A Universal Speech Interface for Appliances

Thomas Kevin Harris, Roni Rosenfeld

Language Technologies Institute
Carnegie Mellon University, Pittsburgh
`tkharris, roni @cs.cmu.edu`

Abstract

Can a single, universal speech interface look-and-feel be used to effectively control a wide variety of appliances? Can such an interface be automatically derived from a functional appliance specification? We built the Speech Graffiti Personal Universal Controller (SG-PUC), a universal interface and framework for human-appliance speech interaction, as a proof-of-concept. Its specification language and communications protocol effectively separate the SG-PUC from the appliances that it controls, enabling mobile and universal speech-based appliance control. To realize such an automatically derived dialog system, the controller employs a universal control language. The development of interfaces to numerous appliances and the results of user studies demonstrate the usefulness of the SG-PUC, indicating that high quality and low cost human-appliance speech interface can be largely appliance agnostic. This investigation also helps to validate the principles of Speech Graffiti as a speech interface paradigm, and provides a baseline for future studies in this area.

1. Introduction

1.1. Background

Speech-based dialog systems have generally fallen into one of three major interface paradigms: “command and control” (C&C), “interactive voice response” (IVR), and “natural language” (NL). C&C interfaces are characterized by a fixed set of understood utterances. The users of the C&C systems must learn what utterances are appropriate under what conditions as well as the limitations of the system. IVR interfaces are characterized by a possibly dynamic set of hierarchically structured utterances. The context of the dialog state determines what utterances are understandable at any particular moment. Instead of requiring the user to memorize the utterances, the system prompts the user with the set of understood utterances at every turn. NL interfaces attempt to parse natural spoken language, extracting the task-related information from the user’s utterance. This is often done with robust parsing and with a slot-filling strategy. Little or no burden of learning or state memory is required of the user. These interfaces require either too much user training (C&C), are too inefficient for frequent use (IVR), or are error-prone and costly to develop (NL).

1.2. A proposed solution – Speech Graffiti

In an attempt to address the shortcomings of the major existing paradigms for speech interface systems, Rosenfeld et al. [2] proposed a fourth paradigm, now known as “speech graffiti” (SG). It was surmised that a universal interface style might be developed that would be both flexible and efficient, while also allowing applications to be robust and easily developed. The SG language could have special mechanisms for dealing with interface issues particular to speech, such as error correction, application help, orientation and list navigation, and that once a user learned these mechanisms, they could be applied universally to all SG applications. These ideas resulted in some working information access applications.

Our current SG design is characterized by a small set of keywords or phrase structures. These keywords need to be learned by the user, as in C&C applications, but they are independent of the task, and as such must be learned only once. The keywords provide the user with a means to explore the particular application’s variables and functions, essentially enabling quick on-line learning for any SG speech interface. The keywords also provide the ability to query and interact with the state of the system and the state of the interaction.

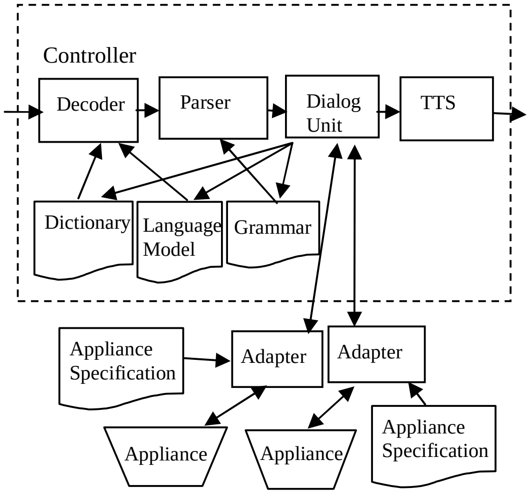
A SG user must be trained in the SG interaction style and keywords, but since these interaction primitives are all task-independent, the user need only go through this learning process once. SG systems exhibit a highly stylized interaction, and thus result in a rather low perplexity. SG systems also have a low development cycle, resulting from the highly stylized interaction language. On first use of a system, the interaction style is much like that of the IVR system. Users must query the functions, variables, and categories of their systems in order to discover the systems capabilities and limitations. Once the system is learned however, the interaction style becomes much more like C&C applications.

Tomko [3] built and tested SG systems for information access, which could be configured to interface with different database back-ends. Several systems were built, including a movie information system, an airline flight information system, and an apartment information system. The movie information system was directly compared in user studies to a NL movie information system with which used the same back-end database. Tomko found the comparison favorable to the SG system. With some success in the information access domain, in this paper we investigate whether a speech graffiti approach could be effective in the domain of physical device (appliance) control.

2. The system

2.1. Architecture

The System Architecture is rendered in the figure below. The controller directs the input and output streams of the subunits, and performs logging services. We use the Sphinx-II [7] as our decoder and Theta [8] for TTS. The parser is an implementation of the Early context-free grammar parsing algorithm. The communication protocols, appliance specifications, and much of the overall architecture are based on the Personal Universal Controller [1]. The dialog unit uses the appliance specifications to generate a grammar, language model, and dictionary. In this way, a dialog system is automatically derived from the current environment of appliances and their specifications.



2.1.1. *Functional specification of the appliance*

The appliance functional specification is described in an xml document. Each node in the appliance tree is classified as *actionable* or not. An actionable node causes an appliance backend operation when invoked, and may return a value when queried. In addition, at any one time, a particular node is said to be *in focus*, and it is generally the last invoked or queried node.

2.1.2. *The SG-PUC language*

The SG-PUC language consists of a small set of keywords (mostly the same as in the original SG design) and template-based commands, listed below. These commands operate on a tree of appliance functions. In addition, any partial path in the appliance tree is a valid command. The SG-PUC interaction grammar is automatically derived from the appliance specification. A brief description of the SG-PUC language for appliance control follows.

2.1.3. *Keywords and templates*

- **Hello James** & **Goodbye James**: starts and ends a session. ("James" is the virtual master butler who controls all the appliances).
- **Options** or **<node> options**: provides a list of functions related to the node or to the current focus.
- **More**: reads a few more items from a list.
- **Status** or **<node> status**: summarizes the status of an appliance of a node.
- **Set** … **to** …: sets a node to a value, e.g. "set station to WDUQ".
- **Repeat**: repeats the last utterance.
- **Help**: lists and tersely explains the keywords.

The SG-PUC language, with only 8 keywords, is designed to be generic enough to handle most appliances, while being easy to learn and affording an efficient interaction. Experiments were conducted to test these design goals.

3. Experiments

Twenty-one subjects were recruited to test the system.

3.1. Procedure

The subjects were offered $12 for participating in the experiment and an additional $1 for each of the 12 tasks that they were able to successfully perform. Each experiment lasted between 1 and 2 hours.

Since Sphinx had been trained on American English, the subjects were screened to be native speakers of American English. Since Tomko [3] demonstrated that people with computer sciences training perform markedly better than the general population in similar tasks, they were screened out of this study. The subjects were all naïve to computer speech interfaces and specifically to Speech Graffiti.

3.1.1. *Training*

In order to use the system, subjects completed an on-line tutorial [4], supervised by the experimenter. The tutorial consisted of four web pages of instruction, examples, and exercises, with a working alarm clock appliance and examples from a television appliance. The tutorial covered all of the keywords, session management, appliance exploration, setting and querying appliance states, and getting help. The experimenter tested the subjects to demonstrate an understanding of these lessons.

3.1.2. *Habitability. cross-application skill transfer, and unification*

Once the subjects mastered the interaction primitives of the system, the training was concluded and the subjects were presented with four tasks related to the use of the alarm clock. This very same appliance was used during their training, so it was expected that they would perform well.

After completing the alarm-clock tasks, the subjects were asked to complete four tasks on an adapted shelf stereo. The subjects had not been trained nor had they had any interaction with the stereo up to that point. These tasks were designed to test the ability of the subjects to navigate the functions of a new appliance using the interaction language.

After completing the transfer tasks on the shelf stereo, the subjects were asked to complete four tasks in the environment where both the stereo and the alarm clock were both active. These tasks were designed to measure the potential for confusion in a multi-appliance environment.

3.1.3. *Questionnaire*

Once the subjects had attempted all 12 tasks, they were asked to complete a questionnaire, composed of a randomized list of the SASSI [5] recommended statements on a 7-point Likert scale. It measures subjective evaluations of speech applications in 6 different categories: Speed, Habitability, Annoyance, Cognitive Demand, and Likeability.

3.2. Experimental variables

The subjects were divided into two groups: Group 1 (n = 11) and Group 2 (n = 10). The experimental procedure did not vary between the two groups; however, five modifications were made to the system for Group 2, which were all designed to improve the recognition rates for Group 2.

1) Group 1 employed automatic utterance segmentation. Group 2 employed a push-to-talk system.
2) Language models for the decoder for Group 1 were created by generating a corpus of 60,000 sentences from a probabilistic CFG (derived from the appliance specification), and then using word counts from the generated corpus to derive tri-grams. For Group 2, exact tri-grams were computed from the probabilistic CFG via Stolcke's method [6].
3) The probabilistic CFG's were maximum-entropy trained on transcripts from Group 1, for use with Group 2.
4) For Group 1, the decoder only computed one codeword per frame. For Group 2, 4 codewords were computed per frame.
5) For Group 1, every set of words that always occurred together in the grammar was combined into a single conjoined word. For Group 2, this practice was abandoned.

4. Analysis

4.1. Task completion and efficiency

Of the 12 tasks, the median number of completed tasks among the subjects was 12 and the first quartile number of completed tasks was 11. Thus, the subjects were able to effectively control the appliances using the interaction language that they were taught.

Efficiency is an important question for expert users. Since all of our experiments involved subjects who were first-time users of these systems, we don't know yet what the interaction efficiency will be like for expert users. The elapsed times and number of utterances that our naive subjects required in order to operate the appliances involved becoming familiar with the interaction language rules that they had just learned, exploring the functions of the appliances, and then executing the functions that they had uncovered.

4.2. Training

Subjects spent an average of 34 minutes learning the interaction language. The instruction was semi-supervised, but was largely the product of reading four web pages and trying some exercises.

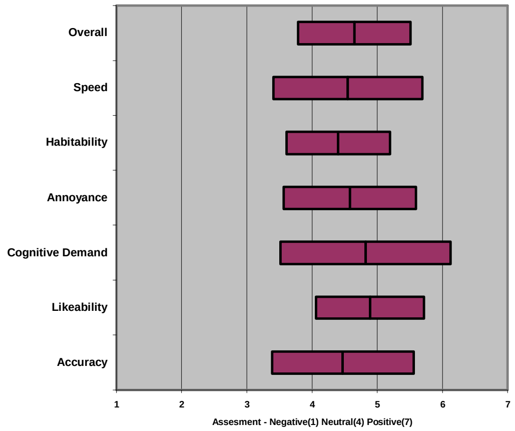
Systems designed for the general public usually require no user training at all, being either natural language systems or interactive voice response systems. The Speech Graffiti philosophy, however, is that some amount of user training is cost effective if that training is pertinent to many Speech Graffiti applications. Half an hour of training might be well justified when amortized across a large number of appliances and a long period of time. In order to test the necessary cross-application SG skill transference, after completing training with the alarm clock as an example, the subjects were introduced to the stereo, an appliance with which they had not had any previous interaction or training.

Of the 6 stereo-related tasks, the median task completion among the 21 subjects was 6 tasks, and the first quartile completed 5 of the 6 tasks. This would seem to indicate that the interaction language that the subjects learned was generic and robust enough for them to transfer their knowledge and operate yet unseen Speech Graffiti appliances.

The instruction that the subjects participated in was developed for the reported experiments. It is expected that these instructions could be iteratively refined to be shorter and more efficient. The instructions could be also be merged with voice registration, which would further reduce the marginal cost of instruction.

4.3. Subjective analysis

After attempting to complete the 12 tasks using the interaction language, the subjects completed a 34-item questionnaire about their assessment of and experience with the system. The questionnaire was a randomized list of the SASSI questions on a 7-point Likert scale.



Group 2 reported more positive subjective evaluations for each of the 6 categories, resulting in a 0.76 point higher overall evaluation. The only differences between the two groups were the Experimental Variables (§3.2), which lowered the WER in Group 2. Not surprisingly, errors in recognition play an important part in subjective evaluations.

The results of the subjective evaluations from Group 2 are shown in the figure above. The center line of each bar shows the average subject's score in each category, with bars to the left and right representing the standard deviation. Only one of the ten subjects had an overall negative (less than 4) evaluation, and the average score in all of the 6 categories is positive. Even so, the subjective evaluations for this system are borderline neutral

One way to improve the user satisfaction would be to further reduce the speech recognition error rates. The entire instructions, system, and task between Groups 1 and 2 were identical except for the recognition system. Even so we find much more significant differences between the performances of these tasks than in any of our demographic analyses of the subjects. Group 2 reported less annoyance ($p < 0.05$), less cognitive demand, and a more habitable ($p < 0.05$) system.

The most interesting result is a significant improvement ($p < 0.05$) in the percentage of grammatical utterances between the two groups. Since our language model is based on our grammar, it is obvious that better grammar would lead to better recognition rates. However, by experimenting with improving the recognition alone, we have found that better recognition causes more grammatical utterances. Our theory is that better recognition leads to more appropriate responses, which entrains (especially the naïve user) within the grammar. In other words, if a grammatical utterance is understood properly, it entrains the user to speak grammatically again. When it is misunderstood, it influences the user to try another approach. This leads to a feedback influence where recognition accuracy influences grammar, which influences recognition accuracy. One can expect that off-line improvements or degradations in recognition accuracy will be compounded on-line.

We examined several demographic factors, including age, gender, second language proficiency, education level, frequency with which they operated computers, alarm clocks, stereos, and other speech-enabled appliances. No strong correlations were found, except that word error rates were higher ($p < 0.05$) for women.

4.4. Grammar

For Group 2, the median ungrammatical utterance rate was 7%, and the median grammatical utterance error rate was 11%. The unproductive utterance rate is the rate of ungrammatical utterances plus the number of incorrectly recognized grammatical utterances, divided by the total number of utterances. The median unproductive utterance rate was 22%. Thus we can expect about 22% of utterances to be unproductive, with a little more than half of those problems due to poor recognition and a little less than half of those problems due to poor grammar. The low utterance recognition error rate of 11% (for a general purpose recognizer) is a direct result of the constrained vocabulary and grammar that this system employs. Nonetheless, it is worth examining the ungrammatical utterances to determine if the grammar could be relaxed in minor ways such that the overall unproductive utterance rate can be diminished.

Of 3143 utterances with the 21 subjects, 592 (~19%) were ungrammatical. We categorized these 592 utterances into 36 sometimes overlapping categories of syntax errors. The top 10 categories, which represent 74.2% of the ungrammatical utterances, are shown below, along with the percentage of those utterances among the entire population of utterances.

Percentage	Syntactic Error
Broken Utterance	4.10%
Generalized Node	2.29%
Unknown Value	2.13%
Out-of-vocabulary	1.59%
Imaginary Function	1.50%
Partial Token	0.86%
Superfluous Set	0.83%
Missing “to”	0.73%
Setting Read-only	0.73%
Combining Commands	0.67%

A “broken utterance”, for example, is one where the subject starts but doesn’t finish a complete command or query. Sometimes the command or query is finished on the next utterance. A typical example is the utterance “set alarm time to…”; the subject trails off in thought and the recognizer, thinking that the utterance is complete, tries to parse the result and fails, usually interrupting the rest of the utterance. Several methods could be employed to deal with this issue. The simplest method is to employ a push-to-talk mechanism, which was done for Group 2.

5. Conclusions

By combining elements of the Universal Speech Interface and the Personal Universal Controller, and refining these methods, we have created a framework for appliance control speech interfaces that is both personal and universal. This achievement, which allows product engineers to integrate speech interfaces into their products with unprecedented ease, comes at a price, however. The interaction language is an artificial subset language that requires user training.

It is clear that, in learning this language, some training is required. Ultimately, this training may be provided online, and with no human intervention. As has already been demonstrated, knowledge of the interaction language transfers from one appliance to another very well.

The use of a universal control language also provides the benefit of clear, unambiguous semantics and low input perplexity. These factors translate into a more robust system, with fewer errors than functionally equivalent natural language speech interfaces.

6. Acknowledgements

Many thanks to Jeff Nichols and Brad Myers for their contributions of the PUC architecture. A PUC-adapted shelf stereo is courtesy Maya, Inc.

7. References

[1] Nichols, J., Myers, B., Harris, T. K., Rosenfeld, R., Shriver, S., Higgins, M., Hughes, J., "Requirements for Automatically Generating Multi-Modal Interfaces for Complex Appliances", *IEEE Fourth International Conference on Multimodal Interfaces*, pp. 377-382, 2002.

[2] Rosenfeld, R., Olsen Jr., D.R., Rudnicky, A., “A Universal Human-Machine Speech Interface”, *Technical Report CMU-CS-00-114, School of Computer Science, Carnegie Mellon University*, 2000.

[3] Tomko, S., “Speech Graffiti: Assessing the User Experience”, *Masters Thesis, School of Computer Science, Carnegie Mellon University*, 2004.

[4] http://www.cs.cmu.edu/~tkharris/usi/tutorial

[5] Hone, K., “Towards a Tool for the Subjective Assessment of Speech System Interfaces (SASSI)”, 2000.

[6] Stolcke, A. and Segal, J., “Precise *n*-gram probabilities from stochastic context-free grammars”, *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pp. 74-79, 1995.

[7] Huang, D., Alleva, F., Hon, H.W., Hwang, M.Y., Lee, K.F., and Rosenfeld, R., “The Sphinx-II Speech Recognition System: An Overview”, *Computer, Speech, and Language*, 7(2):137-148, 1993.

[8] Cepstral, LLC., “Theta: small footprint text-to-speech synthesizer”, 2004.