Development of Language models for Voice Controlled Micro Air Vehicle (MAV)

Lakshmi P*, Arpana R**, Veena S***, Sandhya Lakshmi R**** and Sunil Kumar S Manvi*****
*Senior Scientist, ***Principal Scientist, CSIR- National Aerospace Laboratories, Bengaluru
*lakshmisri@nal.res.in, ***veenas@nal.res.in
**PG Student, *****Director, School of C & IT, REVA University,
**arpanarapheal@gmail.com
****PG Student, RV college of Engineering

Abstract: This paper focuses on the development of the language models for an effective Automatic Speech Recognition (ASR) System integrated into the Ground Control Station (GCS) which controls the Micro Air Vehicle (MAV). The MAV commands are derived based on the functionalities of the GCS and the corresponding language model is integrated with the acoustic model to increase the accuracy of the ASR.

Keywords: ASR, MAV, GCS, Hidden Markov Model (HMM), HTK, Acoustic Model, Language models, Mission Planner.

Introduction

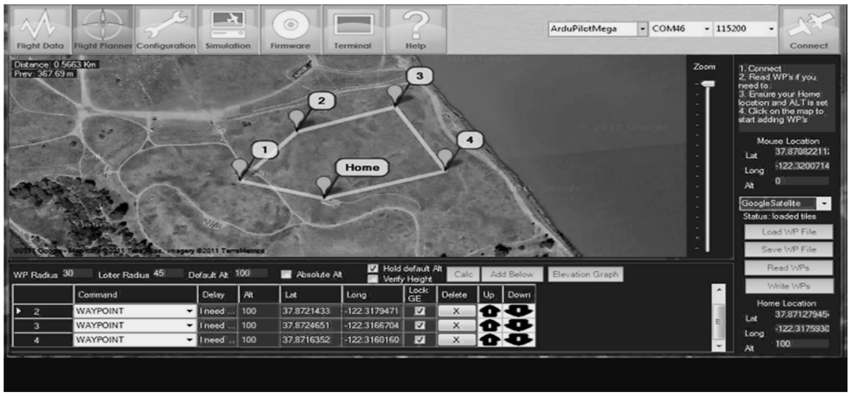
MAVs are used for both civilian and military applications. They are controlled using a GCS. Reference [1] highlights integration of isolated word recognition module into the GCS. Reference [2] is an extension to [1] and proposes incorporation of voice commands spoken as sentences into GCS. The voice based commanding is an application which requires accurate speech recognition. Reference [2] may not be able to achieve this accuracy as it does not use a language model [3] to improve the result obtained by acoustic model. A language model is required for interpretation of the uttered speech.

Statistical techniques are commonly used to develop language models because of the complexities of natural language grammar. They are really helpful when the training and the test sets are identical [4][5]. The language model has to be fast and compact to be used in the real time systems like MAV [6].

The organization of the paper is as follows: Section 2 briefs the overview of the system, followed by implementation in Section 3 and Section 4 talks about the simulation results.

System Overview

The GCS is GUI based software which controls MAV. A speech interface is provided to the GCS software to control the MAV. Fig. 1 shows the snap shot of the Flight Planner tab of the Mission Planner window. There are five waypoints at specific latitudes and longitudes. The selection of the waypoints are done by mouse clicks or keying in the values. The mouse clicks are replaced by the voice commands like "SET WAYPOINT AT FIFTY FOUR POINT TWO, TWELVE POINT SEVEN" or "DISARM THE MOTORS" with ASR technology in place.



ArduPilotMega COM46 115200

1. Connect
2. Read WP's if you need to.
3. Ensure your Home location and ALT is set
4. Click on the map to start adding WP's

Mouse Location
Lat 37.87082211
Long -122.3200714
Alt 0

GoogleSatellite
Status: loaded tiles

WP Radius 30 Loiter Radius 45 Default Alt 100 □ Absolute Alt ☑ Hold default Alt □ Verify Height

	Command	Delay	Alt	Lat	Long	Lock GE	Delete	Up	Down
▸ 2	WAYPOINT	I need ...	100	37.8721433	-122.3179471	☑	X	⬆	⬇
3	WAYPOINT	I need ...	100	37.8724651	-122.3166704	☑	X	⬆	⬇
4	WAYPOINT	I need ...	100	37.8716352	-122.3160160	☑	X	⬆	⬇

Home Location
Lat 37.87127945
Long -122.3175930
Alt 100

Figure 1. Snap shot of the Mission Planner software

Development of Language Model

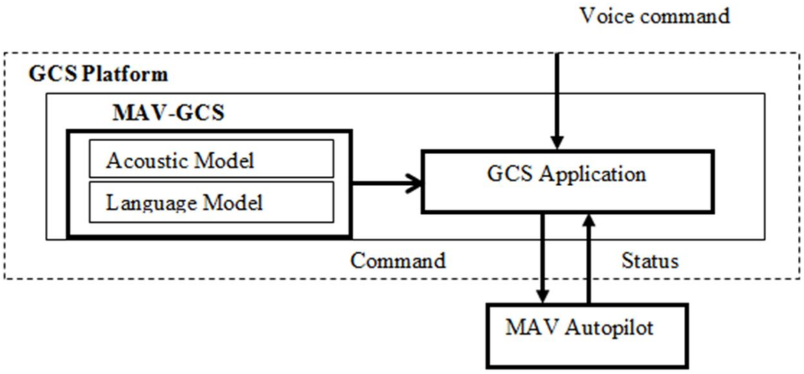


Figure 2. System Overview of Voice Communication Interface with GCS

It is proposed to incorporate Language Model to improve the recognition accuracy for the Mission Planner commands [2], which gives a description of development of acoustic model and its integration into the Mission Planner. This work uses the same framework for the incorporation of the Language Model to achieve better accuracy.

Fig. 2 shows the voice communication interface between the MAV and the GCS. The MAV and the GCS communicate using the wireless X-Bee. The GCS software considered is the Mission Planner [7]. A speech interface is built into the system not only for navigating the menu items but also for setting the MAV parameter values and controlling the MAV maneuvering.

A typical ASR comprises of acoustic and language models. A language model complements the results of the acoustic model to achieve better recognition. An acoustic model is one which models the relationship between words (or parts of words called phonemes) and the acoustic signal, whereas the language model models a language stream with the probabilities of words and sequences of contiguous words. The fundamental units used considered for acoustic and language modeling are phonemes and words, respectively. It is a statistical n-gram word-based model. The probability of each word is predicted by its (n-1) predecessors. This probability is estimated on text training data using maximum likelihood. Back-off to lower order n-grams, or interpolation with lower order models provides probability estimates for events unseen in the training data[8].The back-off probabilities helps in recognizing the unseen word sequence/word in the command. The generated language models must follow ARPA-MIT format.

For developing ASR applications, HTK toolkit [9] and CMU Sphinx[10] are the widely used open source libraries. This paper is based on acoustic model developed in [2] and the language model is developed using the CMU Sphinx lmtool[10].

Reference [2] employs acoustic model using the HTK[9]. The recognition is done using the HVite [9] recognition engine of HTK. In order to improve the recognition accuracy, the proposed work develops a language model which is integrated with the acoustic model.

Corpus

Reference [11] addresses formation of speech commands for the operation of UAV. This system has a limited vocabulary and a fixed set of commands based on the ATC (Air Traffic Control) phraseology. Building ASR for controlling MAV also requires a very limited vocabulary of words and the model to be developed is simple and compact. This paper aims at creating a list of feasible voice commands framed for Mission Planner [12] based on [11].

The set of commands formulated are maintained in a text file which is called the corpus/training set for language. This file is manually created since a readymade corpus is not available for commanding MAV through speech. The voice utterance of the same commands is used for acoustic modeling. Thus the corpus comprises a total number of 130 training commands, which accounts to 698 words out of which 82 are distinct. Each command in the training set is encapsulated within <s> and </s> which indicates the start symbol and the end symbol. In this case, the generated corpus is compact so the resultant language model is also compact. The commands formulated are simple such that the end user is at ease while uttering these commands. Some of the commands which are used for building the language model are as follows:

TURN LEFT SIX DEGREES
CHANGE SPEED TO EIGHT METERS PER SECOND
LOITER TWELVE SECONDS
DISARM THE MOTORS
LOITER AT WAY POINT TWO
LOITER UNLIMITED AT FIFTYONE HUNDRED AND TWELVE AND SIXTY
MOVE DOWN BY THREE FEET
MOVE UP ELEVEN FEET
NAVIGATE TO WAY POINT TWENTY TWO FORTY FIVE SIXTY

PITCH DOWN ONE DEGREE
LOITER FIFTY TIMES IN WAY POINT SIX
PITCH UP FOUR DEGREES
REPEAT RELAY NINE TWENTY TIMES ONE SECONDS
SET ALTITUDE TO HUNDRED FEET
SET GROUND ALTITUDE TWELVE FEET
SET HOME TO TWO FEET
SET RELAY NINE ON
ZOOM IN THREE TIMES
SET RELAY ONE OFF
START THE MISSION
TAKEOFF FROM EIGHTEEN FEET
TAKEOFF FROM HOME
TURN RIGHT THIRTY DEGREES
ZOOM OUT NINE TIMES

A test file is created with a set of 30 distinct MAV commands, which are different from training set but follows the same grammar. These test commands are also encapsulated by <s> and </s>. This accounts to 165 words which are different from the training set. For e.g.
TURN LEFT EIGHT DEGREES
SET GROUND ALTITUDE TWENTY FEET

Language model

The language model built based on the corpus/training set is an n-gram word-based model. The inputs required to generate the language model are the corpus, the dictionary and suitable value for n which determines the maximum value of the gram for which the model is built. Commanding a MAV is a one way communication between a human and a machine, hence the need to consider any OOV in speech recognition does not arise. Extra effort has been taken to ensure that the test set does not contain any out of vocabulary (OOV) words. The language model is created using CMU Sphinx lmtool [10].

A dictionary and a grammar file are manually generated for the command set. The dictionary contains all the words along with their phonemes used in the spoken command. A grammar, which helps in recognition, is framed based on the format of MAV commands [2].

A trigram (n=3) language model is generated as the commands are short. The number of generated unigrams is 82. The number of generated bigrams is 327. The number of generated trigrams is 429. The probability scores of each of the gram help in determining the probably near match to the words in the dictionary and the commands in the corpus.

Simulation Studies

The simulation studies have been conducted using the HTK recognition tool. The goodness of any language model is based on its perplexity i.e. how well the model fits into the speech recognition system. The perplexity of the developed language model is determined by LPlex tool of HTK. Lower the perplexity better is the language model. The perplexity of the language model developed is 4.48. The format of the LPlex tool is as follows:

```
>> LPlex -n 3 -t trigram.lm gcs.txt
```

where n is the value of the gram, t is for printing the output as a text stream, gcs.txt is a text file containing test data.

HDecode toolbox of HTK toolkit is used recognition of speech using both acoustic and language models. It uses (Large Vocabulary Continuous Speech Recognition) LVCSR decoding. The format of the HDecode is as follows:

```
>> hdecode.mod -H model/hmm21/macros -H model/hmm21/hmmdefs -S test.txt -t 220.0 220.0 -C config.hdecode -i recount.mlf -w trigram.lm -p 0.0 -s 5.0 dict.hdecode tiedlistmbt
```

The main inputs for this tool are the HMM acoustic model (model/hmm21/macros,model/hmm21/hmmdefs) the language model(trigram.lm) and the acoustic wave files(test.txt contains the paths to the acoustic wave files). Apart from these the other inputs are:

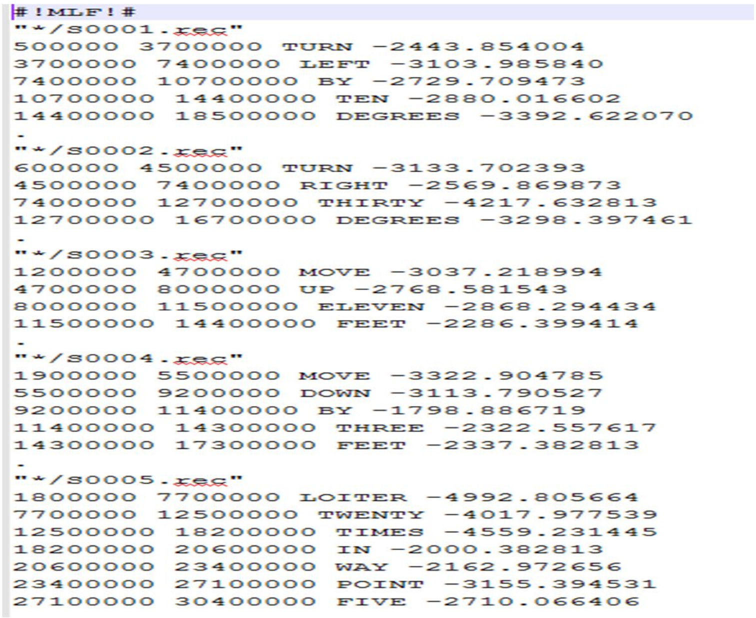
a. Parameter insertion penalty *p,* whose value is set to 0.0 which is a constant added to each token when it transits from one word to another.
b. Grammar scale factor *s,* it is the amount by which the language model probability is scaled before being added to each token as it transits from the end of one word to the start of the next. This parameter value is varied to get the best possible result. It balances the acoustic and the language model [13].
c. The triphone tied list *tiedlistmbt* has been used as a reference i.e., how every word is made up of triphones. Triphone means a sound is made up of 3 phonemes. To use the HDecode recognition tool, the cross-word triphones models have to be used. A cross-word triphone list has been created manually and used in the Hdecode.

d. *t* is the value for main beam pruning parameter which is used to adjust the run time.

e. A configuration file *config.hdecode* which contains the various HDecode parameters.

f. *dict.hdecode* is the dictionary along with phonemes.

To understand the triphones in detail, consider the word TURN, the phonemes for that word are *sil+t+er+n+sp* where *sil* represents silence and *sp* represents pause. The cross-word triphones means the previous word's last phoneme and the next word's first phoneme are tied to get a better clarity apart from the words triphones. For example, the triphones for word sequence "TURN LEFT" is
sil-t+er t-er+n er-n+l l-eh+f eh-f+t f- t+sp
Here the cross-word triphone is er-n+l, which is between the words TURN and LEFT. The output of HDecode is a mlf which consists of the recognized commands. The snapshot of the resultant mlf is shown in Fig. 3.



```
#!MLF!#
"*/s0001.rec"
500000 3700000 TURN -2443.854004
3700000 7400000 LEFT -3103.985840
7400000 10700000 BY -2729.709473
10700000 14400000 TEN -2880.016602
14400000 18500000 DEGREES -3392.622070
.
"*/s0002.rec"
600000 4500000 TURN -3133.702393
4500000 7400000 RIGHT -2569.869873
7400000 12700000 THIRTY -4217.632813
12700000 16700000 DEGREES -3298.397461
.
"*/s0003.rec"
1200000 4700000 MOVE -3037.218994
4700000 8000000 UP -2768.581543
8000000 11500000 ELEVEN -2868.294434
11500000 14400000 FEET -2286.399414
.
"*/s0004.rec"
1900000 5500000 MOVE -3322.904785
5500000 9200000 DOWN -3113.790527
9200000 11400000 BY -1798.886719
11400000 14300000 THREE -2322.557617
14300000 17300000 FEET -2337.382813
.
"*/s0005.rec"
1800000 7700000 LOITER -4992.805664
7700000 12500000 TWENTY -4017.977539
12500000 18200000 TIMES -4559.231445
18200000 20600000 IN -2000.382813
20600000 23400000 WAY -2162.972656
23400000 27100000 POINT -3155.394531
27100000 30400000 FIVE -2710.066406
```

Figure 1. Snapshot of the output mlf file

The HTK command HResults is executed on the output mlf file which contains the resultant recognition of the test set. It gives the percentage of word correctness, sentence correctness, accuracy, the number of word and sentence insertion, deletion and substitution. By using the language model component, the results are found to be better and they are tabulated in **Error! Reference source not found.**.

Table 1. Effect of Language model on recognition

Scale Factor variation(*s*)	Word correctness (%)	Sentence correctness (%)	Accuracy (%)
5.0	95.15	73.33	92.12
10.0	97.58	86.67	96.67
15.0	98.18	90.00	97.58
20.0	100.0	100.00	100.00
25.0	100.0	100.00	100.00

In [2], the maximum word correctness recognition (WCR) and the sentence correctness recognition (SCR) for a sample command achieved were 96.78% and 83%. With the use of language model, it has been found that for values of *s* between 20.0 and 25.0 the maximum recognition rate has been achieved. This could be achieved as there were no insertions, deletions or substitutions. The effectiveness of the language model usage is clearly brought out with the examples in Table 2.
In Example 1, language model resolved the conflict between **FOURTEEN** and **FORTY**. In Example 2 also, the inclusion of language model resulted in the correct recognition. Thus the importance of Language model in an ASR in the MAV context is clearly brought out.

Table 2. Improvement in Recognition with Language Model usage

	Example 1	Example2
Command	TURN LEFT BY FOURTEEN DEGREES	TAKE OFF FROM NINETEEN FEET
Only Acoustic model	TURN LEFT BY FORTY DEGREES	TAKE OFF FROM NINETEEN FOUR EIGHT
With Language model	TURN LEFT BY FOURTEEN DEGREES	TAKE OFF FROM NINETEEN FEET

Conclusion

This paper highlighted the importance of using a language model in the ASR designed for voice controlled MAV application. The initial part of the paper focused on the formulation of MAV command set and creation of the language model for this set. The later part proved the efficacy of this language model in achieving accurate recognition of the commands.

References

[1] Rahul, D. K., S Veena et al., "Development of Voice Activated Ground Control Station." Procedia Computer Science 89 (2016): 632-639.

[2] Sandhyalakshmi, S Veena, Lakshmi P et al., “Development of Continuous Automatic Speech Recognition System for Controlling of MAVs through Natural Speech”

[3] Mirzaei, Saeideh., "Improving Accuracy in Automatic Speech Recognition Systems by Model Adaptation Techniques." (2015).

[4] Kim, Woosung, and Sanjeev Khudanpur., "Language model adaptation for automatic speech recognition and statistical machine translation." Johns Hopkins University, Baltimore, MD (2005).

[5] Whittaker, Edward William Daniel., “Statistical language modeling for automatic speech recognition of Russian and English.” Diss. University of Cambridge, 2000

[6] Pauls, Adam, and Dan Klein., "Faster and smaller n-gram language models." Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. Association for Computational Linguistics, 2011.

[7] http://ardupilot.org/planner/docs/mission-planner-overview.html

[8] Larson, Martha., "Sub-word-based language models for speech recognition: implications for spoken document retrieval." Workshop on Language Modeling and Information Retrieval (2001).

[9] http://htk.eng.cam.ac.uk/

[10] http://cmusphinx.sourceforge.net/

[11] Craparo, Emily Marie., “Natural language processing for unmanned aerial vehicle guidance interfaces” Diss. Massachusetts Institute of Technology, 2004.

[12] http://ardupilot.org/copter/docs/mission-command-list.html

[13] Sanchis, Alberto, Vıctor Jiménez, and Enrique Vidal., "Efficient use of the grammar scale factor to classify incorrect words in speech recognition verification." Pattern Recognition, 2000. Proceedings.15th International Conference on.Vol.3.IEEE, 2000.