Speech Functionality Benchmark Final Specifications

September 23, 2014

1 Specifications

This document reports the final version of the specifications for the Speech Functionality Benchmark that will be released to the participants. It will comprises:

- information about the organization of the Functionality Benchmark
- resources that will be released before the competitions to the participants, that can be used to help in modeling or accomplish the required task.
- description of expected formats of different inputs and outputs, that must be strictly respected in order to correctly perform for the evaluation phase.

1.1 Evaluation Overview

The evaluation phase of the Speech Understanding Functional Benchmark will take place in two steps:

- a phase where the commands to be recognized and interpreted will be uttered directly from a member of the Organizing Committee. The robot should be able to acquire audio inputs coming from a loud-speaker and process them;
- a phase where the audio corresponding to the commands to be processed, will be provided in a file format (see Section 1.2) on a USB stick. For this case, the system should be able to process directly audio files read from the USB stick, bypassing the audio acquisition system.

In both cases, a USB stick will be provided to the teams and the results of the processing must be written on it, according to the format described in Section 1.4 and 1.3.

The benchmarking phase will last 20 minutes. All the teams that will deliver their USB stick after 20 minutes from the beginning, will receive a penalty on the final score.

NOTE: the two phases are both optional, meaning that the evaluation could proceed by performing just one of the two, or them both, at discretion of the OC. For this reason, the processing systems must be able to deal with both phases in order to be evaluated.

NOTE 2: in the case the two phases are both performed, all the data requested for the evaluation must be provided in the USB stick at the end of the process.

NOTE 3: in the case the two phases are both performed, the final evaluation will be executed separately. Results of the two phases will receive different weights, in order to equally balance their contributes in the overall evaluation.

NOTE 4: the number of commands and the relative audio files for the two phases, reported in the following subsections, can vary according to the operational condition of the testbed environment during the set up days.

1.1.1 Evaluation of audio input acquired directly through microphone

For this part of the Functionality Benchmark, the process will consist of these steps:

- Every team will receive a USB stick.
- After a proper communication, a member of the Organizing Committee will pronounce 5 commands using a microphone. The audio will be instantly reproduced using a loud-speaker, conveniently positioned to be equally distant from each robot involved in the benchmark. Each command will be given after an interval of about 15 seconds of silence from the previous one. **NOTE:** The teams will be allowed to use an interface, composed by a single button, to be used to indicate the start of the recording of each uttered command.
- Each team must produce an output file following the format described in Section 1.4. This file must contain the result of the Speech Understanding analysis of the uttered commands, as described in Section 1.3, and must be written on the USB stick.
- Each team must also write on the USB stick the audio files corresponding to the audio of the uttered commands. **NOTE:** For each missing audio file among these, the corresponding line in the output file will not be considered during the final evaluation. A referee will control the consistency of each audio file acquired during this phase, in order to check if they correspond to what has been uttered by the OC member. Any file not corresponding to the uttered command (e.g. containing white noise or only a portion of the uttered command), will be considered as not valid. **NOTE 2:** the audio files must respect the specs described in Section 1.2.
- The USB stick will be delivered to the OC, and the evaluation phase will start. Final results will be given after the designated referee has checked the consistency of all the acquired audio files.

1.1.2 Evaluation of pre-recorded audio files

The evaluation of the Functionality Benchmark starting from the audio files will consist of these steps:

- Every team will receive a USB stick containing approximately 100 audio files representing uttered commands. These audio files will follow the specs described in Section 1.2.
- Each team must produce an output file following the format described in Section 1.4. This file must contain the result of the Speech Understanding analysis of the (approximately) 100 audio files, as described in Section 1.3, and must be written on the USB stick.
- The USB stick will be delivered to the OC, and the evaluation phase will start.

1.2 Audio Input and Output Format

The audio files that will be provided to the teams and the audio file produced by the teams (see Section 1.1.1) must follow a precise audio format. They must be stereo (2 channels) `.WAV` files, encoded with a 44100 Hz sampling rate.

1.3 Semantic Frames Description and Definition

In order to evaluate the correct understanding of a command expressed in natural language (e.g. through a sentence), a semantic representation formalism based on *semantic frame* has been selected. Each frame corresponds to an action, namely a robot command. A set of arguments is associated to each frame, specifying part of the command playing a particular role with respect to the action expressed by the frame. For example, in the command "*go to the dining room*" the *Motion* frame is expressed by the verb *go*, while the part of the sentence "*to the dining room*" corresponds to the GOAL argument, indicating the destination of the *Motion* action. The set of frames defined and selected for this Benchmark is reported in the following list, together with the set of associated arguments (current arguments are reported in bold italic inside the example commands):

- *Motion*: the action performed by the robot itself of moving from one position to another, occasionally specifying a specific path followed during the motion. The starting point is always taken as the current position of the robot.
  - GOAL: the final position in the space to be occupied at the end of the motion action, e.g. "*Go **to the kitchen***"; "*Go **to the right of the sofa***".
  - PATH: the trajectory followed while performing the motion towards the GOAL, e.g. "*Move **along the wall***".

- *Searching*: the action of inspecting an environment or a general location, with the aim of finding a specific entity.
  - THEME: the entity (most of the time an object) to be searched during the searching action, e.g. "*Search **for the glass***".
  - GROUND: the environment or the general location in the space where to search for the THEME, e.g. "*Find the glass **in the living room***".
- *Taking*: the action of removing an entity from one place, so that the entity is in robot possession.
  - THEME: the entity (typically an object) taken through the action, e.g. "*Take **the cereal box***".
  - SOURCE: the location occupied by the THEME before the action is performed and from which the THEME is removed, e.g. "*Grab the mayo **on the table***"; "*Remove the sheets **from the bed***".
- *Placing*: the action of placing an entity that the robot already posses in a place or position in the space.
  - THEME: the entity (typically an object) placed through the action, e.g. "*Drop **the jar***".
  - GOAL: the location that should be occupied by the THEME after the action is performed, e.g. "*Put the can **on the counter***".
- *Bringing*: the action of changing the position of an entity in the space from a location to another.
  - THEME: the entity (typically an object), being carried during the bringing action, e.g. "*Bring **the garbage** to the kitchen*".
  - GOAL: the endpoint of the path along which the carrier (e.g. the robot - and thus the THEME) travels, e.g. "*Bring the garbage **to the kitchen***".
  - SOURCE: the beginning of the path along which the carrier (e.g. the robot - and thus the THEME) travels, e.g. "*Bring the garbage **from the dining room** to the kitchen*".
  - BENEFICIARY: the person to whom the THEME must be brought, e.g. "*Bring **me** the mobile phone*".

1.3.1 Command Frame Representation

In order to represent the semantic frames extracted from the commands in a compact way, a specific syntax has been defined, called *Command Frame Representation* (CFR). Such a representation will have to respect a command/arguments structure, resembling the common syntax for a programming language method: the frame represents the method name, while the arguments represent the method arguments. For example, for the "*go to the dining room*" command,

where a *Motion* frame is expressed and the GOAL argument is instantiated with "*to the dining room*", the corresponding CFR will be:

```
MOTION(goal:‘‘to the dining room’’).
```

It is worth underlying that more than one argument can be expressed in a command.

Results of the Speech Understanding Functionality must be presented according to the CFR formalism. The grammar specifying the correct syntax for a CFR will be also provided, and is reported in the following.

```
Command → Single_command | Composed_command

Composed_command → Single_command#Command

Single_command → Action(Arguments)

Action → MOTION | TAKING | BRINGING | SEARCHING | PLACING

Arguments → Argument | Argument, Arguments

Argument → Argument_name:‘‘Role_filler’’

Argument_name → theme | goal | source | path | ground | beneficiary

Role_filler → Defined_lexicon
```

where `Defined_lexicon` is the lexicon that will be released to the team before the competition, including names of rooms (e.g. *hallway*, *living room*, etc.) and objects (e.g. *cereal box*, *jar*, etc.), see Section 1.5.3.

Composition of actions is also possible in the CFR, corresponding to more complex action as the *Pick_and_place* action, represented by a sequence of *Taking* frame followed by a *Placing* frame (e.g. for the command "*take the box and put it on the table*"). The corresponding CFR will be:

```
TAKING(theme:‘‘the box’’)#PLACING(theme:‘‘it’’,goal:‘‘on the table’’)
```

1.4 Speech Understanding Output File Format

For each command uttered or for each audio file directly provided during the Speech Understanding Functional Benchmark, the system should generate the corresponding transcription and the interpretation in the CFR format. This information have to be saved in an output text file called `results.txt`. In this file, a line has to be added for each command or audio file, following the format

$$\texttt{audio\_file\_name}_i\texttt{|command\_transcription}_i\texttt{|CFR}_i$$

where `audio_file_name`, `command_transcription` and `CFR` represent respectively the name, the transcription and the interpretation of the i-th audio file, separated by a pipe (`|`). The `results.txt` file must be encoded using standard

UTF-8 character encoding: files with a different encoding will be automatically rejected.

NOTE: the correspondence between the data in the results file and the audio file names must be consistent. Audio files corresponding to commands acquired through the microphone system of the robot must be named according to the following syntax

`fb_mic_phase_speech_audio_`i`.wav`,

representing the name of the i-th audio file of the i-th uttered command. In the case of audio files provided directly to the robotic platform without passing from the audio system, thus on a USB stick, the name in this field must be the same of the audio file analyzed.

If the Automatic Speech Recognition fails in transcribing a sentence, the label `BAD_RECOGNITION` must be used. Similarly, if the semantic frame extraction does not produce any correct result, the `NO_INTERPRETATION` label must be reported. An example of `results.txt` file is reported in the following:

```
fb_mic_phase_speech_audio_1.wav|move to the living room|MOTION(goal:‘‘living room’’)
fb_mic_phase_speech_audio_2.wav|BAD_RECOGNITION|NO_INTERPRETATION
fb_mic_phase_speech_audio_3.wav|take the jar on the table|TAKING(theme:‘‘the jar’’, source:‘‘the table’’)
...
```

NOTE: every line not respecting the format required and described above will be skipped during the evaluation phase.

NOTE 2: if both the evaluation phases are performed, both results must be written inside the same `results.txt` files. The audio file name associated to each line will define the belonging of the analysis outcome to one phase or the other.

1.4.1 Annotation Examples

Here we provide some annotation examples. Considering the command "*go to the kitchen and take the coffee mug*", where two frames are evoked (e.g. *Motion*, the two associated annotation will be:

$$[go]_{Motion}\ [to\ the\ kitchen]_{\textsc{Goal}}\ and\ take\ the\ tea\ cup$$

$$go\ to\ the\ kitchen\ and\ [take]_{Taking}\ [the\ tea\ cup]_{\textsc{Theme}}.$$

The frame are associated to the verbs as they evoke the action expressed by the frames. This double tagging of the semantic information will correspond to the conjunction of the CFR of the two frames, that is:

```
MOTION(goal:‘‘to the kitchen’’)#TAKING(theme:‘‘the tea cup’’).
```

Frames can have more than one single arguments, as for the command "*take a towel from the bathroom*", corresponding to the following tagging and CFR:

$$[take]_{Taking}\ [a\ towel]_{\textsc{Theme}}\ [from\ the\ bathroom]_{\textsc{Source}}$$

```
TAKING(theme:‘‘a towel’’,source:‘‘from the bathroom’’).
```

In a sentence, it will be possible to find some arguments that have not been tagged. These will be only arguments not defined for this task (and thus not reported in the list in Section 1.3) as, for example, the MANNER argument, representing the manner in which the action take place. For example, in the command "*Search carefully the bedroom for my wristwatch*", the adverb *carefully*, representing the MANNER, is not tagged:

$$[search]_{Searching}\ carefully\ [the\ bedroom]_{\textsc{Ground}}\ [for\ my\ wristwatch]_{\textsc{Theme}}$$

```
SEARCHING(ground:‘‘the bedroom’’,theme:‘‘for my wristwatch’’).
```

Finally, some commands have been enriched with colloquial forms, as the use of modal verbs, e.g. "*could you please find my jacket?*". These particles are not considered in the tagging, as they represent only inflections that, in this case, don't affect the general meaning of the command:

$$could\ you\ please\ [find]_{Searching}\ [my\ jacket]_{\textsc{Theme}}\ ?$$

```
SEARCHING(theme:‘‘my jacket’’).
```

1.5 Available Resources

Some resources are already available or will be released before the competition to help in developing modules for Speech Understanding or to validate the output required by the competition.

1.5.1 The Robocup@Home Corpus

The Robocup@Home dataset can be used as a benchmark for evaluate a SU system, and is available on line at `http://sag.art.uniroma2.it/HuRIC.html`. It is composed by a set of spoken commands for house servicing robots, paired with their correct transcriptions and provided with syntactic and semantic information, e.g. *semantic frames*. Spoken commands are provided as a set of audio files respecting the format selected for the competition (see Section 1.2).

The Robocup corpus is only one of three datasets composing the *Human-Robot Interaction Corpus* (HuRIC). The other two datasets, namely the Grammar Generated dataset and the S4R Experiment dataset, can be used as well for benchmarking purposes. They present the same exact source of information given by the Robocup corpus, and are provided together with it. **NOTE:** the audio format of the audio file of these two datasets could be different from the one needed for the competition. The main difference among the three datasets is that the syntax of the commands contained in the Grammar Generated and the S4R Experiment is more simple and structured with respect to the one in the Robocup corpus.

In general, all the data provided by the three dataset, thus by the whole HuRIC, can be easily used to develop, test and validate the SU system for this Functionality Benchmark. Moreover, the provided examples can be used to better understand the nature of the different frame arguments (called *frame elements* in the corpus), and correctly tag them.

1.5.2 CFR Validation script

A Python script will be released to validate the CFR format produced by the SU analysis, called `parser.py`. A class called `RockinParser` implements the grammar defined in Section 1.3.1, and thus can recognized correct CFR forms. The following code snippet reports an example of how to use the parser inside a Python script:

```
1 from parser import RockinParser
2
3 test = "MOTION(goal:\"to the kitchen\", source:\"from the bedroom\")"
4 parser = RockinParser()
5 try:
6   parser.parse(test)
7 except Exception, e:
8   print "Cannot parse the input string"
9   print e
```

The method `parse` of the `RockinParser` class performs the validation check on the `test` string passed as input, according to the grammar defined in the `grammar.py` file, provided together with the `parser.py`. **NOTE:** the parser class is defined in the `parser.py` file, and this uses the `grammar.py` file in turn. For this reason, these two files must be in the same directory of your Python script (or in any other directory accessible by the Python `import` statement).

1.5.3 Defined Lexicon

The whole lexicon in the commands will be provided to the teams before the competition. It will consist of six text files, namely:

- `verbs`: contains all the verbs used in the commands, comprehending modal and auxiliary verbs (e.g. "*can*").
- `nouns`: contains all the nouns present in the commands.
- `prepositions_and_positional_adverbs`: contains all the prepositions and positional adverbs (e.g. "*close*") used in the commands.
- `personal_pronouns`: contains the list of the personal pronouns used in the commands (e.g. "*me*").
- `adjectives`: contains the adjectives used in the commands, considering also the one derived from verbs (e.g. "*dining*" for "*dining room*").
- `others`: contains all the words that are not considered in the previous categories, as articles or other kind of adverbs (e.g. "*carefully*").

Each file will be a `txt` containing the list of words separated by a newline, encoded using standard UTF-8 character encoding. For example:

```
1 go
2 move
3 bring
4 place
5 search
6 find
7 ...
```

NOTE: these files report all the lexicon that could be used in the commands. No words outside this lexicon will be used. On the other hand, only one subset of it could be used in the commands, meaning that not all of them must be present in the commands.