audio input is actually addressed to the robot and which one is not. This is even more so, since within a competition there usually is a person that describes to the audience what is currently happening in the arena via loudspeakers. The words used for the presentation often are very similar if not even the same used to instruct the robot. This complicates the task of robust speech recognition even more.

We propose an architecture that tackles the problem of robust speech recognition in the above setting. It comprises two steps. First, we use a threshold based close speech detection module to segment utterances targeted at the robot from the continuous audio stream recorded by the microphone. Then, we decode these utterances with two different decoders in parallel, namely one very restrictive decoder based on finite state grammars and a second more lenient decoder using N-grams. We do this to filter out false positive recognitions by comparing the output of the two decoders and rejecting the input if it was not recognized by both decoders.

The paper is organized as follows. In Section 2, we describe some basics of common speech recognition systems. Then we propose our architecture and discuss related work. We go into detail about our close speech detection in Section 3 and the dual decoding in Section 4. After an experimental evaluation in Section 5, we conclude in Section 6.

2 Foundations and Approach

We first sketch properties of common statistical speech recognition systems. Then we propose a system architecture that tries to combine the features of different approaches to tackle the problems present in our target domain.

2.1 Statistical Speech Recognition

Most statistical speech recognition systems available today use hidden Markov models (HMMs). For a given vector of acoustical data x they choose a sequence of words w_{opt} as best-hypothesis by optimizing

$$w_{opt} = \arg\max_{w} p(w|x) = \arg\max_{w} \frac{p(x|w) \cdot p(w)}{p(x)} = \arg\max_{w} p(x|w) \cdot p(w). \qquad (1)$$

Here $p(w|x)$ is the posterior probability of w being spoken, the fundamental Bayes' decision rule is applied to it. The constant normalization probability $p(x)$ can be omitted for the maximization. $p(x|w)$ denotes the probability of observing the acoustical data x for the assumption of w being spoken. This is given to the recognizer by the acoustic-model. $p(w)$ denotes the probability of the particular word-sequence w occurring. This prior probability is provided to the recognizer by the so-called language-model. The language model can either be represented by a grammar, e.g. a *finite state grammar* (FSG), or by means of a statistical model, mostly in form of so-called N-*grams* that provide probabilities for a word dependent on the previous (N-1) words. Common speech-recognizers use 3-grams, also called *TriGrams*.

Standard statistical speech recognition systems process a given speech utterance time-synchronously. Each time-frame, possible sub-word-units (modeled by HMM-states) and word-ends are scored considering their acoustical probability and their language model probability. Most of the possible hypotheses score considerably worse than the best hypothesis at this time frame and are pruned away. In the search for a best hypothesis information about possible near alternatives can be kept to allow for useful post-processing. For each time-frame, the most probable hypotheses of words ending at that frame are stored along with their acoustical scores. This information can be appropriately represented by a directed, acyclic, weighted graph called *word-graph* or *word-lattice*. Nodes and edges in the graph denote words, their start-frames and their acoustic likelihoods. Any path through the graph, starting at the single start-node and ending in the single end-node, represents a hypothesis for the complete utterance. By combining the acoustical likelihoods of the words contained along this path with the language model probability we obtain the score $p(x|w) \cdot p(w)$. The so-called N*-best list* contains all possible paths through the word-graph that were not pruned in the search, ordered by their score.

The language model used in searching for hypotheses largely influences the performance of a speech recognition system. Thus, it is crucial to choose a model appropriate for the particular target application to achieve sufficiently good results. On the one hand, FSG-based decoders perform good on sentences from their restricted grammar. On the other hand, they get easily confused for input that does not fit the grammar used. This can lead to high false recognition rates. N-gram based language models are good for larger vocabularies, since utterances do not have to follow a strict grammar.

2.2 Approach

For our target application we are confronted with a high amount of non-stationary background noise including speech similar to the vocabulary used to instruct the robot. Only using an FSG-based decoder would lead to high false recognition rates. We aim to eliminate false recognitions with a system that exploits the properties of different language models described above. In a first step, we try to segment utterances that are potential speech commands issued to the robot from the continuous audio stream recorded by the robot's microphone. Then, we decode those utterances using two decoders in parallel, one FSG-based and a second TriGram based one to combine the benefits of both. An overview of our system's architecture is shown in Figure 1.

Segmentation of close speech sections

We employ a module that is supposed to segment *close speech sections* from the continuous live stream, i.e. sections where the main person speaks closely into the microphone. We call this *close speech detection* (CSD). Doing this provides us with two advantages. First, with the (reasonable) presupposition that the speech to be recognized, we call it *positive speech*, is being carried out close to the microphone, we can discriminate it from other (background) speech events that are not relevant and thus may cause false recognitions. These false recognitions would be wrongly matched to a speech command for the robot. Second, the performance of speech recognition engines

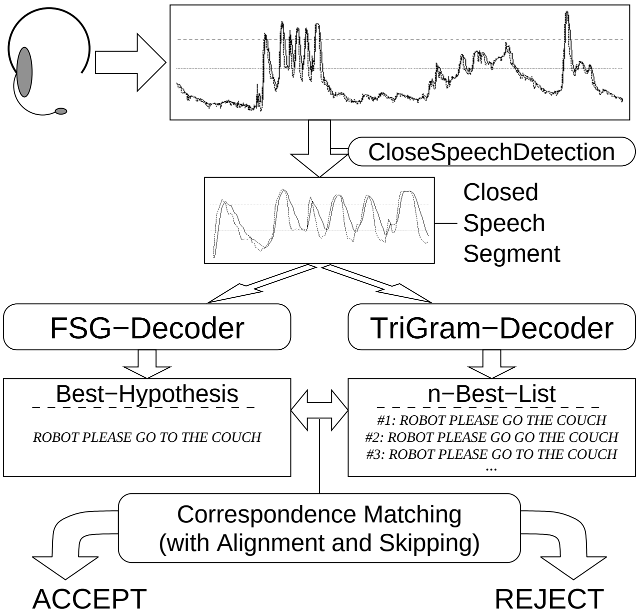

CloseSpeechDetection
Closed
Speech
Segment
FSG−Decoder
TriGram−Decoder
Best−Hypothesis
ROBOT PLEASE GO TO THE COUCH
n−Best−List
#1: ROBOT PLEASE GO THE COUCH
#2: ROBOT PLEASE GO GO THE COUCH
#3: ROBOT PLEASE GO TO THE COUCH
...
Correspondence Matching
(with Alignment and Skipping)
ACCEPT
REJECT

Fig. 1: Architecture of our dual decoder system

like SPHINX [2] increases considerably, if the speech input occurs in closed utterances instead of a continuous stream. Furthermore, we are able to reduce the computational demands for speech recognition if we only process input of interest instead of decoding continuously. This is especially useful for mobile robotic platforms with limited computing power.

Multiple decoders

As already mentioned, different types of decoders exhibit different properties we would like to combine for our application. For one, we are interested in the high accuracy that very restricted FSG decoders provide. But we cannot afford to accept a high rate of falsely recognized speech that is then probably matched to a legal command. For another, TriGram-based decoders are able to reliably detect words from a larger vocabulary and they can generate appropriate hypotheses for utterances not coming from the grammar, i.e. that are not positive speech. However, that comes with the drawback of an increased error rate in overall sentence recognition. Still, a sentence at least similar to the actual utterance will very likely be contained in the N-best list, the list of the n hypotheses with the highest score. By decoding with an FSG and a TriGram decoder in parallel we seek to eliminate each decoders drawbacks retaining its benefits. We can look for similarities between the FSG's output and the N-best list of the trigram decoder. This allows detecting and rejecting *false positive recognitions* from the FSG decoder.

In principle, any automatic speech recognition system that provides the ability to use both, FSG and N-gram based decoding with N-best list generation, could be employed within our proposed architecture. We chose to use SPHINX 3 because it is a freely available open source software, it is under active development with good support, it is flexible to extend, and it provides techniques for speaker adaption and acoustic model generation. For an overview of an earlier version of the SPHINX system we refer to [2].

2.3 Related work

For speech detection, also referred to as speech activity detection (SAD), endpoint detection, or speech/non-speech classification, speech events have to be detected and preferably also discriminated against non-speech events on various energy levels. There has been work on this problem in the last decades, some of which also employs threshold-approaches like an earlier work of Rabiner detecting energy pulses of isolated words [3], and [4]. One of the main differences to our approach is that they dynamically adapt the threshold to detect speech on various energy levels. For our application, however, it is more preferable to use a static threshold since the environmental conditions may vary but the characteristics of the aural input of interest do not. Furthermore, we dynamically allocate the distance allowed between two spoken words and we apply simple smoothing of a signal's energy-value sequence. For a more general solution to the problem of speech activity detection threshold-based approaches often do not work since they are not robust enough on higher noise levels. More robust approaches use, for example, linear discriminant analysis (LDA) like [5] and [6], or HMMs on Gaussian Mixtures [7]. Our aim is to use detection of close speech only as a pre-processing step before decoding. That is why we do not want to put up the additional costs for these more sophisticated approaches.

To improve the accuracy of speech-recognition systems on grammar-definable utterances while also rejecting false-positives, usually in-depth knowledge of the low-level HMM-decoding processes is required. There has been work on integrating N-grams and finite state grammars [8] in one decoding process for detecting FSG-definable sentences. They assume that the sentences to detect are usually surrounded by carrier phrases. The N-gram is aimed to cover those surrounding phrases and the FSG is triggered into the decoding-process if start-words of the grammar are hypothesized by the N-gram. To reject an FSG-hypothesis they consult thresholds on acoustical likelihoods of the hypothesized words. Whereas this approach requires integration with low-level decoding processes, our dual-decoder approach only performs some post-processing on the hypotheses of the N-best-list. In combination with the CSD front-end we achieve acceptable performance for our application without modifying essential parts of the underlying system. Instead of using two decoders in parallel, a more common method could be to use an N-gram language model in a first pass and to re-score the resulting word-graph or N-best list using an FSG based language model afterwards. However, independently decoding with an FSG-based decoder can be expected to provide higher accuracy for the best hypothesis than the best hypothesis after re-scoring a word-graph with an FSG language model. It would be promising, though, to combine a two-pass approach or our dual-decoder with a method for statistically approximating confidences [9] (in terms of posterior-probabilities) of hypothesized words given a word-graph. A reliable confidence measure would provide a good method for rejection with a threshold.

3 Close Speech Detection

Our approach to detect and segment sections of close speech from a continuous audio stream is quite simple. It makes use of the straightforward idea that sounds being produced close to the microphone exhibit considerably high energy levels. The *energy values* of an audio input are provided when working with speech recognition systems as they extract cepstral coefficients as features from the acoustic signals. The first value of the cepstral coefficients can be understood as the signal's logarithmic energy value. Close speech is detected by first searching for energy values that exceed some upper threshold. Then, we determine the start and the end-point of the segment. Therefore, we look in forward and backward direction for points where the speech's energy values fall below a lower threshold for some time. Note that this straightforward approach can only detect speech carried out close to the microphone. However, this is expressly aimed for in our application since it provides a simple and robust method to discriminate between utterances of the ”legal speaker” and other nearby speakers as well as background noise.

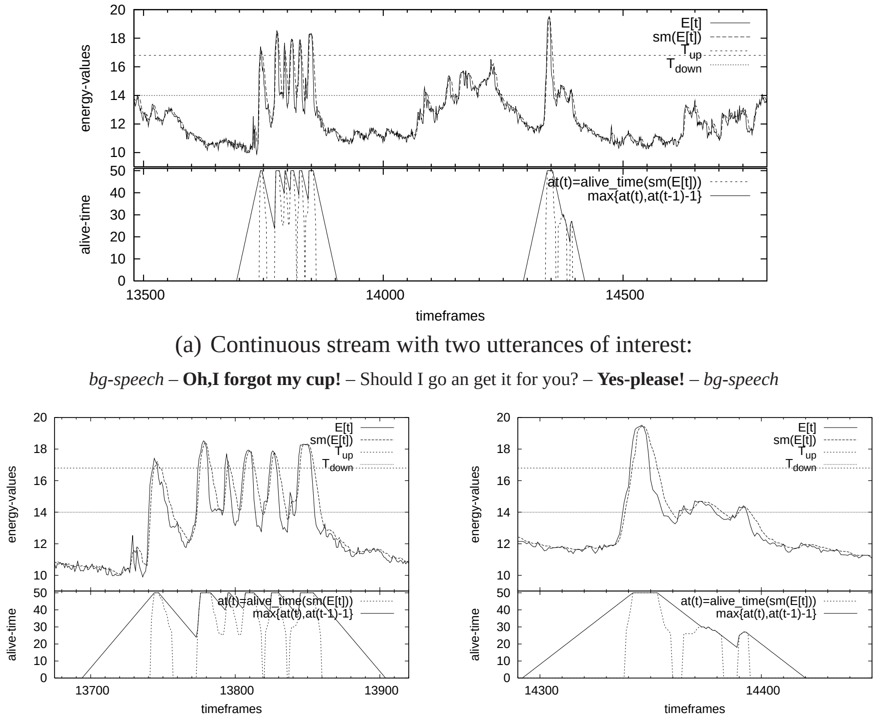
3.1 Detailed Description

Examining a sequence of energy values, speech-segments are characterized by adjacent heaps (see Figure 2: E[t]). For our aim of detecting close speech segments we use two thresholds, namely T_{up} and T_{down}. The first threshold T_{up} mainly serves as a criterion for detecting a close-speech-segment when some energy values exceed it. Thus, T_{up} should be chosen so that for close speech segments some of the heaps are expected to exceed T_{up} while other segments do not.

After this initial detection, the beginning and the end of the speech segment have to be determined such that the resulting segment contains all heaps adjacent to the initial peak. Therefore, starting from the detection point, we proceed in forward and backward direction. We search for points where the energy value drops below the second threshold T_{down} and does not recover again within a certain amount of time-frames. We thereby identify the beginning and the end of the speech segment, respectively. T_{down} should be chosen largest so that still all heaps of a close speech section are expected to exceed it and lowest so that the energy-level of the background noise and most background sound events do not go beyond T_{down}. The amount of time-frames given for recovering again represents the maximal distance we allow between two heaps, i.e. between two consecutive words. We call this the *alive-time*. We further enhance this approach by smoothing the sequence of energy levels before processing it and by dynamically allocating the time to recover from dropping below T_{down}.

Smoothing the energy values The energy-value-sequence is smoothed (cf. $sm(E,t)$ in Figure 2) to prevent punctual variations to take effect on the detection of speech-segments and the determination of the alive-time. We compute the smoothed values by averaging over the current energy-value and the three largest of the previous six energy-values, i.e. for an energy-sequence E and time-frame t we use the smoothing function:

$$sm(E,t) = \frac{1}{4} \cdot \max\{E[t] + E[t_1] + E[t_2] + E[t_3] \mid t_i \in \{t-1, \dots, t-6\}\}.$$



(a) Continuous stream with two utterances of interest:

bg-speech – **Oh,I forgot my cup!** – Should I go an get it for you? – **Yes-please!** – *bg-speech*

(b) First utterance “Oh, I forgot my cup!”

(c) Second utterance “Yes please!”

Fig. 2: Segmentation of close speech segments

Start/End-point detection The amount of time-frames given to recover from falling below T_{down} is not fixed but is determined dependent on the height of the last heap's peak and the distance to this peak. Thus, the closer the peak of the last heap is to T_{down}, i.e. the less the confidence is for the last heap being produced by a close speech, the less time-frames we grant before the next heap must occur (cf. alive-time of right-most heap in Figure 2(c)). This helps to prevent that background sounds which intersect with the close-speech segment or directly succeed/precede them and which exceed T_{down} (like a nearby speech) cause the fixation of the start or end point of a close-speech segment to be postponed over and over again. For energy-values greater than T_{up} we assign the alive-time AT_{up}, for the value T_{down} we assign the alive-time AT_{down} and for values between T_{up} and T_{down} we calculate a time-value by linearly interpolating between AT_{up} and AT_{down}:

$$\text{alive\_time}(v) = \begin{cases} AT_{up} & , v \geq T_{up} \\ \frac{AT_{up} - AT_{down}}{T_{up} - T_{down}} \cdot (v - T_{down}) + T_{down} & , T_{down} < v < T_{up} \\ 0 & , v \leq T_{down} \end{cases}$$

After a close-speech segment is detected we proceed in forward/backward direction (see Figure 2(b) and Figure 2(c)). For each time-frame t, we compute the alive-time at

as the maximum of the value associated with the smoothed energy-value $at(sm(E,t))$ and the alive-time value chosen in the previous time-frame minus one ($at(t-1)-1$). Obviously, when the energy-value-sequence falls below T_{down} the alive-time decreases each time-frame. If 0 is reached before the values recover again, we determine the start and end point of the close-speech-segment at that frame. As soon as we have determined the start point of a segment, we can start passing the input to the decoders. This drastically increases the reactivity of the system. For now, we manually define the actual thresholds T_{up} and T_{down} based on the environmental conditions at a particular site. We fixed AT_{up} at 50 time-frames (500 ms) and AT_{down} at 25 time-frames (250 ms). These values were determined empirically.

4 Dual Decoding

As mentioned in Section 2.1, in statistical SR-systems the optimization of the posterior probabilities $p(w|x)$ is approached by maximizing the scores $p(x|w) \cdot p(w)$ where the likelihood $p(x|w)$ is given by an acoustical model and the prior probability $p(w)$ is provided by a language model. The set of utterances to recognize in our target application per task is quite limited and very structured. It can thus appropriately be defined by a grammar. Consequently, a language model based on a finite state grammar (FSG) seems most suitable. Even though we assume our CSD already filtered out some undesired input, we are confronted with a high rate of false positive recognitions of *out-of-grammar* (OOG) utterances which we cannot afford and have to take care of. Given an OOG utterance x, a restricted FSG-based decoder cannot come around to hypothesize x as an *in-grammar* (IG) sentence w (or prefixes of it), since the word-sequence probability for all other sentences $p(w')$ is 0 because they are not part of the grammar. This holds even if we suppose the acoustical probability $p(x|w)$ for an IG-sentence to be low. The acoustical probability mainly plays a decisive role for discrimination between different utterances w from within the language model. A TriGram-based language model contains many more possible utterances, hence a decoder using such a language model can also hypothesize those other sentences when it is given an utterance that is OOG (with respect to the FSG). Unfortunately, it cannot provide us with an accurate *best hypothesis* reliably enough. That is, the correct sentence w_x for a given IG-utterance x will not be the best hypothesis often enough. Otherwise, we could just stay with a single TriGram-based decoder for recognition. But we can utilize a TriGram language model to help rejecting OOG utterances hypothesized by the FSG-based decoder. For this the TriGram has to comprise a larger vocabulary than the specific grammar and provide not-too-low probabilities for appropriate combinations of these words, i.e. for OOG sentences.

Let us consider a false positive recognition where an OOG-utterance x is falsely recognized as an IG-sentence w by the FSG-based decoder. With an appropriate modeling of the TriGram we can assume it to provide OOG-sentences w' with significant probabilities within its language-model. We can also assume that the acoustical probability $p(x|w')$ for those w' corresponding to the actual utterance exceed the acoustical probability of each falsely hypothesized IG-sentence w considerably. So for the TriGram-based decoding process the comparatively low acoustical probability $p(x|w)$ causes some words of w to be pruned away around the corresponding time-frames they were hypothesized at by the FSG-decoder. Hence, the word-graph and thus the N-best list produced by the TriGram-based decoder will not contain the sequence w. On the other hand, given an IG utterance x and its sentence w_x, the comparatively high acoustical probability $p(x|w)$ (in combination with a still sufficiently high language probability) likely prevents that words of w_x are pruned in the decoding process. Therefore, the N-best-list will still contain w_x.

Consequently, we accept the hypothesis of the FSG-based decoder, if it can be matched with some hypothesis within the N-best list of the TriGram decoder. To not compare utterances from different instances of time, the matching also takes word-start-frames into consideration.

4.1 Hypothesis matching

For the matching of the FSG-best hypothesis w_{FSG} with one of the N-best hypotheses of the TriGram w_n, we require that the words of w_{FSG} occur in the same order in w_n. The difference in the start-frames of the matched words shall not exceed a predefined maximal offset. For this, we simply iterate through the word-sequence w_n. If the current word of w_n matches with the current word of w_{FSG} (considering the maximal offset allowed), we proceed to the next word of w_{FSG}. If all words of w_{FSG} are processed, we accept the FSG's hypothesis. Within this matching, we always omit hypothesized filler-words like *SILENCE*. For some cases, we experienced that an additional heuristics can improve the acceptance rate. As so, for longer word-sequences w_{FSG} hypothesized by the FSG-based decoder it can be reasonable not to require that all words in w_{FSG} have to be matched on the N-best hypothesis compared to. There is a trade-off between good acceptance rate and good rejection rate when relaxing the matching. Since we only want to make sure that the FSG's hypothesis is not a false positive we argue that enough evidence is given if the FSG's and the TriGram's hypotheses have been similar enough. Therefore, we allow to skip some words of w_{FSG} during the matching dependent on the number of words of w_{FSG} (e.g. in our application we allow to skip 1 word if $|w_{FSG}| \geq 4$, skip 2 words if $|w_{FSG}| \geq 6$ ...). This can be incorporated very easily in our matching procedure we explained above.

5 Experimental Evaluation

To evaluate our approach we conducted several experiments on speech input recorded in the ROBOCUP@HOME environment during a competition. We use a freely available speaker independent acoustic-model for the SPHINX 3 speech engine build with the WSJ-corpus [10]. The FSG decoder was run with the specific grammar for the navigation task shown in Table 1.

The performance of the our dual-decoder systems is influenced by several parameters. We adjusted these in such a way, that the trade-off between higher acceptance-rate of IG-utterances and higher rejection-rates of OOG-utterances tends to a higher acceptance rate. That is because we expect to let pass a fairly low amount of OOG speech-like utterances in the close-speech detection step already.

command	=	[salut] instruct TO THE location | STOP
salut	=	ROBOT [PLEASE]
instruct	=	GO | NAVIGATE | DRIVE | GUIDE ME
location	=	ARM CHAIR | PALM [TREE] | WASTE (BASKET | BIN) | TRASH CAN | UPLIGHT | REFRIGERATOR | FRIDGE | COUCH | SOFA | PLANT | BOOKSHELF | SHELF | (COUCH | SIDE | COFFEE | DINNER | DINNING) TABLE | [FRONT] DOOR | LAMP

Table 1: Grammar for the navigation task

(a) Dual decoder

	rejected	accepted	
recognized	8.6%	77.6%	86.2%
falsely recognized	3.6%	10.2%	13.8%
	12.2%	87.8%	

(b) error rates and real-time factors (RTF) for single decoders

	WER	SER	RTF
TriGram-based	9.9%	30.7%	0.99
FSG-based	4.1%	13.8%	0.24

Table 2: Accuracy and rejection results of dual decoder for legal commands

5.1 Recognition Accuracy

To assess the overall recognition performance of our dual decoder system compared to a TriGram-only and an FSG-only system, we compiled a set of utterances that are legal commands of a particular task in the RoboCup@Home domain. The FSG decoder is using the corresponding grammar of this specific task. The TriGram decoder in our dual decoder system uses a language model constructed from all tasks (excluding the task used for evaluating the rejection of OOG-utterances) of the RoboCup@Home domain with an additional set of 100 sentences of general purpose English. To achieve best performance the TriGram-only decoder uses a language model constructed from navigation sentences only.

In our particular evaluation setup we fed 723 (20.6 minutes) correct commands from the navigation task (cf. Table 1) to all three decoders. Table 2 shows the accuracy and rejection results. For our dual-decoder, we consider an utterance successful if it is recognized correctly and accepted. The recognition rates are based on the FSG decoder while the rejection rates are based on the matching between the FSG's hypothesis and the first 25 entries of the TriGram's N-best list. In the TriGram-only case, we take the best hypothesis as the recognition output. The results indicate that using two decoders in parallel yields successful processing. Adding up 13.8% of falsely recognized commands and 8.6% of correctly recognized but rejected commands, we receive a total of 22.4% of unsuccessfully processed utterances in comparison to 30.7% of an TriGram-based decoder. The sentence-error rate (SER) is a more meaningful measure than the word-error rate (WER) here, because we are interested in the amount of sentences containing errors and not in the number of errors per sentence. The 10.2% of falsely recognized but accepted utterances are critical in the sense that they could have caused a false command to be interpreted. Depending on the application, hypotheses that differ from the reference spoken can still result in the same command, e.g. ”ROBOT PLEASE GO TO THE REFRIGERATOR” yields the same command as “DRIVE TO FRIDGE”. To give an idea about possible proportions, for the dual-decoder on our navigation task half of the potentially critical utterances (10.2%) are matched on the same command. For the TriGram-decoder this is the case for one fifth of the 30.7% of all sentence errors. 24.2% (overall) are OOG-sentences and thus are not matched to commands at all. To compare

Decoder	$\mathrm{FP}_{accepted}$	Error rate on correct commands
Single (FSG only)	93.9%	13.8% (SER)
Single (TriGram only)	16.1%	30.7% (SER)
Dual (FSG+TriGram)	17.7%	13.8% (SER) + 8.6% (falsely rejected)

Table 3: Acceptance rates of false positive (FP) utterances and error rates on legal commands

the processing speed, we also measured the real-time factor (RTF), i.e. the time it takes to process a signal of duration 1. We achieved an RTF of 1.16 for our dual-decoder system on a Pentium M with 1.6 GHz. This is fast enough for our application, since we are given closed utterances by our CSD front-end and we only decode those. RTFs for the single-decoder systems (with relaxed pruning thresholds for best accuracy) on the same machine are listed in Table 2(b).

5.2 Rejection Accuracy

To assess the performance of our dual decoder system in rejecting OOG utterances (with respect to the FSG) we collected a set of utterances that are legitimate commands of the RoboCup@Home domain (all tasks) but that do not belong to the specific task the FSG decoder is using. Please note that this is close to a worst case analysis since not all of the utterances that make it to the decoder stage in a real setup will be legal commands at all. In our particular case we took 1824 commands (44 minutes) from the final demonstration task and the manipulation task and fed those commands to an FSG-only system, a TriGram-only system, and our proposed dual decoder system. All three decoders had the same configuration as in the recognition setup above. The FSG decoder for the single case and within our dual-decoder system was using the navigation task grammar (cf. Table 1). As can be seen in Table 3, the single FSG decoder setup would have matched over 93% of the false positive utterances to valid robot commands. With our dual decoder approach, on the other hand, the system was able to reject more than 82% of those false utterances. With a TriGram-only decoder we would have been able to reject 84%, but this would have come with a prohibitive error rate of more than 30% for correct commands as shown in Table 3 and Table 2(b) already.

6 Conclusion

In this paper, we presented an architecture for a robust speech recognition system for service-robotics applications. We used off-the-shelf statistical speech recognition technology and combined two decoders with different language models to filter out false recognitions which we cannot afford for a reliable system to instruct a robot. The advantages of our system in comparison to more sophisticated approaches mentioned are as follows. It provides sufficiently accurate speech detection results as a front-end for ASR-systems. Our approach is computationally efficient and relatively simple to implement without deeper knowledge about speech recognition interiors and sophisticated classifiers like HMMs, GMMs or LDA. It is therefore valuable for groups lacking background knowledge in speech recognition and aiming for a robust speech recognition system in restricted domains.

As results are very promising so far, a future issue would be to examine the system's performance for far-field speech, that is not using a headset. We imagine this to be worthwhile especially when we integrate filter methods such as beam forming for on-board microphones with sound-source localization [11, 12].

Acknowledgment

This work was supported by the German National Science Foundation (DFG) in the Graduate School 643 *Software for Mobile Communication Systems* and partly within the Priority Program 1125. Further, we thank the reviewers for their comments.

References

1. van der Zant, T., Wisspeintner, T.: Robocup x: A proposal for a new league where robocup goes real world. In Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y., eds.: RoboCup. Volume 4020 of Lecture Notes in Computer Science., Springer (2005) 166–172
2. Huang, X., Alleva, F., Hon, H.W., Hwang, M.Y., Rosenfeld, R.: The SPHINX-II speech recognition system: an overview. Computer Speech and Language **7**(2) (1993) 137–148
3. Lamel, L., Rabiner, L., Rosenberg, A., Wilpon, J.: An improved endpoint detector for isolated word recognition. IEEE Transactions on Acoustics, Speech, and Signal Processing [see also IEEE Transactions on Signal Processing] **29**(4) (Aug 1981) 777–785
4. Macho, D., Padrell, J., Abad, A., Nadeu, C., Hernando, J., McDonough, J., Wolfel, M., Klee, U., Omologo, M., Brutti, A., Svaizer, P., Potamianos, G., Chu, S.: Automatic speech activity detection, source localization, and speech recognition on the chil seminar corpus. IEEE Int. Conf. on Multimedia and Expo, 2005 (ICME 2005) (6-6 July 2005) 876–879
5. Padrell, J., Macho, D., Nadeu, C.: Robust speech activity detection using lda applied to ff parameters. Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '05) **1** (March 18-23, 2005) 557–560
6. Rentzeperis, E., Stergiou, A., Boukis, C., Souretis, G., Pnevmatikakis, A., Polymenakos, L.: An Adaptive Speech Activity Detector Based on Signal Energy and LDA. In: 3rd Joint Workshop on Multi-Modal Interaction and Related Machine Learning Algorithms. (2006)
7. Ruhi Sarikaya, J.H.L.H.: Robust Speech Activity Detection in the Presence of Noise. In: Proc. of the 5th Int. Conf. on Spoken Language Processing, (Inc. the 7th Australian Int. Speech Science and Technology Conf. (1998)
8. Lin, Q., Lubensky, D., Picheny, M., Rao, P.S.: Key-phrase spotting using an integrated language model of n-grams and finite-state grammar. In: Proc. of the 5th European Conference on Speech Communication and Technology (EUROSPEECH-1997). (1997) 255–258
9. Wessel, F., Schlüter, R., Macherey, K., Ney, H.: Confidence measures for large vocabulary continuous speech recognition. IEEE Transactions on Speech and Audio Processing **9**(3) (Mar 2001) 288–298
10. Seymore, K., Chen, S., Doh, S., Eskenazi, M., Gouvea, E., Raj, B., Ravishankar, M., Rosenfeld, R., Siegler, M., Stern, R., Thayer, E.: The 1997 CMU Sphinx-3 English Broadcast News transcription system. In: Proc. of the DARPA Speech Recognition Workshop. (1998)
11. Calmes, L., Lakemeyer, G., Wagner, H.: Azimuthal sound localization using coincidence of timing across frequency on a robotic platform. Journal of the Acoustical Society of America **121**(4) (2007) 2034–2048
12. Calmes, L., Wagner, H., Schiffer, S., Lakemeyer, G.: Combining sound localization and laser based object recognition. In Tapus, A., Michalowski, M., Sabanovic, S., eds.: Papers from the AAAI Spring Symposium, Stanford CA, AAAI Press (2007) 1–6