Robotic Hearing in Noise

L.J Martinus*, J.J. Hanekom†
*Email: LMartinus@csir.co.za
†Email: johan.hanekom@up.ac.za
Dept. of Electrical, Electronic and Computer Engineering
University of Pretoria, South Africa

***Abstract*—In this paper, the problem of speech recognition in noise is considered. The focus of this project was developing a system that uses voice instructions to control a robot in a noisy environment. Speech enhancement of the noisy speech takes place through the use of a Kalman filter, which provides a significant improvement in the quality and intelligibility of the signal. Speech recognition takes place through the use of Hidden Markov models with mel-frequency cepstral coefficients used for feature extraction. The overall accuracy of the developed system is 64% and a 0% error rate when tested with input signals with signal-to-noise ratios varying between -15 dB and 30 dB of additive white Gaussian noise. This paper provides the detailed design and implementation of the system.**

***Index Terms*—Speech enhancement, Kalman filter, speech recognition, Hidden Markov models, mel-frequency cepstral coefficients.**

I. Problem identification

Voice recognition is an emerging and fast growing field in modern technology. Utilising speech recognition inside any application or technological device increases its usability for both abled and disabled persons. The use of robotics has grown significantly in recent times: military examples include four legged robotic pack animals; manufacturing examples include automated welding machines; examples in personal life include companion toys which provide emotional support. Coupling speech recognition with robotics gives an added communication channel to such robotic devices. The environment where these robotic devices operate (i.e. the battle field or a factory), however, are rarely silent. Being able to extract verbal commands from a noisy background is critical.

The aim of this project was to develop a system that uses voice instructions to control a robot in a noisy environment. The system was designed to recognise voice instructions to a robot mixed with real environmental noises, as such the system needs to be viable for a real world implementation. The system consists of a head mounted on a platform that can be tilted up and down and can be turned left and right. The robotic head was designed to respond to simple voice commands, originating from anywhere in the robot's auditory field.

This system entails the design of both hardware and software, however, the focus of the task is mainly in the design of the speech recognition algorithm receiving commands within a noisy environment.

The main areas of focus within this system were the following.

- The elimination of noise from the obtained signal.
- The speech recognition algorithm.
- The accuracy of the system.

Due to the presence of noise, speech enhancement methods were vital to the success of the speech recognition. The quality and intelligibility of the speech needed to be improved. Various speech enhancement methods have been proposed in the past, each with their own advantages and disadvantages.

According to [1], Wiener filters are frequently considered as a fundamental method of noise reduction. However, these Wiener filters are known to have an adverse effect on the speech signal's quality. [1] shows that the reduction of noise due to an optimal Wiener filter is proportional to the degradation of the speech signal's quality. This degradation may be decreased through the use of a suboptimal filter and/or the use of multiple microphones. Wiener filters do not perform well on speech because the filter is stationary whilst speech is not.

Unlike the Wiener filter, non-stationary speech can be filtered through a Kalman filter. [2] confirmed that the use of a Kalman filter provides a advantage over the use of a Wiener filtering method when tested with speech. [2] provides a comparison between the Wiener filter and Kalman filter speech enhancement SNR values. A downside to using a Kalman filter over a Wiener filter is that the Kalman filtering method is computationally more complicated. Due to the advantage in enhancement of speech, the Kalman filtering process was decided on as the main speech enhancement process in the system.

Once the speech signal has been enhanced, the system can begin attempting recognition. Hidden Markov models (HMM), as described in [3], when used for speech recognition, can provide a 98% recognition accuracy. If the begin and end points of the speech signal are known, the use of the left-to-right HMM is recommended. HMMs can be used to recognise distinctive periods in a speech signal, as well as use the duration of states (obtained through segmentation) to recognise speech.

Feature extraction provides the observation sequence for HMMs. Various feature extraction methods are available. [4] provides a table of the current feature extraction methods. [4] concluded that mel-frequency cepstral coefficient (MFCC) method is the most commonly used and HMMs could provide the best result when compared to the other modelling techniques. Due to this conclusion, the speech recognition system

978-1-5386-2775-4/17/$31.00 ©2017 IEEE

designed makes use of these methods in the implementation and construction.

II. Methods

The design and implementation of the system is divided into three main parts; speech enhancement, speech recognition, and the robotic head. The input signal (the sound to be recognised) is enhanced in two steps: first through an Finite Impulse Response (FIR) low-pass filter, and then through a Kalman filter. The speech recognition takes place through the use of HMMs and MFCC feature extraction. The robotic head consists of a microphone, a DSP board, and two stepper motors.

A. Speech Enhancement

An FIR low-pass digital filter removes the high frequency components of the input signal.

$$y(n) = \Sigma_{k=0}^{N-1} h(k)x(n-k) \tag{1}$$

An FIR filter, characterised by **Eq. 1**, was designed by first determining the specifications. The following specifications were required for the system.

- The type of filter needed was a low-pass filter,
- a cut-off frequency of 3.4 kHz and
- a sampling frequency of 20 kHz were used.

The window method for coefficient calculation was used. The impulse response for the ideal low-pass filter is given by **Eq. 2**.

$$h_d(n) = \frac{2f_c sin(n-\omega_c)}{n\omega_c} \tag{2}$$

The Blackman window function, described by **Eq. 3**, was used.

$$w(n) = 0.42 - 0.5cos(\frac{2\pi n}{N-1}) = 0.08cos(\frac{4\pi n}{N-1}) \tag{3}$$

The values of the coefficients, $h(n)$ (**Eq. 4**), is calculated by multiplying the ideal impulse response (**Eq. 2**) with the window function (**Eq. 3**).

$$h(n) = h_d(n)w(n) \tag{4}$$

The coefficients obtained by this calculation are then convolved with the input signal according to **Eq. 1** in order to obtain the filtered signal.

In order to use the Kalman Filter on the speech signal, segmentation, as described in [5], of the signal first takes place. The input signal is normalised to have a zero mean using the equation

$$norm_i = \frac{x_i - mean}{std}. \tag{5}$$

The normalised signal is cut into overlapping windows. The overlap factor used is $s = 4$ and the segment length used is $T = 12$.

Auto Regressive (AR) processes model time-varying data in many areas of study. Speech is one of these areas that can be represented by an AR model as explained by [6]. This model can be described by **Eq. 6**. The AR coefficients can be generated using MATLAB's *ar()* function, using the Yule-Walker approach.

$$s(k) = a_k s(k-1) + ... + a_p s(k-p) + u(k) \tag{6}$$

This system can be represented in state-space as described by **Eq. 7**.

$$\begin{bmatrix} s(k-p+1) \\ s(k-p+2) \\ \vdots \\ s(k) \end{bmatrix} = A \begin{bmatrix} s(k-p) \\ s(k-p+1) \\ \vdots \\ s(k-1) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} u(k) \tag{7}$$

$$A = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -a_p & -a_{p-1} & \dots & \dots & -a_1 \end{bmatrix} \tag{8}$$

This A matrix (**Eq. 8**) is used as the A matrix in the Kalman filter process described below, in accordance with [2].

The process description and equations as obtained from [2] are described for the Kalman filtering process. The initial state matrix X_0 is initialised to zero and the process covariance matrix P_0 to the initial error in the estimates.

The initial state becomes the previous state $(k-1)$, leading to X_{k-1} and P_{k-1}. The predicted state (kp) is calculated by **Eq. 9**.

$$X_{kp} = AX_{k-1} + Bu_k + w_k \tag{9}$$
$$P_{kp} = AP_{k-1}A^T + Q_k$$

where *u* is the control variable matrix, *w* is the predicted state noise matrix, and *Q* is the process noise covariance matrix. The measured data X_{km} is input to the system using **Eq. 10**.

$$Y_k = CX_{km} + z_k \tag{10}$$

where z_k is the measurement noise. The Kalman gain *K* and the new measurement X_k is obtained through **Eq. 11** and **Eq. 12**.

$$K = \frac{P_{kp}H^T}{HP_{kp}H^T + R} \tag{11}$$

$$X_k = X_{kp} + K(Y_k - HX_{kp}) \tag{12}$$

The process covariance matrix is updated through **Eq. 13**.

$$P_k = (I - KH)P_{kp} \tag{13}$$

X_k and P_k become X_{k-1} and P_{k-1} for the next iteration. The above processes are repeated for all the input data.

B. Speech Recognition

MFCC collectively make up a mel-frequency cepstrum (MFC) and are used as the feature extraction for the system. An MFC represents a short term power spectrum of a sound. This representation is built on a linear cosine transform calculated from the logarithm of the power spectrum of the sound, represented on a non-linear mel-frequency scale. The mel-scale of frequency equation is obtained from **Eq. 5.13** of [7].

$$mel(f) = 2595 log_{10}(1 + \frac{f}{700}) \tag{14}$$

On short time scales, an audio signal becomes statistically stationary. For this reason, the signal, after normalisation, must be framed into shorter frames of 20-40 ms. This ensures that the signal does not change significantly throughout the frame, and that there are enough samples to obtain a spectral estimate that is reliable. Each frame is multiplied by the Hamming function (**Eq. 15**) in order to reduce the discontinuities in the windows.

$$x'_n = (0.54 - 0.46cos(\frac{2\pi(n-1)}{N-1})x_n) \tag{15}$$

The frequency spectrum of each frame is calculated in order to identify of which frequencies the frames are composed. The magnitude of the Fourier transform is taken.

The mel filterbank (H) is calculated and multiplied with the frequency spectrum to determine the filterbank energies. The filterbank is a set of 26 triangular filters. H is a matrix of 26 vectors, each of the length of the Fourier transform. In order to calculate the filterbank, b, c, and F first need to be calculated.

$$b = ((upper\_mel - lower\_mel)/(26 + 1)); \tag{16}$$

$$c[i] = (exp((lower\_mel + (i \times b))/1127) - 1) \times 700; \tag{17}$$

$$F[i] = (i \times 2.0/NFFT) \times Fs/2; \tag{18}$$

where *upper_mel* and *lower_mel* are the lower and upper frequency cut-offs in the mel-scale, and $NFFT$ is the length of the Fourier transform. The equation of for H (**Eq. 19**) is obtained from **Eq. 141** of [8].

$$H_m[f] = \begin{cases} 0, & f < c[m-1] \\ \frac{f - c[m-1]}{c[m] - c[m-1]}, & c[m-1] \le f \le c[m] \\ \frac{c[m+1] - f}{c[m+1] - c[m]}, & c[m] \le f \le c[m+1] \\ 0, & f > c[m+1] \end{cases} \tag{19}$$

The Discrete Cosine Transform (DCT) of the log filterbank energies is calculated by **Eq. 20** in order to decorrelate the coefficients. This ensures that diagonal covariance matrices can later be used for modelling of the features.

$$c(i) = \sqrt{\frac{2}{N}} \sum_{N}^{j=1} m_j cos(\frac{\pi i}{N}(j - 0.5)) \tag{20}$$

A lifter routine is used on the DCT to result in a smoother signal. The cepstral lifter routine (**Eq. 21**) is obtained from **Eq. 5.12** of [8].

$$c'_n = (1 + \frac{L}{2} sin(\frac{\pi n}{L}))c_n \tag{21}$$

Due to the fact that the common C programming language libraries do not include a Fourier transform function, a function was created to perform an FFT on the data. An FFT function was written from first principles described in [9] in order to obtain the frequency spectra of the created data. The FFT function's number of points needs to be a power of 2, in order to obtain this the file is zero-padded to the nearest power of 2 higher than the file size. Shuffling of data then occurs through in-place bit reversal. The number of stages is calculated and then for each stage the FFT computation is performed by calculating the weight factors and the butterfly computation is done. The resultant magnitudes are stored as the frequency spectrum.

To create an isolated word recogniser using HMMs for the specified words, the following process must be followed. For each of the words, represented by *v*, in the vocabulary, an HMM λ^v must be built by estimating the values of the parameters (A, B, π) (**Eq. 22**). The following equations were obtained from [3].

$$\lambda = (A, B, \pi) \tag{22}$$

O represents the input observation sequence, with *M* as the number of distinct observation symbols. N is the number of individual states (*S*) in the model, in this case three. The state at time *t* is denoted as q_t. The individual symbols are denoted as *V*. *A* is the state transition probability distribution and is calculated by **Eq. 23**.

$$A = a_{ij} \tag{23}$$
$$a_{ij} = P[q_{t+1} = S_j | q_t = S_i]$$
$$1 \le i, j \le N$$

B is the observation symbol probability distribution, and is calculated by **Eq. 24**.

$$B = b_j(k) \tag{24}$$
$$b_j(k) = P[v_k att | q_t = S_j]$$
$$1 \le j \le N, 1 \le k \le M$$

π is the initial state distribution and is calculated by **Eq. 25**.

$$\pi = \pi_j \tag{25}$$
$$\pi_j = P[q_1 = s_i]$$
$$1 \le i \le N$$

The training of the model progresses as follows. The parameters are initialised at random; π is initialised to the normalised values of randomly assigned integer values up to the number of states; A is initialised to the stochastic values of randomly assigned integer values up to the number of states; the mean matrix is initialised to random data points of the observation; the variance matrix is initialised to a 3D matrix of the empirical diagonal covariance matrix of the observation.

The model of each word is trained with 30 variations of the same word with varying levels of Additive White Gaussian Noise (AWGN). The training function starts by calculating the state likelihood, a matrix of the multi-variant normal probability distribution of the observation sequence with the model's mean and variance matrices. The α and β matrices are calculated by calling the *forwards* and *backwards* function. The normalised γ matrix is calculated by **Eq. 26**.

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\Sigma_{i=1}^{N}\alpha_t(i)\beta_t(i)} \tag{26}$$

The model of the word is updated in the following manner. The previous A matrix is multiplied by the α, β, and state likelihood matrices. The sum of the normalised results becomes the new A matrix. The π matrix is set to the first vector of the γ matrix.

The variance must be ensured for positive semi-definiteness in order to avoid NaN results.

The *forwards* algorithm calculates the α matrix using **Eq. 27** and **Eq. 28** for initialisation and induction.

$$\alpha_1(i) = \pi_i b_i(O_1) \tag{27}$$
$$1 \leq i \leq N$$
$$\alpha_{t+1}(j) = [\Sigma_{i=1}^{N}\alpha_t(i)a_{ij}]b_j(O_{t+1}) \tag{28}$$
$$1 \leq t \leq T-1, 1 \leq j \leq N$$

The *log likelihood* is calculated by summing the log of the scaling factor of each iteration.

The *backwards* algorithm calculates the β matrix using **Eq. 29** and **Eq. 30**.

$$\beta_T(i) = 1 \tag{29}$$
$$1 \leq i \leq N$$
$$\beta_t(i) = \Sigma_{j=1}^{N} a_{ij} b_j(O_{t+1})\beta_{t+1}(j) \tag{30}$$
$$t = T-1, T-2, ..., 1, 1 \leq i \leq N$$

At the end of the training process, each model's parameters are stored for use in the prediction phase.

To predict the input word, the *log likelihood* of the observation to the HMM model of all the words in the vocabulary is calculated. The model which has the highest *log likelihood* is the word that is predicted. If the maximum *log likelihood* is close in value to the second maximum *log likelihood*, the system does not predict a word but rather requests the user to repeat what was input. The process of predicting the word is done by calculating the state likelihood by using the multi-variant normal probability distribution function, and by using the *forwards* algorithm as described previously.

C. Robotic Head

The robotic head is the hardware implementation of the system. The functions for speech enhancement and speech recognition run on the DSP board, and the output results in the movement of the stepper motors in the specified manner. The DSP board used for the system is the STM 32F429iDISCOVERY. Two stepper motors and two stepper motor drivers facilitate the movement of the head. The input words are enhanced and recognised. Depending on the recognised command, one stepper motor is turned on to create the "tilt" or "turn" motion, in the specified direction (up/down and left/right), at the specified speed of "fast" or "slow".

A microphone was connected to the ADC port of the board, and sampled the input signal at 8 kHz. The lower sampling frequency decreases the computational complexity of the program, allowing the board to sufficiently process the input signal. The values are stored in the data type form of float instead of double. This decreases the accuracy of the system, but takes up less memory in the system.

III. Results

The system performs at

- 100% accuracy for an SNR value of 30 dB,
- 93% accuracy for an SNR value of 10 dB,
- 49% accuracy for an SNR value of 0 dB,
- 0% accuracy for an SNR value of -15 dB.

The overall accuracy of the system is 64% with a 0% error rate. **Fig. 1** depicts the overall accuracy of the system.

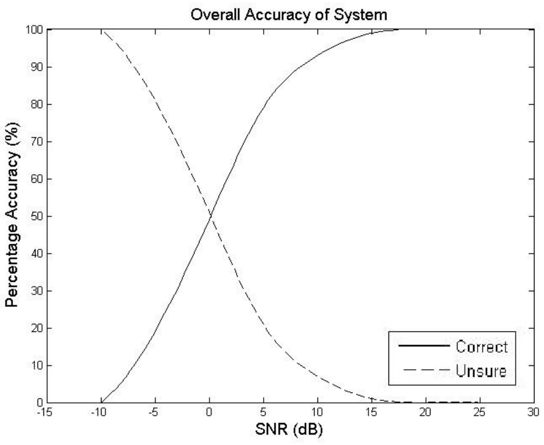

Fig. 1. Overall accuracy of the system

On average, the input speech is improved from:

- -10 dB SNR to 0.1 dB SNR,
- 0 dB SNR to 8 dB SNR,
- 10 dB SNR to 16.9 dB SNR,

when tested with AWGN through the final speech enhancement system using AR models. The average improvement is 7.4 dB SNR. **Fig. 2** depicts the speech enhancement of the final system when compared to no enhancement.

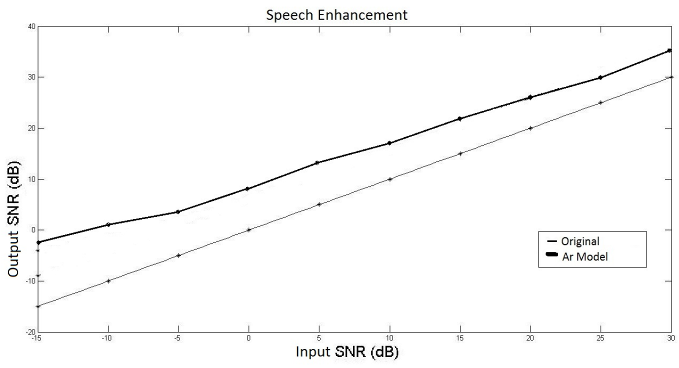

Fig. 2. Speech enhancement improvement results

TABLE I
SPECIFICATION ADHERENCE

Accuracy Required	SNR (dB)	Actual Accuracy	Requirement Satisfied?
100%	30	100%	yes
85%	10	93%	yes
$>$ 70%	0	49%	no
70%	-15	0%	no

The speech recognition system performs at

- 100% accuracy for an SNR value of 30 dB,
- 100% accuracy for an SNR value of 10 dB,
- 63% accuracy for an SNR value of 0 dB,
- 12% accuracy for an SNR value of -15 dB.

The overall accuracy of the speech recognition system is 72% with an error rate of 28% when tested with an input noise range from -15 dB SNR to 30 dB SNR.

Incorrectly recognised words were deemed as more detrimental to the system than accuracy of the system at lower SNR levels, so the speech recognition algorithm was altered to rather request the user to repeat the word that was input rather than incorrectly predict the word. This alteration lead to a lower accuracy of the speech recognition, but with a 0% error rate. The final speech recognition algorithm has an accuracy of 49% when tested with an input noise range from -15 dB SNR to 30 dB SNR and a 0% error rate.

The DSP board runs a simplified version of the speech recognition process by decreasing the sampling frequency and using less memory extensive data types. Once the speech has been recognised, in order to depict the movement of the robotic head, the DSP board controls the movement of the head in a left-right-up-down motion.

IV. Discussion

The specifications set out for the overall system's performance as well as the adherence to these specifications is outlined in **Table I**. The overall accuracy of the system is 64% with a 0% error rate.

As can be seen in **Table I**, the system adheres to specification greater than expected for SNR values higher than 5 dB. The system performs below specification for the SNR value of -15 dB. The overall system is has an error rate of 0%. The system will constantly request the user to repeat what was input at SNR levels lower than -5 dB.

A. Strong Points

The speech enhancement system performs consistently with an average increase of 7.4 dB.

The speech recognition system performs consistently with an accuracy of 72% with an input noise range from -15 dB to 30 dB SNR, as well as an accuracy of higher than 80% for all input SNR values higher than 5 dB. The final speech recognition system has a lower accuracy of 49% but performs with a 0% error rate.

The overall accuracy of the system is 81% with an input noise range from -15 dB to 30 dB SNR, as well as an accuracy of higher than 80% for all input SNR values higher than 0 dB. The final system performs at a 0% error rate, at 64% accuracy with an input noise range from -15 dB to 30 dB, as well as an accuracy of higher than 79% for all input SNR values higher than 5 dB.

B. Aspects to be Improved

The system would work with greater accuracy if the prediction of the speech could perform better at lower levels of SNR. The accuracy of the speech recognition could possibly be improved through the the use of more training data with greater variations in noise.

Due to the computational complexity of the program, the version of the program running on the DSP board is simplified and provides a lower accuracy. The system could be improved by increasing the computational power of the board upon which it executes.

V. Conclusion

A. Summary of the Results

The overall accuracy of the system is 64% with a 0% error rate. The average improvement in SNR for the final speech enhancement system is 7.4 dB. The speech recognition algorithm performs at an accuracy of 49% when tested with an input noise range from -15 dB to 30 dB SNR. The system performs adequately in terms of accuracy, and will not incorrectly predict a word; in the case of uncertainty, a request for repeat input is made.

This system has no current impact on health and safety aspects. Furthermore, robotic systems can be used to improve the health and safety aspects of many work environments, especially industrial environments as the robotic systems can be used to perform unsafe and repetitive tasks.

Robotic systems that can understand and respond to a user's command have many social benefits. These benefits could be in terms of factory robots performing dangerous jobs, military robots used as pack animals or defusing explosives, and personal companions for the use of emotional support or every day household chores. Many more socially beneficial uses of robots exist.

Legal and ethical problems come into play for many reasons. The reasons include the increased number of robots in industry with the subsequent problem of increased human unemployment. In addition as robots become more intelligent, the problem of controlling such robots and the problem of ensuring that such robots do not put humans in danger need to be solved.

References

[1] J. Chen, J. Benesty, Y. Huang, and S. Doclo, "New insights into the noise reduction wiener filter," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, pp. 1218–1234, 2006.

[2] K. Paliwal and A. Basu, "A speech enhancement method based on kalman filtering, proceedings of ieee int," in *Conf. Acoust. Speech*, 1987.

[3] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[4] S. K. Gaikwad, B. W. Gawali, and P. Yannawar, "A review on speech recognition technique," *International Journal of Computer Applications*, vol. 10, no. 3, pp. 16–24, 2010.

[5] J. Kybic, "Kalman filtering and speech enhancement," *Diploma w0rk, Ecole polytechnique federale De Lausanne*, 1998.

[6] M. A. Berezina, D. Rudoy, and P. J. Wolfe, "Autoregressive modeling of voiced speech," in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2010, pp. 5042–5045.

[7] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey *et al.*, "The htk book," *Cambridge university engineering department*, vol. 3, p. 175, 2002.

[8] X. Huang, A. Acero, H.-W. Hon, and R. Foreword By-Reddy, *Spoken language processing: A guide to theory, algorithm, and system development*. Prentice hall PTR, 2001.

[9] E. C. Ifeachor and B. W. Jervis, *Digital Signal Processing: A Practical Approach*, 2nd ed. Prentice-Hall International, 2002.