Integration of Sound Source Localization and Separation to Improve Dialogue Management on a Robot

Maxime Fréchette[1], Dominic Létourneau[1], Jean-Marc Valin[2] and François Michaud[1]

***Abstract*— To demonstrate the influence of an artificial audition system on speech recognition and dialogue management for a robot, this paper presents a case study involving soft coupling of ManyEars, a sound source localization, tracking and separation system, with the CSLU Dialogue Management system. Trials were conducted in a laboratory and a cafeteria. Results indicate that preprocessing of the audio signals by ManyEars improves speech recognition and dialogue management of the system, demonstrating the feasibility and the added flexibility provided by ManyEars for a robot to interact vocally with humans in a wide variety of contexts.**

I. INTRODUCTION

Giving robots the ability to process natural language comes with great challenges, as they have to operate in changing and diverse conditions. Natural language systems usually process audio streams recorded from one microphone using three main components: 1) a speech recognition module (e.g., Sphinx [1], NUANCE); 2) a dialogue manager (e.g., COLLAGEN [2], MIT's Galaxy Communicator [3]); 3) a text-to-speech synthetiser (e.g., Festival [4], Gnuspeech). These systems usually assume that speech is acquired from a microphone located close to the interlocutor (usually attached on a headset) to get clear audio streams. However, this assumption is not valid for a mobile robot operating in open settings, interacting with multiple people and in different contexts.

Recently, a sound source localization, tracking and separation system called ManyEars [5], [6], [7], [8] has been released [9], and is also used by Kyoto University's HARK system [10], [11]. It consists of an array of eight microphones placed on the robot's body. The localization and tracking algorithm is based on a frequency-domain implementation of a steered beamformer along with a particle filter-based tracking algorithm. Results show that a mobile robot can localize and track in real-time up to four moving sources of different types, over a range of 7 meters. Sound source separation is accomplished with a real-time implementation of geometric source separation (GSS) and a postfilter that gives a further reduction of interference from other sources. Compared to using one microphone, ManyEars improves word recognition of simultaneous speech in controlled conditions (using recordings and three loudspeakers), going from a 10% recognition rate, to 25% (with a 10° angle between the center speaker and the side loudspeaker, which is more difficult to separate because of the proximity of the sources) and up to 72% (with a 90° angle between the center speaker and the side loudspeaker) recognition rate [6].

Up to now, ManyEars has been used mostly for localizing sound sources and as a pre-processing module for speech recognition of words, and has not yet been integrated with a natural language processing system. To evaluate if ManyEars can improve performances for speech recognition and dialogue management of a robot operating in natural settings, we decided to conduct a case study integrating ManyEars to the CSLU (Center for Spoken Language Understanding) Toolkit [12], [13], a dialogue management system. We chose CSLU because it is complete system and it offers interesting features, such as:

- the graphical tool Rapid Application Developer (RAD) [14], used to easily create dialogue scenarios;
- more accurate and more realistic voice interactions by easily and quickly change voices, pitches and rates [12];
- tokens that can be added before and after valid grammar strings, to add flexibility in recognizable speech patterns;
- the ability to dynamically change grammars used for speech recognition. This is an important feature because it allows the system to add or remove words in the system's lexicon according to the interaction context (which depends on the robot's current task), optimizing processing time and making it possible to adapt to the robot's intention;
- a dialogue repair tool used when the result of speech recognition does not meet a predefined threshold of efficiency. In our case, the process consists of conducting a second iteration of speech recognition on the audio stream with a more flexible grammar, by adding garbage collectors around the grammar itself and by dynamically changing the out of vocabulary rejection and word spotting medians, allowing the recognizer to be more permissive during its second pass [14];
- an optimized version of the University of Edinburgh's Festival [4] package for text-to-speech synthesis.

This paper presents how ManyEars can be used as a preprocessing module for CSLU on a mobile robot, and analyzes the results obtained from having people interact vocally using complete sentences (and not just words) with

*This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), the Fonds Québécois de la Recherche sur la Nature et les Technologies (FQRNT), and the Canada Research Chair in Mobile Robotics and Autonomous Intelligent Systems.

[1]M. Fréchette, D. Létourneau and F. Michaud are with the Département de génie électrique et informatique at Université de Sherbrooke, Québec, Canada, {maxime.frechette, dominic.letourneau, francois.michaud} at usherbrooke.ca

[2]J.-M. Valin is with Mozilla inc. jmvalin at jmvalin.ca

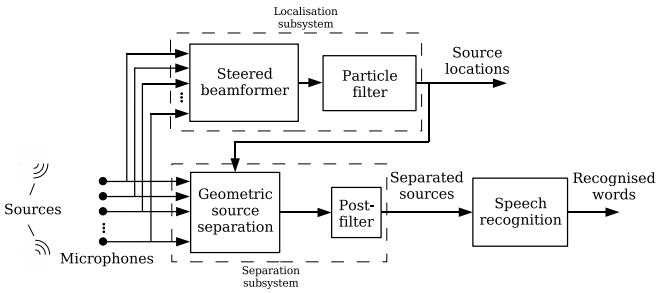

Fig. 1. ManyEars' architecture.

a robot, in laboratory conditions and in a cafeteria. The paper is organized as follows. Section II presents a brief overview of ManyEars [5], [6], [7], [8]. Section III presents the approach used to integrate ManyEars and the CSLU toolkit. Section IV follows with a description of the trials and results, and Section V concludes with a discussion on the observed performance and future work.

II. MANYEARS

ManyEars[1], also known as AUDIBLE, is illustrated in Fig. 1. ManyEars is composed of a sound source localization subsystem that detects, localizes and tracks sound sources in the environment, and a sound source separation subsystem that uses the localization information to separate each source. The sampling rate used in the original system is 48 kHz (16 bits/sample). Speech recognition is not done by the system itself, but occurs at a subsequent stage. More specifically, ManyEars acts as a pre-processing module that provides sound source localization information and separated audio streams to be processed by other decisional modules.

A. Sound Source Localization

The sound source localization subsystem consists of an initial localization step based on the steered response power algorithm and a tracking step that is performed using particle filtering. For the steered response power algorithm, the source direction is initially searched on a 2562-point spherical grid using a lookup table that returns the time delay of arrival (TDOA) between microphones i and j for the searched direction d, and $R_{i,j}$, the relevance-weighted phase transform (RWPHAT) [5]. The search process is repeated to find a preset number of sources (i.e., four), which leads to false detections when fewer sources are present. The number of sources to simultaneously locate was set empirically to optimize computation time and performance. The search is based on the far-field assumption (large distance to the array) with a grid that provides a maximum error of 2.5° (best case), which corresponds to the radius covered by each of the 2562 regions around its centre.

It is however possible to improve the resolution by performing a refined search, constrained to the neighborhood of the first result found. In this second search, we can include the distance. While this distance estimate is not reliable enough to be useful, it helps improve the direction accuracy. In addition to the refining stage, most floor reflections can be eliminated by having the search exploit the fact that a reflection always has the same azimuth as the direct path, but with a higher absolute elevation.

The direction information found by the steered beamformer contains a large number of false positives and false negatives. Moreover, the source directions found are instantaneous (or memoryless), and it is thus not possible to keep track of sources over time, especially when there are gaps in the localization data for a source. This justifies the role of the particle filtering stage. The choice of particle filtering is motivated by the fact that taking into account false positives and false negatives makes error statistics depart significantly from the Gaussian model. Each source being tracked is assigned a particle filter and each observed direction is assigned to a tracked source using a probabilistic model [5]. By using the simple sample importance resampling (SIR) algorithm, it is possible to use 1000 particles per source while maintaining a reasonable complexity.

B. Sound Source Separation

The sound source separation subsystem is also composed of a linear sound source separation algorithm, followed by a non-linear post-filter. The initial linear source separation is achieved using a variant of the Geometric Source Separation (GSS) algorithm [15] that operates in real-time and with reduced complexity [16].

The GSS algorithm alone cannot completely attenuate the noise and interference from other sources, so a multi-source post-filter is used to improve the signals of interest. The post-filter is based on the short-term spectral amplitude estimator approach originally proposed by Ephraim and Malah [17]. Unlike the classical algorithm, the noise estimate used is the sum of two terms: stationary background noise and interference from other sources. The interference term is computed by assuming a constant leakage from the other sources [18], [19].

III. INTEGRATION OF MANYEARS AND THE CSLU TOOLKIT

Figure 2 illustrates the architecture of the natural language processing systems implemented using ManyEars and CSLU. Audio streams processed by ManyEars are sent to the CSLU Toolkit for recognition, evaluation and decision. When required, vocal responses are synthesized to audio streams by the CSLU Toolkit and are sent to the audio server for playback on the robot's loudspeakers. The system runs on three computers.

- Computer 1 is equipped with a multi-input sound card and is dedicated to ManyEars' audio processing algorithm and generate the audio streams to process.
- Computer 2 runs the CSLU Toolkit for dialogue management.
- Computer 3 hosts the decision-making processes of the robot, described in details in [20].

[1]manyears.sourceforge.net, available freely with an open source license.

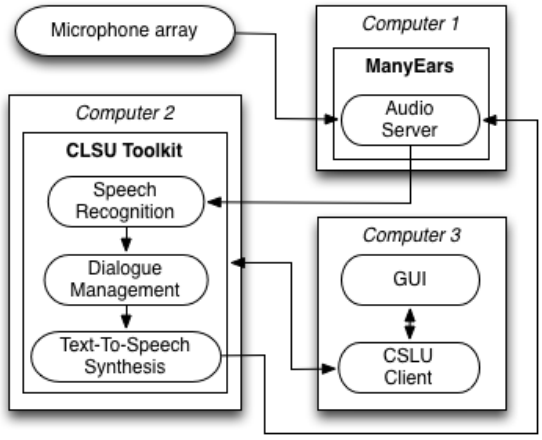

Fig. 2. Natural language processing architecture using ManyEars.

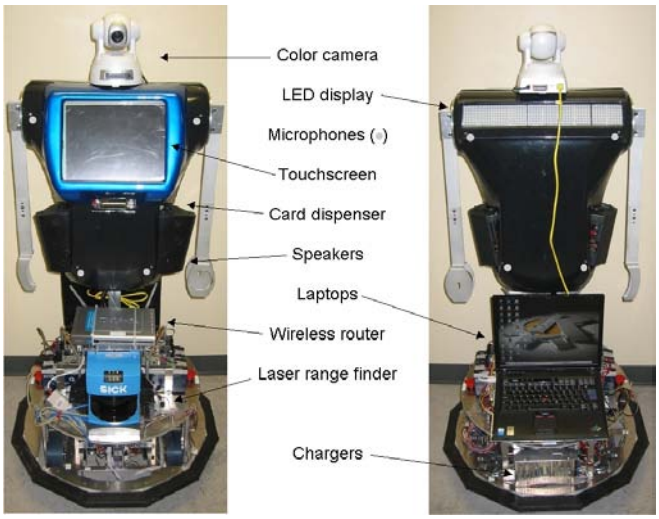

Fig. 3. Spartacus platform used in our trials.

Figure 3 shows the Spartacus robot used in the trials. Our human-robot interaction framework involves having interlocutors talk to the robot to respond to questions or to navigate vocally through graphical user interface (GUI) windows displayed on the robot's touch screen. Components of GUI windows (buttons, fields, widgets, etc.) are added or removed to the grammar (formatted according to a modified version of the W3C Speech Recognition Grammar Specification) as they are made available by the robot to the interlocutors. Dynamic changes to the grammar is done through the CSLU Client module which communicates with the CSLU Toolkit module using a network socket. When a recognition is performed successfully by CSLU's Speech Recognition module, or when a decision is taken by the Dialogue Management module, a message is sent back to the CSLU Client module, which in turn interprets the requests and executes the related tasks on the robot. Figure 4 shows an example of a GUI window as well as a text-to-speech on-screen display.

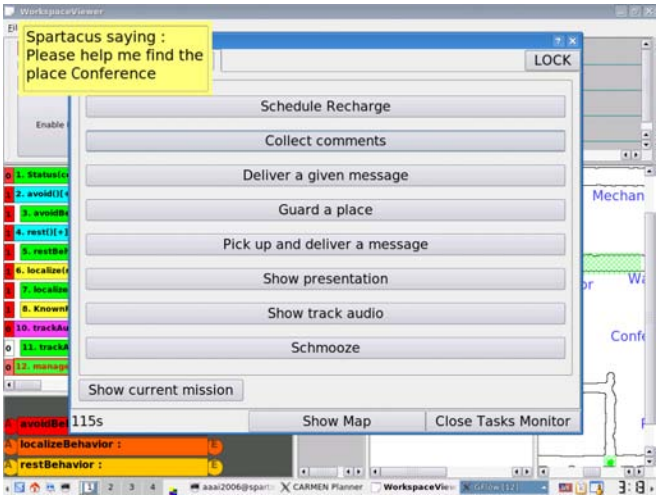

Fig. 4. Robot's tasks manager and text-to-speech on-screen display.

Figure 5 illustrates the flow diagram of the Dialogue Management module. At startup, the system initializes the system's parameters. It then enters an infinite loop, through the state *check_system_status*, which evaluates five conditions:

- *set_values*. This element makes it possible to dynamically load new grammars, as determined by the robot's decision-making processes and communicated through the CSLU Client. Grammars processing and pronunciation extraction for speech processing require important processing time, and the robot has to interact with people in real-time. Therefore, when a new grammar must be loaded into the system, pronunciations are extracted, grammar attributes are set, text-to-speech strings and state change requests are loaded in the associated state block (*recog_speech*, *perform_TTS*, etc.). A confirmation is sent back to the CSLU Client upon receiving valid data. The grammar is copied in memory and saved in a file when the system is stopped. When the CSLU Toolkit is later restarted, it reopens the file, reads all saved values and stores them in memory. This avoids having to process grammars every time the dialogue context changes, decreasing execution time from a few seconds to milliseconds.
- *perform_TTS*. This blocks performs text-to-speech (TTS) synthesis by sending the audio streams to the audio server for playback, and to the CSLU Client to display on the GUI. This occurs when the robot's state changes and before speech recognition can continue, because we want the interlocutor to be aware of the interaction context with the robot. For instance, if the interlocutor selects (either vocally or by touch) a button on the screen, the robot would communicate out loud its new task.
- *get_state*. The robot has different pre-programmed interaction modes: *Trivias*, *jukebox*, *fortune_cookie*, *Schmooze* and *entertain*. These modes are activated by the interlocutor through vocal or touch interaction with

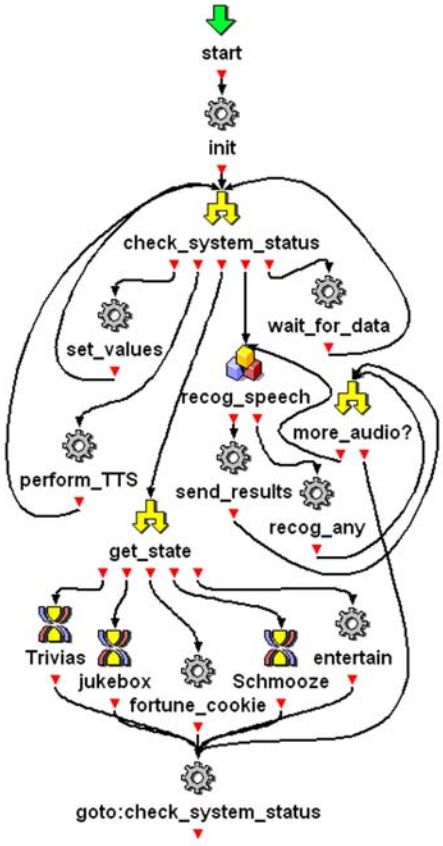

Fig. 5. Flow diagram of the Dialogue Management module, implemented using CLSU RAD tool.

the GUI windows.

- *recog_speech*. When new audio data is received from the audio server, speech recognition is performed using the prevailing grammar. Valid recognitions are sent to the CSLU Client via *send_results* and a corresponding action is taken by the robot. Otherwise, the CSLU dialogue repair is applied (*recog_any*). If repair fails, depending on the repair state, the system checks if there is additional audio streams (*more_audio?*) and continues to perform speech recognition.
- *wait_for_data*. When there is no configuration requests or audio streams to process, the system puts itself to sleep to minimize processing power.

IV. EVALUATION AND RESULTS

Tests were conducted using a convenience sample of 12 healthy students in the Faculty of Engineering of the Université de Sherbrooke. Participants were asked to navigate vocally through the GUI windows, and to provide answers to requests made by the robot. For both type of interactions, participants were asked to formulate complete sentences starting with the robot's name, and then make their request or respond just like they would normally do. For instance,

- In the specific context of interacting through the GUI, as shown in Fig. 4, it was possible for the participant to select the "Schedule Recharge" button by saying "Spartacus, press the Schedule Recharge button please", or to close the Tasks Monitor window by stating "Spartacus, push the Close Tasks Monitor button".
- In the Trivia mode, answering the question "In which city were the 2010 olympic games?" could be done using a complete sentence like "Spartacus, the answer is Vancouver".

Audio streams shorter than 1 sec were discarded by default because they had to be longer to form complete sentences.

Trials were conducted in two environments: in the lab (LAB) and in a cafeteria (CAF). Both environments were not controlled, i.e., our system was used in the natural conditions of these environments, with multiple people talking simultaneously in unconstrained fashion (at least two on average) and noise coming from different sources (chairs, door closing, laughter, etc.). Comparing the two, the lab environment had less background noise, the latter being much more challenging for ManyEars to separate sound sources. Therefore, more trials were conducted in the lab environment because this is where the added benefit of using ManyEars with a dialogue management system can be best evaluated. In the cafeteria, performances were affected by ManyEars' ability to separate sound sources in such extreme conditions, a factor that goes beyond our case study which aims to evaluate the feasibility and advantages of integrating an artificial audition system to a dialogue management system.

The robot remained immobile during the trials, to avoid having to deal with the complexity that brings mobility for a posteriori analysis of dialogues. Recognition performances were derived for the system using ManyEars, and from audio streams generated by using the signal coming from one microphone. This makes it possible to evaluate the added capabilities provided by ManyEars.

A. Evaluation Criteria

A total of 2343 audio streams were recorded during these trials using ManyEars (1488) and using one microphone (855). They were then categorized based on audio stream quality (by listening to them to validate subjectively the quality of the recorded speech) and by looking at what resulted from the Dialogue Management module (i.e., Successful or Unsuccessful recognition). Stream quality is characterized by five types:

- **Good**: the audio stream contains audible speech with no imperfections.
- **Noisy**: the audio stream contains audible speech but with a pitch boost (e.g., a sudden noise) or similar inconsistencies that affect recognition. Also, the text-to-speech process is independent of audio processing, and we tried to avoid listening when the robot is talking. However, no feedback could be provided to indicate when the robot was done talking. Therefore, in a small number of cases, audio streams are corrupted with the robot's own voice.
- **Duplicated**: with ManyEars, sometimes the same sound is detected in two different locations (by the reflection

TABLE I

DIALOGUE MANAGEMENT PERFORMANCES USING MANYEARS

Stream Quality	Cnd	n	Successful	Unsuccessful
Good	LAB	351	77.8	22.2
	CAF	46	50	50
	LAB+CAF	397	74.6	25.4
	Abs	1488	19.8	6.8
Noisy	LAB	167	79.6	20.4
	CAF	42	54.8	45.2
	LAB+CAF	209	74.6	25.4
	Abs	1488	10.6	3.6
Duplicated	LAB	186	58.1	41.9
	CAF	10	0	100
	LAB+CAF	196	55.1	44.9
	Abs	1488	7.2	6
Incorrect	LAB	162	21	79
	CAF	39	12.8	87.2
	LAB+CAF	201	19.4	80.6
	Abs	1488	2.7	10.9
Useless	LAB	244	0	100
	CAF	241	0	100
	LAB+CAF	485	0	100
	Abs	1488	0	32.6
Overall	-	1488	40.3	59.7

TABLE II

DIALOGUE MANAGEMENT PERFORMANCES USING ONE MICROPHONE

Stream Quality	Cnd	n	Successful	Unsuccessful
Good	LAB	413	29.5	70.5
	CAF	8	0	100
	LAB+CAF	421	29	71
	Abs	855	14.2	35.1
Noisy	LAB	40	5	95
	CAF	3	0	100
	LAB+CAF	43	4.7	95.3
	Abs	855	0.3	4.7
Incorrect	LAB	137	11.7	88.3
	CAF	27	0	100
	LAB+CAF	164	9.8	90.2
	Abs	855	1.9	17.3
Useless	LAB	136	0	100
	CAF	91	0	100
	LAB+CAF	227	0	100
	Abs	855	0	26.6
Overall	-	855	16.4	83.6

of sound on an object), generating two distinct but quasi-identical audio streams. This may affect the performance of the Dialogue Management module (which is influenced by the lexicon, which in turn depends on what has been recognized and leading to a state change on the robot).

- **Incorrect**: the audio stream has incorrect speech caused by mispronunciation, grammatical errors, word missing, etc. CLSU's dialogue repair tool is then exploited to try to provide a valid recognition.
- **Useless**: the audio stream contains audible speech that cannot be used to influence the robot's state (because it is not part of the grammar available to the robot), and therefore must not result in successful recognition by the Dialogue Management module.

B. Results

Table I and Table II summarize the performances observed using ManyEars and using one microphone, respectively. For each combination of stream quality and trials conditions, the numbers n of associated audio streams are presented, along with the ratio of Dialogue Management results (Successful or Unsuccessful) associated with the audio stream quality type. LAB+CAF refers to the overall performance observed relative to stream quality, while Abs relates to the absolute performance observed for the audio streams processed using ManyEars ($n = 1488$) or using one microphone ($n = 855$). Note that for the one microphone case, the Duplicated type is not observed because this phenomenon occurs only with ManyEars.

With an overall result of 40.3% successful recognition using ManyEars compared to 16.4% with one microphone, the integration of ManyEars to CSLU brings significant improvement. However, the objective is for the system to process successfully streams with audible speech (Good and Noisy), and otherwise not result in successful recognition (Useless). For Good and Noisy audio streams processed by ManyEars, results are similar:

- For Good audio streams, the system successfully recognized 74.6%, which is much better than what is achieved with one microphone (29.5% for Good and 5% for Noisy audio streams, in laboratory conditions only – in the cafeteria, the streams could not lead to successful recognition).
- For Noisy audio streams, even though ManyEars introduced some unwanted distortions in 14.2% ($10.6\% + 3.6\%$) of the audio streams, CSLU was able to process 79.5% of them in laboratory conditions (and 54.8% in the cafeteria).

Useless audio streams from ManyEars are also processed correctly by not leading to false positive. Therefore, considering successful recognition for Good and Noisy audio streams, and unsuccessful recognition for Useless audio streams, the system performs adequately for 63% ($19.8\% + 10.6\% + 32.6\%$) over 73.3% ($(397+209+485)/1488$) of the audio streams generated by ManyEars, compared to 41.1% ($14.2\% + 0.3\% + 26.6\%$) over 80.8% ($(421+43+227)/855$) for audio streams derived using one microphone.

For the Duplicated audio streams, while a good proportion of successful recognition is observed in laboratory conditions, none is observed in the cafeteria. In such an open settings, vocal interactions occur faster and state change of the robot happens more often. We observed that Duplicated audio streams occur mainly in scenarios where a single interlocutor with a loud voice was interacting with Spartacus precisely when the environment became quieter. ManyEars then detected two different sound sources, which sometimes led to unpredictable behavior for the dialogue manager. This is something to improve on ManyEars by adding for instance the ability to identify the sound sources before sending it for recognition and process by the Dialogue Management

module. For the trials conducted, this would have resulted in a 6% improvement. For the Incorrect audio streams, CLSU's repair feature led to the recovery of only a small proportion of the vocal interactions, for both ManyEars and the one microphone setup. It may not be as beneficial as initially expected, because it can lead to false-positives. Therefore, considering that unsuccessful recognition of Useless audio streams is coherent with what the system has to do in these cases, we can consider the overall performance of the integration to be 72.9% $(40.3\% + 32.6\%)$ compared to 43% $(16.4\% + 26.6\%)$ with the use of one microphone, and with successful recognition of 59.8% $(40.3\%/(100\% - 32.6\%))$ compared to 22.3% $(16.4\%/(100\% - 26.6\%))$ respectively.

Comparing LAB and CAF performances, as expected recognition is better in the laboratory conditions. This can be explained because ManyEars has difficulty tracking and separating many sound sources simultaneously in such extreme noisy conditions, leading to incomplete audio streams and the introduction of inconsistencies. However, in spite of the very difficult conditions, the system was able to get successful recognition in the cafeteria (e.g., 50% of Good audio streams, 54.8% of Noisy audio streams, and even 12.8% of Incorrect audio streams), while none were recognized using one microphone. The problem comes from the inability to discriminate sound sources in noisy conditions using only one microphone. For instance, during a six minute trial in the cafeteria, only 12 audio streams (compared to 103 using ManyEars) were recorded, with an average length of 26 sec (the longest one lasting 108 sec). Recording starts when someone begin talking and is performed until no corresponding observation from the beamformer for approximately 0.5 second would occur around Spartacus. All interlocutors were thus recorded on top of each other, resulting in long incomprehensible audio streams for speech recognition, even hard to interpret for a human listener.

V. CONCLUSION AND FUTURE WORK

This paper presents the integration of ManyEars, a sound source localization, tracking and separation pre-processing system, with the CLSU dialogue management system, to study how it can affect natural language processing performances. Results from our trials suggest that such integration improves dialogue management performance in challenging conditions, in comparison with the direct use of speech input coming from a regular microphone. The results are encouraging, but there are still improvements to be made for natural language processing. In future work, ManyEars will be improved for better quality in open and complex settings. For instance, we are currently working on an approach to dynamically adapt the input gain of ManyEars to minimize distortions presented in the separated audio streams. Speaker identification [21] will also be added to facilitate tracking and identification of sound sources.

ACKNOWLEDGMENT

The authors would like to thank all the volunteers who participated in the trials.

REFERENCES

[1] X. Huang, F. Alleva, H.-W. Hon, M.-Y. Hwang, and R. Rosenfeld, "The SPHINX-II speech recognition system: An overview," *Computer Speech and Language*, vol. 7, no. 2, pp. 137–148, 1993.

[2] D. DeVault, C. Rich, and C. Sidner, "Natural language generation and discourse context: Computing distractor sets from the focus stack," in *Proc. 7th Int. Florida Artificial Intelligence Research Symp.*, 2004.

[3] S. Bayer, C. Doran, and B. George, "Exploring speech-enabled dialogue with the Galaxy Communicator infrastructure," in *Proc. Int. Conf. on Human Language Technology Research*, 2001, pp. 1–3.

[4] A. Black and P. Taylor, "Festival speech synthesis system: System documentation (1.1.1)," Human Communication Research Centre, Tech. Rep., 1997.

[5] J.-M. Valin, F. Michaud, and J. Rouat, "Robust localization and tracking of simultaneous moving sound sources using beamforming and particle filtering," *Robotics and Autonomous Systems Journal*, vol. 55, no. 3, pp. 216–228, 2007.

[6] J.-M. Valin, S. Yamamoto, J. Rouat, F. Michaud, K. Nakadai, and G. Okuno, "Robust recognition of simultaneous speech by a mobile robot," *IEEE Trans. on Robotics*, vol. 23, no. 4, pp. 742–752, 2007.

[7] S. Briere, J.-M. Valin, F. Michaud, and D. Letourneau, "Embedded auditory system for small mobile robots," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2008.

[8] A. Badali, J.-M. Valin, F. Michaud, and P. Aarabi, "Evaluating real-time audio localization algorithms for artificial audition on mobile robots," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2009.

[9] IntRoLab, "The ManyEars Project," http://sourceforge.net/apps/mediawiki/manyears/index.php?title=Main_Page, 2010.

[10] K. Nakadai, H. G. Okuno, H. Nakajima, Y. Hasegawa, and H. Tsujino, "An open source software system for robot audition HARK and its evaluation," in *Proc. IEEE-RAS Int. Conf. Humanoid Robotics*, 2008, pp. 561–566.

[11] K. Nakadai, T. Takahashi, H. Okuno, H. Nakajima, Y. Hasegawa, and H. Tsujino, "Design and implementation of robot audition system 'HARK' - Open source software for listening to three simultaneous speakers," *Advanced Robotics*, vol. 24, no. 5-6, pp. 739–761, 2010.

[12] R. Cole, D. Massaro, J. de Villiers, B. Rundle, K. Shobaki, J. Wouters, M. Cohen, J. Beskow, P. Stone, P. Connors, A. Tarachow, and D. Solcher, "Tools for research and education in speech science," in *Proceedings ESCA/SOCRATES Workshop on Method and Tool Innovations for Speech Science Education*, London, 1999, pp. 45–52.

[13] R. Cole, D. Massaro, B. Rundle, K. Shobaki, J. Wouters, M. Cohen, J. Beskow, P. Stone, P. Connors, A. Tarachow, and D. Solcher, "New tools for interactive speech and language training: Using animated conversational agents in the classrooms of profoundly deaf children," *M.A.T.I.S.S.E*, vol. 1, no. 1, pp. 45–52, 1999.

[14] M. McTear, "Software to support research and development of spoken dialogue systems," in *Proc. Eurospeech*, Budapest, Romania, 1999, pp. 339–342.

[15] L. C. Parra and C. V. Alvino, "Geometric source separation: Merging convolutive source separat ion with geometric beamforming," *IEEE Trans. on Speech & Audio Processing*, vol. 10, no. 6, pp. 352–362, 2002.

[16] J.-M. Valin, J. Rouat, and F. Michaud, "Enhanced robot audition based on microphone array source separation with post-filter," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2004, pp. 2123–2128.

[17] Y. Ephraim and D. Malah, "Speech enhancement using minimum mean-square error short-time spectral amplitude estimator," *IEEE Trans. on Acoustics, Speech & Signal Processing*, vol. 32, no. 6, pp. 1109–1121, 1984.

[18] J.-M. Valin, "Auditory System for a Mobile Robot," Ph.D. dissertation, Département de génie électrique et de génie informatique, Université de Sherbrooke, Québec, Canada, 2005.

[19] J.-M. Valin, J. Rouat, and F. Michaud, "Microphone array post-filter for separation of simultaneous non-stationary sources," in *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, 2004.

[20] F. Michaud, D. Letourneau, E. Beaudry, M. Frechette, F. Kabanza, and M. Lauria, "Iterative design of advanced mobile robots," *International Journal of Computing and Information Technology, Special Issue on Advanced Mobile Robotics*, vol. 4, pp. 1–16, 2009.

[21] F. Grondin and F. Michaud, "WISS, a speaker identification system for mobile robots," in *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 1817–22, 2012.