A Situated, Embodied Spoken Language System for Household Robotics

Adam Vogel, Karthik Raghunathan, Stefan Krawczyk

Department of Computer Science, Stanford University, Stanford, California, USA
{av, rkarthik, stefank}@cs.stanford.edu

Abstract

We present an embodied, situated natural language user in the form of a spoken dialog system on the STanford AI Robot (STAIR). Combining a limited vocabulary HMM-based speech recognizer with sliding window object detection from monocular vision and and robotic manipulation with a five degree-of-freedom arm, we demonstrate the feasability of a mobile robot able to find (perceive) and manipulate common household objects. Our main focus is on the connection from speech and language to non-linguistic knowledge (perception of objects currently in the visual field) and capabilities (e.g. grasping of objects or speech synthesis). This task provides a natural *extrinsic* evaluation of the various subcomponents. We can now more accurately address questions such as how accurate our object recognition system needs to be to realize a concrete activity.

1. Introduction

We present a situated, embodied natural language user in the form of a spoken dialog system on the STanford AI Robot (STAIR). In contrast to telephone dialog systems or the analysis of text corpora, we explore the connection between language - in the form of speech recognition and synthesis, perception - in the form of monocular object detection and depth estimation, and non-linguistic physical activity - in the form of a 5 degree-of-freedom (DOF) grasping arm. Receiving audio input through a wireless microphone worn in the ear of the user, STAIR uses the CMU Sphinx3 automatic speech recognizer to extract the maximum a posteriori word string from a given utterance. We restrict output over a limited vocabulary related to the fetching or bringing of common household objects. STAIR connects the usage of these words to the world in two ways: concrete nouns have a corresponding trained object classifier (see Section 3), and realizations of verbs with the correct syntactic arguments induce routines of activity on STAIR.

Situated, embodied language use falls under the banner of *pragmatics*, the study of the *usage* of language[1]. In contrast to other NLP tasks such as information extraction (IE) [7], which focus on the descriptive or propositional content of utterances, situated language use deals in *illocutionary* acts (e.g. speech acts [18]) and *performative* utterances [6], such as asking someone to get something for you, or promising to do something in the future. As IE systems depend on epistemological theories such as standard probabilistic knowledge representation [14, 12], we rely on theories of *activity*, or more plainly stated: the constant redecision of what to do now [5, 13, 8]. In this project, requests and commitments to fetch objects are illocutionary acts we employ, and the propositional or descriptive content of the utterances are the concrete objects mentioned (persistent patterns of perceptual activation in the case of vision, and also those affordances ready-to-hand with the object, such as the "easy" grasp point of the handle of a coffee mug [11, 9]).

Our system is a set of software modules in a distributed, real-time computing environment called the Robot Operating System (ROS). The rest of this technical report outlines the design of the specific modules, both linguistic and non-linguistic, and on the quantitative evaluation of each subcomponent. These modules communicate in a variety of formats, such as text strings from the ASR component and to the speech synthesis module (see Section 2)

We report positive evaluation results of our speech recognition and "understanding" system in Section 6. Unfortunately we do not include an empirical system-wide evaluation, as we have not yet completely realized the object fetching activity. Instead, after STAIR recognizes a verbal fetch request, STAIR looks for an object of the corresponding type (i.e. a stapler) in its visual field. If one is found, it moves its arm to one of three positions, depending on whether the object appears in the left, center, or right of its visual field. The remaining work to fully implement this activity is non-linguistic, in the form of more, improved object classifiers and fully realized grasping and manipulation.

2. Speech Processing

2.1. Automatic Speech Recognition

We use CMU's state-of-the-art large vocabulary speech recognition system, Sphinx-3 [1] for automatic speech recognition of the user's speech commands. Sphinx uses Hidden Markov Models (HMM) with continuous output probability density functions and is based on the conventional Viterbi search algorithm and beam search heuristics. Sphinx-3's "Live Decode" mode also has a speaker adaptation component, which improves ASR performance after about ten seconds of speech.

To decode speech with Sphinx-3, we first need to provide it with an acoustic model, a language model and a phonetic lexicon. We use the HUB4 (broadcast news) acoustic model trained on wideband (16kHz) speech, which is available for free download from the sphinx website. The models have been trained with Mel-frequency cepstra (MFC) vectors obtained from 140 hours of 1996 and 1997 hub4 training data. Each vector is a 39-dimensional vector composed of 13 cepstral coefficients, 13 delta cepstra and 13 double delta cepstra. The models are 3-state within-word and cross-word triphone HMMs with no skips permitted between states. They comprise of 6000 senones and the number of codewords used in the sub-vector quantization is 1024 for 1 Gaussian/state models, 2048 for 2 Gaussian/state models and 4096 for 4,8 Gaussian/state models.

We used the Sphinx Knowledge Base Tool [4] to train a trigram language model on a sample corpus of about 100 sen-

[1] As opposed to the study of speaker or situation-independent meaning, usually referred to as *semantics*. There is an active debate over whether there is really any distinction to be made here [21].

tences from our domain (object retrieval). This tool also generated the phonetic lexicon containing pronunciations for each of the 34 words in our vocabulary. Our vocabulary consists of synonyms for fetching (grab, fetch, get, bring), a few function words (a, the, me, your), and common household objects (stapler, cup, mug, keys, apple, peach, banana, bottle, remote). In addition, we added vocabulary for basic movement commands, like "move your arm left".

2.2. Natural Language Understanding

To make sense of the decoded utterance provided by the ASR component, we need to perform some basic Natural Language Understanding (NLU). We focus on verb-object syntactic forms, like "Grab the stapler", since in our current activity it is always a person asking STAIR to do something (so subjects are always implied). Thus, ASR output strings that contain only one verb and one noun (not pronoun) are mapped to a command to "verb the noun". On the other hand, strings we cannot make sense of are more complicated noun phrase structures, which in our case are a good indicator that we had problems with ASR (e.g. "fetch the remote banana a") and thus should seek clarification before acting. Some of the "rules" that the NLU has are:

- If the utterance contains "arm" or "hand", "move" and one of "up", "down", "left", or "right" then move the arm in the desired direction.
- If the utterance contains one of "mug", "banana", "stapler", "apple", "peach", "cup", "bottle", "glass", "remote", or "phone" and one of "fetch", "grab", "get", or "bring" then fetch the desired object.
- If the utterance does not meet the above conditions, do nothing.

2.3. Speech Synthesis

We use the Festival Speech Synthesis System [2] for text to speech synthesis using an American English male voice. It uses the UniSyn residual excited LPC diphone synthesizer and the letter to sound rules trained from the CMU lexicon for pronunciation. The intonation and duration information are trained from the Boston University FM Radio corpus. The node controlling Festival receives the text to be synthesized from the NLU and then "speaks" this response out to the user.

3. Vision

We require as input to our system a trained object classifier for any class of objects which we have a corresponding noun. We used the STAIR Vision Library [19] which contains code for training and evaluating sliding window detectors, among other common computer vision tasks like image segmentation.

The focus of this quarter project is on speech and not vision, so our methods used here are basic and leave much room for improvement. Ideally we would have a pre-trained object detector for every noun in our lexicon, but we only manage to get object classifiers for "mug" and "stapler". These detectors take a window in the image and return a score (normalized between zero and one) of how likely the given object is to occur in that window.

These detectors are trained on hand labeled training instances, using a form of boosted decision stumps over a dictionary of "patches", or common local patterns of image features which are robust to occlusions and shape variations [20]. See [16] for more details on the training procedure.

To detect objects in a new image, we "slide" this window over the image, varying the window size. Next we threshold the detection scores at a given level θ, treating all detection windows with score greater than θ as positive and ignoring the rest.

Critically, we do not perform any post-processing on the output of the sliding window object detector. This means that if for a given window we get positive results from both detectors, we will believe there is both a mug and a stapler there. This could actually be the correct answer in the case of occlusions, or if we are viewing the objects from a perspective that makes them collinear when projected onto the camera lens. Secondly, we do not use any *context* features, such as the output of an image segmentation algorithm. Recent research in computer vision [10] suggests that false positives can be greatly reduced through the use of nearby image segmentation labels. This interaction captures the fact that staplers are oftentimes on the tops of tables, but not on walls for instance. Lastly, STAIR also has access to accurate depth maps returned by the "Borg" laser scanner, as introduced in [16], which shows that depth information can also aid in object recognition.

4. Manipulation

STAIR is fitted with a Neuronics "Katana" 5 degree-of-freedom (DOF) arm. Fitted with two fingers for grasping, the arm is driven by harmonic drives which are located at each joint. It can lift roughly 3kg, which is sufficient to lift most common household objects.

The arm software includes source code for common tasks such a calibrating the arm, opening and closing the gripper, and moving to a given configuration, which is specified by the desired angle of all 5 degrees of freedom. The STAIR group has developed code to predict grasp points on novel images [17] and also for motion planning and control to grab an object at a given point in arm-space.

As the focus of this work was on speech recognition and synthesis, we did not fully implement grasping. Instead, given a positive object detection that we want to grab, we move the arm in the general direction of the object (left, straight ahead, or right), varying by the midpoint of the bounding box of the object.

5. System Architecture

The Robot Operating System (ROS) [15] is an open source distributed platform for controlling robot components. Systems using ROS are composed of nodes, are processes which communicate messages over "topics", which other nodes can subscribe and publish to. There is a master controller node which contains the registry of what is on the network, with a network consisting of a series of ROS nodes and "topics". When nodes are started they register with the master controller, announcing what messages they subscribe to and what messages they publish. The core node then facilitates telling publishers where the subscribers are. We have eight ROS nodes, broken up into six packages: Audio, Speech Recognition, Natural Language Understanding, Speech Synthesis, Vision, Robotics. Each package consists of one or more ROS nodes, as depicted in Figure 1.

5.1. Audio

ROS audio signal processing utilizes the portaudio library to interface with the audio card and write audio files. The audio is sampled at 16kHz clipped when the energy level increases

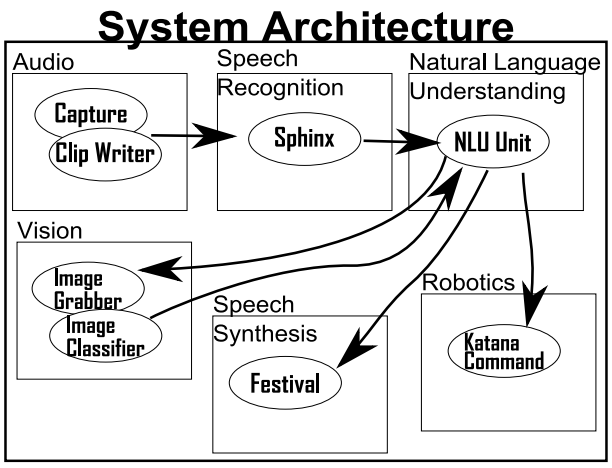

Figure 1: Message flow between ROS nodes

above a set threshold, until energy drops below the threshold. The audio module publishes to the **audio_loc** ROS topic at the completion of every clip, allowing further processing nodes to run.

5.2. Speech Recognition

We wrote a ROS node for calling Sphinx3. We modified Sphinx3's *livepretend* program to accept input directly from our ROS wrapper (which subscribes to the **audio_loc** topic to get audio clips). This enabled us to decode a "cached" speaker clip first, so that we don't have to rerun speaker adaptation for every utterance. The string of decoded speech was published to the **nlu** topic.

5.3. Natural Language Understanding

This node performed the "Natural Language Understanding" computation, which was invoked for every message on the **nlu** topic. The outputs from the NLU script which resulted in an action were: vision, for publishing commands to the vision channel; arm, for publishing general robot arm movements; hand, for publishing opening or closing of hand movements; Borg, for firing the laser; and null for doing nothing. The vision, arm and hand commands, were followed by arguments specifying what kind of action to publish.

The NLU node also subscribed to the result of image classification on the **object_detection** topic. Once it published a vision command, it stored the linguistic state, so that when a message arrived on the **object_detection** channel, it could correlate it and publish an appropriate preprogrammed response to the **tts** channel and if appropriate the robot arm.

5.4. Vision

The vision package was composed of an image grabber node and an image classification node. The first node subscribed to the **image_grab** channel and when it got input it took a photo with the camera. It then published this information to the **image_loc** topic where the image classifier was listening and the image classifier ran the image classifier script. The output was then published to the **object_detection** topic where the NLU node interpreted it.

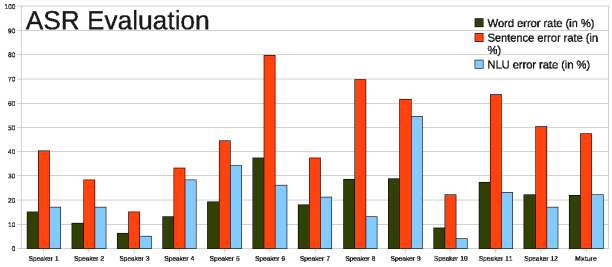

Figure 2: Error rates for various speakers

5.5. Speech Synthesis

This node simply subscribed to the **tts** topic and invoked festival to produce utterances received over **tts**.

5.6. Robotics

This node controlled the position of the arm. It subscribed to the **katana_command** topic and executed incoming commands, which are currently propositional. For example, if **LEFT** was received, then it proceeded to move the robot base joint left thirty degrees.

6. Evaluation

We evaluated the performance of our speech system by conducting a user study with twelve test subjects. Each user read a set of one hundred commands to the system and the number of times the system performs as per the user's intentions is calculated (NLU accuracy). We also use the NIST speech recognition scoring toolkit (SCLITE) [3] to do a more detailed analysis of the speech recognition errors. SCLITE computes the Word Error Rate (WER), the Sentence Error Rate (SER), and classifies errors into those arising from substitution, insertion and deletion.

While WER certainly is a good indicator of the ASR accuracy, the user's utterance being recognized verbatim is not necessary for our system to function correctly. Our performance is more strongly dependent on correctly recognizing content words (nouns, verbs), while nuanced usage of determiners (like "a", "an" and "the") is not as important. Thus, we calculate a separate NLU error rate, i.e. the number of times the STAIR misunderstood the meaning of the user's utterance and did something in contrast to what was expected. The results from our experiments are summarized in Table 1.

Metric	Percentage Error
NLU Error Rate	21.8
Word Error Rate	19.7
Sentence Error Rate	45.5

Table 1: Evaluation results on a test set of 1,188 sentences containing a total of 4,104 words, with 12 different speakers speaking 99 sentences each.

The above results indicate that on an average, our system performs correctly nearly 80% of times it is used. Further error analysis reveals that:

- Our system did a worse job on speakers who tried to be overcautious in enunciating every word properly (and

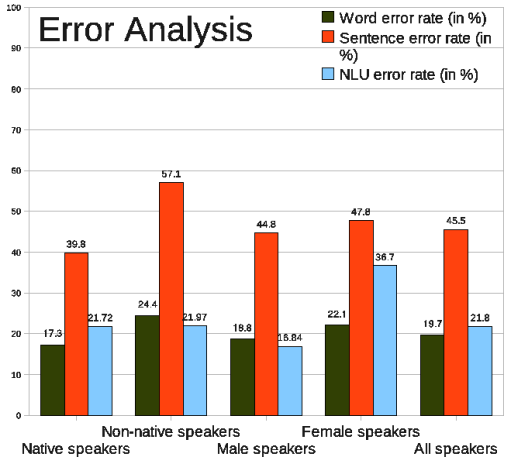

Figure 3: Error rates for different speaker categories

hence ended up sounding artificial) than speakers who spoke in a natural free flowing manner.

- Our system performed better with male speakers than female speakers on average.
- Our system had a much better recognition accuracy with native English speakers than non-natives. However, the NLU error rates for both were almost the same indicating that most of the mistakes that the ASR made with non-native speakers was in the "a's" and "the's". Thus the system is usable by speakers of both kinds.

7. Future Work

Several extensions and improvements which we plan to implement next quarter are:

- improve and expand object detectors to more household things
- train an acoustic model for noisy environments
- implement a real dialog system with task confirmations and clarifications
- implement grasping with the Katana arm, as well as grasp point prediction in the vision module
- integrate the STAIR navigation components to be able to move the entire robot
- build a more mature NLU component, using more sophisticated theories of syntax (CS224N)
- add a more sophisticated theory of activity, integrating reactivity to changes in the environment with intentional, temporally extended policies.

8. Acknowledgments

We thank Quoc Le, Olga Russakovsky, Adam Coates, Morgan Quigley, Angrew Ng, and Dan Jurafsky for help with ROS and STAIR.

9. References

[1] Sphinx-3 guide. http://cmusphinx.sourceforge.net/sphinx3/.

[2] The festival speech synthesis system. http://www.cstr.ed.ac.uk/projects/festival.

[3] Nist evaluation tools. http://www.nist.gov/speech/tools/.

[4] Sphinx knowledge base tool. http://www.speech.cs.cmu.edu/tools/lmtool.html.

[5] Philip E. Agre and David Chapman. Pengi: an implementation of a theory of activity. In *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87)*, pages 268–272, 1987.

[6] John Austin. *How to do Things with Words*. Harvard University Press, Cambridge, USA, 1955.

[7] Michele Banko, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. Open information extraction from the web. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2007.

[8] David Chapman. *Vision, instruction, and action*. MIT Press, Cambridge, MA, USA, 1991.

[9] Peter Gorniak and Deb Roy. Situated language understanding as filtering perceived affordances. *Cognitive Science: A Multidisciplinary Journal*, 31(2):197–231, 2007.

[10] Geremy Heitz and Daphne Koller. Learning spatial context: Using stuff to find things. In *ECCV '08: Proceedings of the 10th European Conference on Computer Vision*, pages 30–43, Berlin, Heidelberg, 2008. Springer-Verlag.

[11] Nikolaos Mavridis and Deb Roy. Grounded situation models for robots: Where words and percepts meet. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2006.

[12] Brian Milch, Bhaskara Marthi, Stuart Russell, David Sontag, Daniel L. Ong, and Andrey Kolobov. *BLOG: Probabilistic Models with Unknown Objects*. 2007.

[13] Nils J. Nilsson. Teleo-reactive programs for agent control. *Journal of Artificial Intelligence Research*, 1:139–158, 1994.

[14] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems : Networks of Plausible Inference*. Morgan Kaufmann, September 1988.

[15] Morgan Quigley. Robot operating system. http://pr.willowgarage.com/wiki/ROS.

[16] Morgan Quigley, Siddharth Batra, Stephen Gould, Ellen Klingbeil, Quoc Le, Ashley Wellman, and Andrew Y. Ng. High-accuracy 3d sensing for mobile manipulation: Improving object detection and door opening. In *International Conference on Robotics and Automation (ICRA)*, 2009.

[17] Ashutosh Saxena, Justin Driemeyer, and Andrew Y. Ng. Robotic grasping of novel objects using vision. In *In International Journal of Robotics Research (IJRR)*, 2008.

[18] J. R. Searle. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press, 1969.

[19] Andrew Y. Ng Stephen Gould and Daphne Koller. The stair vision library. In *http://ai.stanford.edu/ sgould/svl*, 2008.

[20] Antonio Torralba, Kevin P. Murphy, and William T. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(5):854–869, 2007.

[21] Ludwig Wittgenstein. *Philosophical Investigations*. 1953.