VOICE CONTROLLED CAMERA ENABLED ROBOT

Submitted in partial fulfillment of the requirements
of the degree of

Bachelor in Engineering

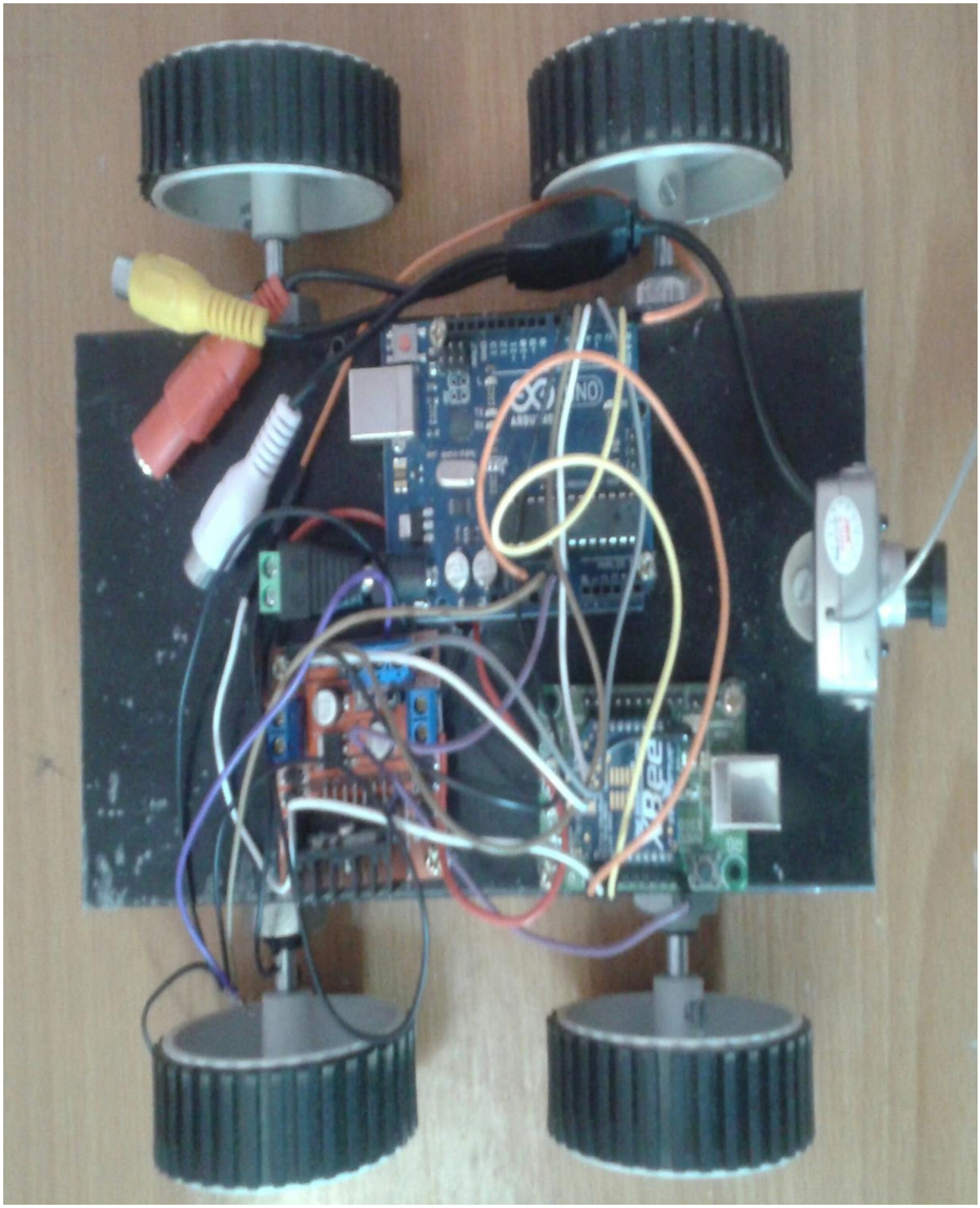
By

KHAN AZHAR WAHID ABDUL WAHID 12ET26
KHAN FUWAD AHMAD MOHD. TAQUI 12ET27
KHAN MOHD. AMIR ABDUL HAMEED 12ET30

Supervisor (s):
Asst. Prof. SHAHEBAZ M. ANSARI

Department of Electronics and Telecommunication Engineering
Anjuman-i-Islam Kalsekar Technical Campus,
New Panvel

2015-2016



Project Report Approval for B.E

This project report entitled ***Voice Controlled Camera Enabled Disaster Management Robot*** by ***Khan Azhar Wahid Abdul Wahid, Khan Fuwad Ahmad Mohd. Taqui, and Khan Mohd. Amir Abdul Hameed*** is approved for the degree of ***Bachelor in Engineering***.

Examiners:

1._____

2._____

Supervisor(s):

1._____
Asst. Prof. SHAHEBAZ M. ANSARI

H.O.D(EXTC):

_____
Asst. Prof. MUJIB A. TAMBOLI

Date:

Place:

Declaration

We declare that this written submission represents our ideas in our own words and where other sides or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

1._____
KHAN AZHAR WAHID ABDUL WAHID.
12ET26

2._____
KHAN FUWAD AHMAD MOHD. TAQUI.
12ET27

3._____
KHAN MOHD AMIR ABDUL HAMEED.
12ET30

Date:

Place:

ACKNOWLEDGEMENT

We appreciate the beauty of a rainbow, but never do we think that we need both the sun and the rain to make its colors appear. Similarly, this project work is the fruit of many such unseen hands. It's those small inputs from **Asst. Prof. ZARRAR KHAN** and **Asst. Prof. AFZAL SHAIKH** as well as some different people that have lent a helping to our project.

I take this opportunity to express my profound gratitude and deep regards to my guide **Asst. Prof. SHAHEBAZ M. ANSARI** for his exemplary guidance, monitoring and constant encouragement throughout the course of this project work.

I also take this opportunity to express a deep sense of gratitude to **Asst. Prof. MUJIB A. TAMBOLI**, HOD of E.X.T.C. Dept. for his cordial support, valuable information and guidance, which helped me in completing this task through various stages.

I am obliged to staff members of **AIKTC**, for the valuable information provided by them in their respective fields. I am grateful for their cooperation during the period of my project work.

ABSTRACT

Disaster management is the main objective of this project besides surveillance and security purposes. The aim of the project is to find or search the people who are stuck due to some kind of disaster using a robot mechanism. The project mainly is based on speech processing and wireless communication. In this project, at the user end, the user will be feeding the voice signals or commands as the input via a microphone connected to a laptop or desktop.

These voice signals will further be processed by using the software, MATLAB. The robot will be trained using LPC and artificial neural network algorithm. Then, the processed voice signals will be transmitted through air via Zigbee module. The transmitted signals will be received via Zigbee module at the Robot end. The controller at the robot end will decode the signals received and will match the same with the ones stored in the Robot controller and thus the Robot will be controlled via voice signals.

There is a camera which is to be fixed at the robot end. This camera will give a real-time video output to the user on the laptop or computer via a small GUI-graphic user interface which is to be built in MATLAB. Thus the people trapped due to some disaster can be searched by use of the camera in the project.

This project can also be served for security and surveillance applications.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

CHAPTER 1

INTRODUCTION

The project "Voice controlled Camera enabled Robot" is basically Robot mechanism which is controlled by a user, using Voice commands. It also has a camera fixed on it so as to get the real time video data.

In this project, at the user end, the user will be feeding the voice signals or commands as the input via a microphone connected to a laptop or desktop. The commands will be recorded and processed in software called as MATLAB. The project is mainly based on speech processing, neural networks and wireless communication. The method by which the speech processing and recognition is carried out is Mel Frequency Cepstral Co-efficient (MFCC). The robot will be trained using Artificial Neural Networks in such a way that only one user can input the commands to the robot. This will make the project more secure and intrusion free.

Then, the processed voice signals will be transmitted through Zigbee module. The transmitted signals will be received via Zigbee module at the Robot end. The Arduino controller at the robot end will send the signals received from Zigbee to the Motor Controller and accordingly the motors will be controlled. Thus the robot will be controlled via voice signals. There will be a camera fixed at the robot end which will give a real-time video output to the user on the laptop or computer via a small Graphic User Interface which is built in MATLAB.

1.1 THEORY BEHIND THE PROJECT:

The project is mainly based on Speech Recognition. So, the speech of the user has to be recognized by the robot for its functioning. To make this possible, it is required to process the input speech signals and extract its features. There are various techniques for feature extraction, of which Mel-Frequency Cepstrum Coefficients is used in this project along with Feed-Forward Back propagation neural networks for classification and feature matching.

1.1.1 MFCC:

In sound processing, the **mel-frequency cepstrum** (**MFC**) is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency.

Mel-frequency Cepstrum Coefficients (**MFCCs**) are coefficients that collectively make up an MFC. They are derived from a type of cepstral representation of the audio clip (a nonlinear "spectrum-of-a-spectrum"). The difference between the cepstrum and the mel-frequency cepstrum is that, in the MFC, the frequency bands are equally spaced on the mel scale, which approximates the human auditory system's response more closely than the linearly-spaced frequency bands used in the normal cepstrum. This frequency warping can allow for better representation of sound, for example, in audio compression. MFCCs are commonly derived as follows:

1. Take the Fourier transform of (a windowed excerpt of) a signal.
2. Map the powers of the spectrum obtained above onto the mel scale, using overlapping windows.
3. Take the logs of the powers at each of the mel frequencies.
4. Take the discrete cosine transform of the list of mel log powers, as if it were a signal.
5. The MFCCs are the amplitudes of the resulting spectrum.

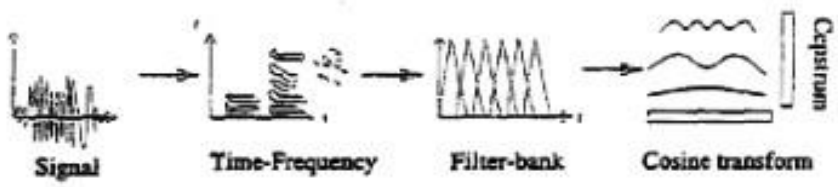


Fig1.1: Pictorial representation of MFCC

1.1.2 ANN:

In machine learning and cognitive science, **artificial neural networks** (**ANNs**) are a family of statistical learning algorithms inspired by biological neural networks (the central nervous systems of animals, in particular the brain) and are used to estimate or approximate functions that can depend on a large number of inputs and are generally unknown. An artificial neural network is an interconnected group of nodes, similar to the vast network of neurons in a brain. Here, each circular node represents an artificial neuron and an arrow represents a connection from the output of one neuron to the input of another.

For example, a neural network for handwriting recognition is defined by a set of input neurons which may be activated by the pixels of an input image. After being weighted and transformed by a function (determined by the network's designer), the activations of these neurons are then passed on to other neurons. This process is repeated until finally, an output neuron is activated. This determines which character was read.

Like other machine learning methods - systems that learn from data - neural networks have been used to solve a wide variety of tasks that are hard to solve using ordinary rule-based programming, including computer vision and speech recognition.

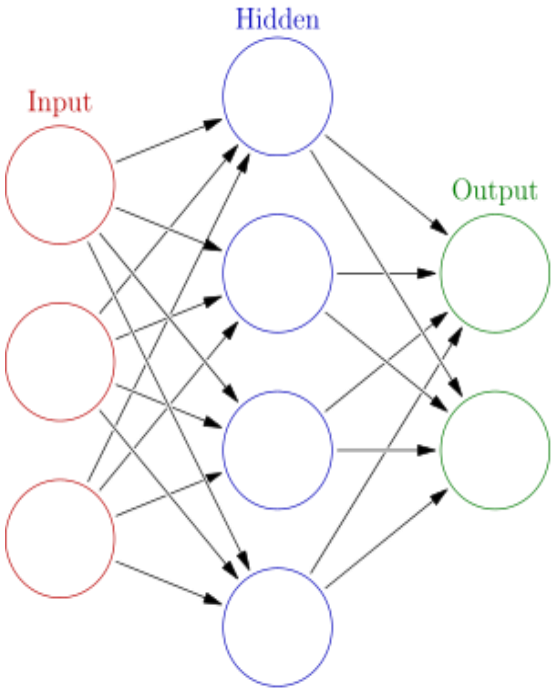


Fig1.2: ANN

1.2 NEED OF PROJECT:

The main objective of the project is that, there are some areas where human beings cannot physically go. For e.g. some temperature sensitive industrial areas, furnaces, etc. So, the project can help people reach and monitor those specific areas.

The robot is trained, so that it is secured and only one user can have command on it. So it can be helpful in military applications for spying and surveillance. Another objective of the project is to find the people who are trapped in some disasters, for ex. Floods. By getting the video output, the user can easily find out where exactly one is trapped and can send the rescue team.

CHAPTER 2

LITERATURE SURVEY

CHAPTER 2

LITERATURE SURVEY

First this project was implemented using LPC (Linear Preductive Coding) and using hamming window in 2010. But as we know that human speech is nonlinear in nature and linear preductive coding basically works on linear computation so it not to suitable. Also the hamming window which was used in this method has less noise rejection. So in our project we are using MFCC(Mel Frequency Cepstral Coefficient) and Blackmann window so we can improve its accuracy, and security. In below Table 2.1 shows that different techniques used for features extraction.

Table 2.1: Literature survey on different techniques

Author	**Year**	**Technique used for Feature Extraction**	**Technique used for Speech Classification**
Antanas Lipeika, Joana Lipeikien E, Laimutie Telksnys	2002	LPC	DTW
Talal Bin Amin, Iftikhar Mahmood	2008	MFCC	DTW
Vimal Krishnan V.R, Athulya Jayakumar, Babu Anto. P	2008	DWT	ANN
Anup Kumar Paul, Dipankar Das, Md. Mustafa Kamal	2009	LPC	ANN

Ahmad A. M. Abushariah, Teddy S. Gunawan, Othman O. Khalifa,Mohd A.M Abudharaih	2010	MFCC	HMM
R. B. Shinde, Dr. V. P. Pawar	2012	LPC	ANN

CHAPTER 3

ANALYSIS & DESIGN

CHAPTER 3

ANALYSIS & DESIGN

3.1 PRELIMINARY SURVEY:

Speech Recognition can be defined as the process of converting speech signal to a sequence of words by means Algorithm implemented as a computer program. Speech processing is one of the exciting areas of signal processing. The goal of speech recognition area is to developed technique and system to developed for speech input to machine based on major advanced in statically modeling of speech ,automatic speech recognition today find widespread application in task that require human machine interface. Speech recognition system can be separated in different classes by describing what type of utterances they can recognize.

A.Isolated Word

Isolated word recognizes attain usually require each utterance to have quiet on both side of sample windows. It accepts single words or single utterances at a time. Isolated utterance might be better name of this class.

B.Connected Word

Connected word systems are similar to isolated words but allow separate utterance to be run together with minimum pause between them.

C.Continuous speech

Continuous speech recognizers allows user to speak almost naturally, while the computer determine the content. Recognizer with continues speech capabilities are some of the most difficult to create because they utilize special method to determine utterance boundaries.

D.Spontaneous speech

At a basic level, it can be thought of as speech that is natural sounding and not rehearsed. An ASR System with spontaneous speech ability should be able to handle a variety of natural speech feature such as words being run together.

Speech Recognition is a special case of pattern recognition. There are two phase in supervised pattern recognition, viz., Training and Testing. The process of extraction of features relevant for classification is common in both phases. During the training phase, the parameters of the classification model are estimated using a large number of class examples (Training Data) During the testing or recognition phase, the feature of test pattern (test speech data) is matched with the trained model of each and every class. The test pattern is declared to belong to that whose model matches the test pattern best.

The speech recognition system may be viewed as working in a three main stages:

a) Feature extraction

b) Classification

c) Feature Matching

3.1.1. Feature extraction:

In machine learning, feature extraction starts from an initial set of measured data and builds derived values (features) intended to be informative, non-redundant, facilitating the subsequent learning and generalization steps, in some cases leading to better human interpretations. Feature extraction is related to dimensionality reduction.

When the input data to an algorithm is too large to be processed and it is suspected to be redundant, then it can be transformed into a reduced set of features (also named feature vector). This process is called feature extraction. The extracted features are expected to contain the relevant information from the input data, so that the desired task can be performed by using this reduced representation instead of the complete initial data.

Feature extraction involves reducing the amount of resources required to describe a large set of data. When performing analysis of complex data one of the major problems stems from the number of variables involved. Analysis with a large number of variables generally requires a large amount of memory and computation power or a classification algorithm which overfits the training sample and generalizes poorly to new samples. Feature extraction is a general term for methods of constructing combinations of the variables to get around these problems while still describing the data with sufficient accuracy.

There are various methods for extracting the features of the speech signals.

3.1.1.1 LPC:

Linear predictive coding (LPC) is defined as a digital method for encoding an analog signaling which a particular value is predicted by a linear function of the past values of the signal. It was first proposed as a method for encoding human speech by the United States Department of Defense in federal standard 1015, published in 1984. Human speech is produced in the vocal tract which can be approximated as a variable diameter tube. The linear predictive coding (LPC) model is based on a mathematical approximation of the vocal tract represented by this tube of a varying diameter. At a particular time, *t*, the speech sample *s(t)* is represented as a linear sum of the *p* previous samples. The most important aspect of LPC is the linear predictive filter which allows the value of the next sample to be determined by a linear combination of previous samples. Under normal circumstances, speech is sampled at 8000 samples/second with 8 bits used to represent each sample. This provides a rate of 64000 bits/second. Linear predictive coding reduces this to 2400 bits/second. At this reduced rate the speech has a distinctive synthetic sound and there is a noticeable loss of quality. However, the speech is still audible and it can still be easily understood. Since there is information loss in linear predictive coding, it is a lossy form of compression.

3.1.1.2 MFCC:

Mel Frequency Cepstral Coefficents (MFCCs) are a feature widely used in automatic speech and speaker recognition. They were introduced by Davis and Mermelstein in the 1980's, and have been state-of-the-art ever since.

In sound processing, the **mel-frequency cepstrum** (**MFC**) is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency.

Mel-frequency cepstral coefficients (**MFCCs**) are coefficients that collectively make up an MFC. They are derived from a type of cepstral representation of the audio clip (a nonlinear "spectrum-of-a-spectrum"). The difference between the cepstrum and the mel-frequency cepstrum is that in the MFC, the frequency bands are equally spaced on the mel scale, which approximates the human auditory system's response more closely than the linearly-spaced frequency bands used in the normal cepstrum. This frequency warping can allow for better representation of sound, for example, in audio compression.

A. Use of MFCC in speech recognition**:**

An audio signal is constantly changing, so to simplify things we assume that on short time scales the audio signal doesn't change much (when we say it doesn't change, we mean statistically i.e. statistically stationary, obviously the samples are constantly changing on even short time scales). [13]This is why we frame the signal generally into 20-40ms frames. If the frame is much shorter we don't have enough samples to get a reliable spectral estimate, if it is longer the signal changes too much throughout the frame.

The next step is to calculate the power spectrum of each frame. This is motivated by the human cochlea (an organ in the ear) which vibrates at different spots depending on the frequency of the incoming sounds. Depending on the location in the cochlea that vibrates (which wobbles small hairs), different nerves fire informing the brain that certain frequencies are present. Our periodogram estimate performs a similar job for us, identifying which frequencies are present in the frame.

The periodogram spectral estimate still contains a lot of information not required for Automatic Speech Recognition (ASR). In particular the cochlea cannot discern the difference between two closely spaced frequencies. This effect becomes more pronounced as the frequencies increase. For this reason we take clumps of periodogram bins and sum them up to get an idea of how much energy exists in various frequency regions. This is performed by our Mel filter bank: the first filter is very narrow and gives an indication of how much energy exists near 0 Hertz. As the frequencies get higher our filters get wider as we become less concerned about variations. We are only interested in roughly how much energy occurs at each spot. The Mel scale tells us exactly how to space our filter banks and how wide to make them.

Once we have the filter bank energies, we take the logarithm of them. This is also motivated by human hearing: we don't hear loudness on a linear scale. Generally to double the perceived volume of a sound we need to put 8 times as much energy into it. This means that large variations in energy may not sound all that different if the sound is loud to begin with. This compression operation makes our features match more closely what humans actually hear. Why the logarithm and not a cube root? The logarithm allows us to use cepstral mean subtraction, which is a channel normalization technique.

The final step is to compute the DCT of the log filter bank energies. There are 2 main reasons this is performed. Because our filter banks are all overlapping, the filter bank energies are quite correlated with each other. The DCT decorrelates the energies which means diagonal

covariance matrices can be used to model the features in e.g. a HMM classifier. But notice that only 12 of the 26 DCT coefficients are kept. This is because the higher DCT coefficients represent fast changes in the filter bank energies and it turns out that these fast changes actually degrade ASR performance, so we get a small improvement by dropping them.

B. MFCC Calculations:

The Mel Scale:

The Mel scale relates perceived frequency, or pitch, of a pure tone to its actual measured frequency. Humans are much better at discerning small changes in pitch at low frequencies than they are at high frequencies. Incorporating this scale makes our features match more closely what humans hear[5].

The formula for converting from frequency to Mel scale is:

$$M(f) = 1125\ln(1 + \frac{f}{700}) \quad (1)$$

To go from Mels back to frequency:

$$M^{-1}(m) = 700(\exp\left(\frac{m}{1125}\right) - 1) \quad (2)$$

C. Implementation steps:

We start with a speech signal, we'll assume sampled at 16kHz.

1. Framing:

Frame the signal into 20-40 ms frames. 25ms is standard. This means the frame length for a 16kHz signal is 0.025*16000 = 400 samples. Frame step is usually something like 10ms (160 samples), which allows some overlap to the frames. The first 400 sample frame starts at sample 0, the next 400 sample frame starts at sample 160 etc. until the end of the speech file is reached. If the speech file does not divide into an even number of frames, pad it with zeros so that it does.

The next steps are applied to every single frame, one set of 12 MFCC coefficients is extracted for each frame. A short aside on notation: we call our time domain signal S(n). Once it is framed we have $S_i(n)$ where n ranges over 1-400 (if our frames are 400 samples) and ranges over the number of frames. When we calculate the complex DFT, we get $S_i(k)$ - where the $_i$ denotes

the frame number corresponding to the time-domain frame. $P_i(k)$ is then the power spectrum of frame.

2. Discrete Fourier Transform:

To take the Discrete Fourier Transform of the frame, perform the following:

$$S_i(k) = \sum_{n=1}^{N} S_i(n)h(n)e^{-\frac{j2\pi kn}{N}} \qquad (3)$$

where h(n) is an N sample long analysis window (e.g. hamming window), and K is the length of the DFT. The periodogram-based power spectral estimate for the speech frame $s_i(n)$ is given by:

$$P_i(k) = \frac{1}{N}|S_i(k)|^2 \qquad (4)$$

This is called the Periodogram estimate of the power spectrum. We take the absolute value of the complex Fourier transform, and square the result. We would generally perform a 512 point FFT and keep only the first 257 coefficients.

3. Computation of Mel-spaced filter bank:

Compute the Mel-spaced filter bank. This is a set of 20-40(26 is standard) triangular filters that we apply to the periodgram power spectral estimate from step 2.Our filter bank comes in the form of 26 vectors of length 257(assuming the FFT settings from step 2). Each vector is mostly zeros, but is non-zero for a certain section of the spectrum. To calculate filter bank energies we multiply each filter bank with the power spectrum and then add up the coefficients. Once this is performed we are left with 26 numbers that give us an indication of how much energy was in each filter bank.we can see in below figure 3.1.

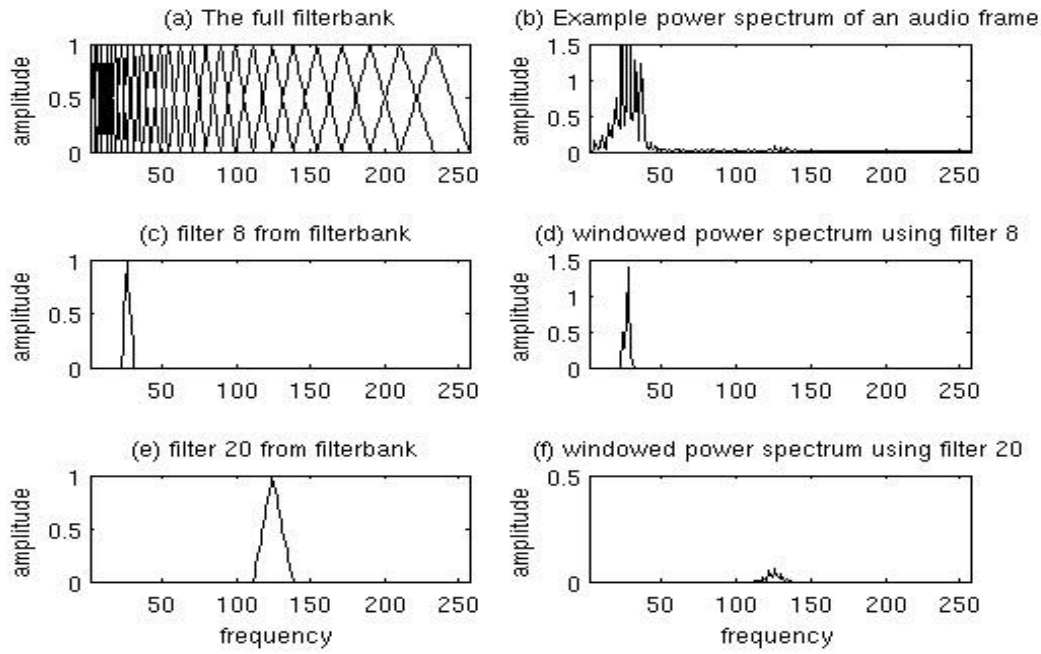


Fig 3.1: Plot of Mel Filter bank and windowed power spectrum.

4. Log:

Take the log of each of the 26 energies from step 3. This leaves us with 26 log filter bank energies.

5. Discrete Cosine Transform:

Take the Discrete Cosine Transform (DCT) of the 26 log filter bank energies to give 26 cepstral coefficients. For ASR, only the lower 12-13 of the 26 coefficients are kept. The resulting features (12 numbers for each frame) are called Mel Frequency Cepstral Coefficients.

Computing the Mel filter banks:

In this example will use 10 filter banks because it is easier to display, in reality 26-40 filter banks can be used.

To get the filter banks shown in figure, we first have to choose a lower and upper frequency. Good values are 300Hz for the lower and 8000Hz for the upper frequency. Of course if the speech is sampled at 8000Hz our upper frequency is limited to 4000Hz. Then follow these steps:

1. Using equation 1, convert the upper and lower frequencies to Mels. In our case 300Hz is 401.25 Mels and 8000Hz is 2834.99 Mels.
2. For this example we will have 10 filter banks, for which we need 12 points. This means we need 10 additional points spaced linearly between 401.25 and 2834.99. This comes out to be: m(i) = 401.25, 622.50, 843.75, 1065.00, 1286.25, 1507.50, 1728.74, 1949.99, 2171.24, 2392.49, 2613.74, 2834.99

3. Now use equation 2 to convert these back to Hertz:
   h(i) = 300, 517.33, 781.90, 1103.97, 1496.04, 1973.32, 2554.33, 3261.62, 4122.63, 5170.76, 6446.70, 8000
   Notice that our start- and end-points are at the frequencies we wanted.
4. We don't have the frequency resolution required to put filters at the exact points calculated above, so we need to round those frequencies to the nearest FFT bin. This process does not affect the accuracy of the features. To convert the frequencies to fft bin numbers we need to know the FFT size and the sample rate, f(i) = floor((nfft+1)*h(i)/sample rate) This results in the following sequence:

   f(i) = 9, 16, 25, 35, 47, 63, 81, 104, 132, 165, 206, 256

   We can see that the final filter bank finishes at bin 256, which corresponds to 8 kHz with a 512 point FFT size. Now we create our filter banks. The first filter bank will start at the first point; reach its peak at the second point, then return to zero at the 3rd point. The second filter bank will start at the 2nd point, reach its max at the 3rd, then be zero at the 4th etc. A formula for calculating these is as follows:

   Where is the number of filters we want, and is the list of M+2 Mel-spaced frequencies.

$$H_m(k) = \begin{cases} 0 & k < (m-1) \\ \frac{k - f(m-1)}{f(m) - f(m-1)} & f(m-1) < k < f(m+1) \\ \frac{f(m+1) - k}{f(m+1) - f(m)} & f(m) < k < f(m+1) \\ & k > f(m+1) \end{cases} \quad (5)$$

5. The final plot of all 10 filters overlapped on each other is:

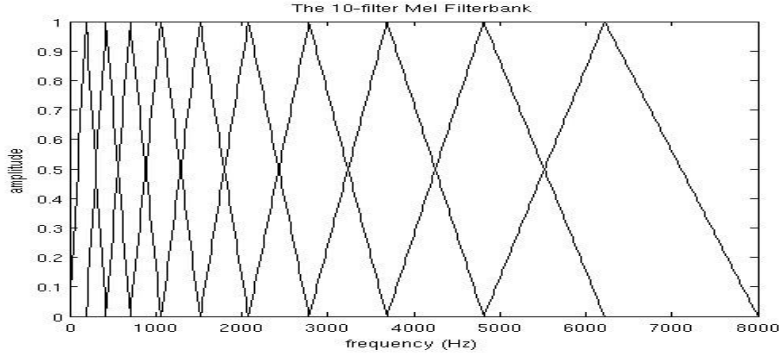


Fig 3.2: A Mel filter bank containing 10 filters.

Deltas and Delta-Deltas

Also known as differential and acceleration coefficients. The MFCC feature vector describes only the power spectral envelope of a single frame, but it seems like speech would also have information in the dynamics i.e. what are the trajectories of the MFCC coefficients over time. It turns out that calculating the MFCC trajectories and appending them to the original feature vector increases ASR performance by quite a bit (if we have 12 MFCC coefficients, we would also get 12 delta coefficients, which would combine to give a feature vector of length 24).

To calculate the delta coefficients, the following formula is used:

$$d_t = \frac{\sum_{n=1}^{N} n(c_{t+n} - c_{t-n})}{2\sum_{n=1}^{N} n^2}$$

where d(t) is a delta coefficient, from frame t computed in terms of the static coefficients c_{t+N} to c_{t-N}. A typical value for N is 2. Delta-Delta (Acceleration) coefficients are calculated in the same way, but they are calculated from the deltas, not the static coefficients.

D. Applications:

MFCCs are commonly used as features in speech recognition systems, such as the systems which can automatically recognize numbers spoken into a telephone[10]. They are also common in speaker recognition, which is the task of recognizing people from their voices. MFCCs are also increasingly finding uses in music information retrieval applications such as genre classification, audio similarity measures, etc.

Various methods of Feature Extraction:

For knowing various properties and procedure of implanting them we can see the below table 3.1 which provide various information on this topic.

Table 3.1: Feature extraction methods.

Sr.No.	Method	Property	Procedure of implement ation
1	Principal Component Analysis(PCA)	Nonlinear feature extraction method, Linear map, fast, Eigen vector based.	Traditional, Eigen vector based method, also known as karhumen-Loeve expansion. Good for Gaussian data.
2	Linear Discriminate Analysis(LDA)	Nonlinear feature extraction method, Supervised linear map, fast, Eigen vector based.	Better than PCA for classification[9]
3	Independent Component Analysis(ICA)	Nonlinear feature extraction method, Linear map, iterative nonGaussian.	Blind course separation, used for de-mixing nongaussian distributed sources (features).
4	Linear Predictive Coding.	Static feature extraction method, 10 to 16 lower order coefficient.	It is used for feature extraction at lower order.

5	Cepstral Analysis	Static feature extraction method, Power spectrum.	Used to represent spectral envelope[9].
6	Mel-frequency Scale Analysis	Static feature extraction method, spectral analysis.	Spectral analysis is done with fixed resolution along a subjective frequency scale i.e. Mel-frequency scale.
7	Filter Bank Analysis	Filters tuned frequencies.	
8	Mel-Frequency cepstral (MFFC)	Power spectrim is computed by performing Fourier Analysis.	This method is used to find out features.
9	Kernel Based Feature Extraction Method.	Nonlinear transformations	Dimensionality reduction leads to barrier classification & it is used to redundant features & improvement in classification error.
10	Wavelet	Better time resolution at high frequencies then fourier Transform.	It replaces the fix bandwidth of fourier transform with one proportional to

			frequency which allow better time resolution at high frequencies then fourier transform.
11	Dynamic Features Exraction • LPC • MFCC	Acceleration and delta coefficients i.e. 2 & 3 order derivative of normal LPC and MFCC coefficient.	It is used by Dynamic or Runtime features.
12	Spectral Subtraction	Robust feature extraction.	It is used on basis of spectrogram.
13	Cepstral Mean Subtraction	Robust feature extraction.	It is same as MFCC but working on mean ststically parameter.
14	RASTA filtering	For Noisy speech.	It is found out in noisy data extraction.
15	Integrated Phoneme subspace method.(Compound method	A transformation based on PCA-LDA-ICA.	Higher accuracy then the existing method.

3.1.2 CLASSIFICATION USING ARTIFICIAL NEURAL NETWORKS:

What are Artificial Neural Networks?

Artificial Neural Networks are relatively crude electronic models based on the neural structure of the brain. The brain basically learns from experience. It is natural proof that some

problems that are beyond the scope of current computers are indeed solvable by small energy efficient packages. This brain modeling also promises a less technical way to develop machine solutions. This new approach to computing also provides a more graceful degradation during system overload than its more traditional counterparts.

Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques[6]. A trained neural network can be thought of as an "expert" in the category of information it has been given to analyze. This expert can then be used to provide projections given new situations of interest and answer "what if" questions.

Other advantages include:

Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience.

Self-Organization: An ANN can create its own organization or representation of the information it receives during learning time.

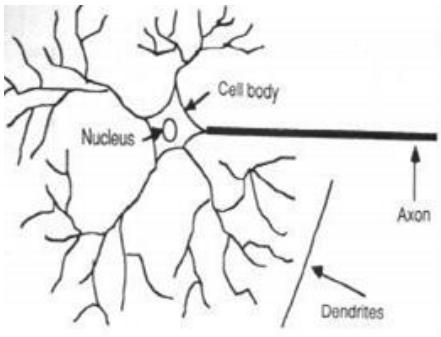
Real Time Operation: ANN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which take advantage of this capability.

Fault Tolerance via Redundant Information Coding: Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage.

Human and Artificial Neurons - investigating the similarities How the Human Brain Learns?

Much is still unknown about how the brain trains itself to process information, so theories abound. In the human brain, a typical neuron collects signals from others through a host of fine structures called *dendrites*. The neuron sends out spikes of electrical activity through a long, thin stand known as an *axon*, which splits into thousands of branches. At the end of each branch, a structure called a *synapse* converts the activity from the axon into electrical effects that inhibit or excite activity from the axon into electrical effects that inhibit or excite activity in the connected neurons. When a neuron receives excitatory input that is sufficiently large compared with its inhibitory input, it sends a spike of electrical activity down its axon. Learning occurs by changing the effectiveness of the synapses so that the influence of one neuron on another changes.

Below figures 3.3 & 3.4 provide the information about component of neuron and axon respectively.



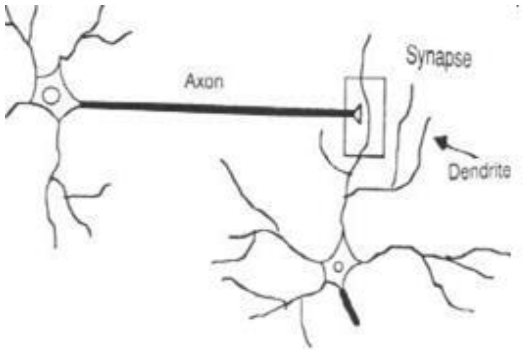


Fig 3.3: Components of a neuron *Fig 3.4: The synapse*

From Human Neurons to Artificial Neurons

We conduct these neural networks by first trying to deduce the essential features of neurones and their interconnections. We then typically program a computer to simulate these features. However because our knowledge of neurones is incomplete and our computing power is limited, our models are necessarily gross idealisations of real networks of neurones.

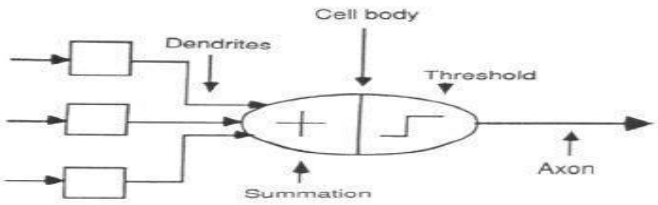


Fig 3.5: The neuron model

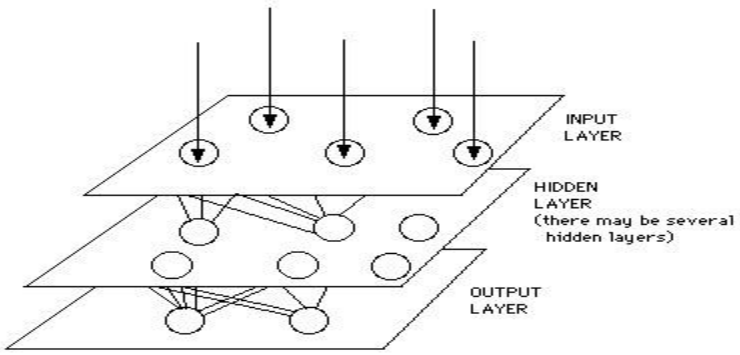


Fig 3.6: A Simple Neural Network Diagram.

Basically, all artificial neural networks have a similar structure or topology as shown in above Figure 3.5 & 3.6. In that structure some of the neurons interface to the real world to receive its inputs. Other neurons provide the real world with the network's outputs. This output might be the particular character that the network thinks that it has scanned or the particular image it thinks is being viewed. All the rest of the neurons are hidden from view of fig 3.6.

Architecture of neural networks

1. Feed-forward networks

In figure 3.7 Feed-forward ANNs allow signals to travel one way only; from input to output. There is no feedback (loops) i.e. the output of any layer does not affect that same layer. Feed-forward ANNs tend to be straight forward networks that associate inputs with outputs. They are extensively used in pattern recognition. This type of organization is also referred to as bottom-up or top down.

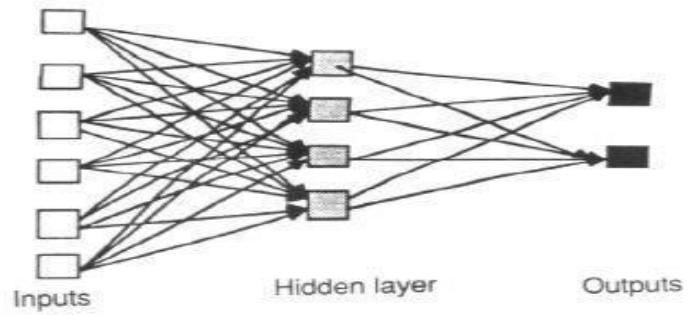


Fig 3.7: An example of a simple feed forward network

Robotics Applications:

A fairly simple home-built robot probably doesn't have much need for a Neural Network. However, with larger-scale projects, there are many difficult problems to be solved. A robot that walks on two legs will have some sort of gyro or accelerometer system that is equivalent to the human inner-ear. This data must be processed along with the position of each part of the body, and with variations in the terrain. A robot that responds to a variety of voice commands must analyze the time, amplitude, and frequency components of what it hears; and compare it to a known vocabulary. A game-playing robot must respond to the unpredictable behavior of its opponent. Also, it may want to "learn from experience" how to play a better game.

2. Feedback networks

Feedback networks can have signals travelling in both directions by introducing loops in the network. Feedback networks are very powerful and can get extremely complicated. Feedback networks are dynamic; their 'state' is changing continuously until they reach an equilibrium point. They remain at the equilibrium point until the input changes and a new equilibrium needs to be found. Feedback architectures are also referred to as interactive or recurrent, although the latter term is often used to denote feedback connections in single-layer organizations. This can be illustrated in figure 3.8.

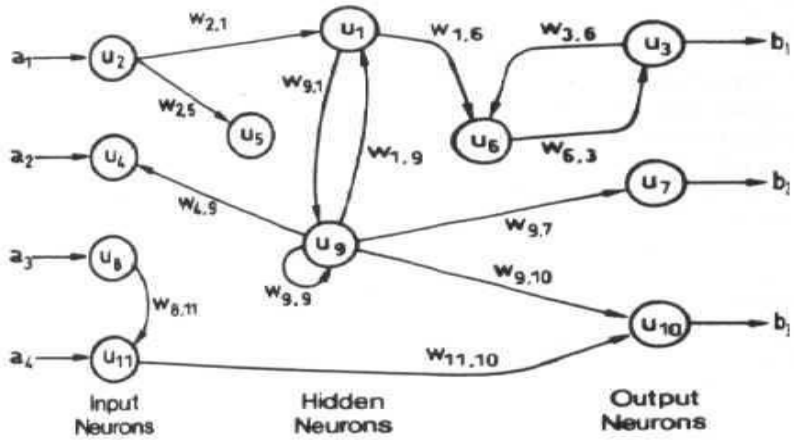


Fig 3.8: An example of a complicated network

The Learning Process

We can distinguish two major categories of neural networks:

1. **Fixed networks:** In which the weights cannot be changed, ie dW/dt=0. In such networks, the weights are fixed a priori according to the problem to solve.

2. **Adaptive networks:** which are able to change their weights, ie dW/dt ≠ 0.

All learning methods used for adaptive neural networks can be classified into two major categories:

Supervised learning: which incorporates an external teacher, so that each output unit is told what its desired response to input signals ought to be. During the learning process global information may be required. Paradigms of supervised learning include error-correction learning, reinforcement learning and stochastic learning.

An important issue concerning supervised learning is the problem of error convergence, i.e. the minimization of error between the desired and computed unit values. The aim is to determine a set of weights which minimizes the error. One well-known method, which is common to many learning paradigms, is the least mean square (LMS) convergence.

Unsupervised learning: It uses no external teacher and is based upon only local information. It is also referred to as self-organization, in the sense that it self-organizes data presented to the network and detects their emergent collective properties. Paradigms of unsupervised learning are Hebbian lerning and competitive learning.

From Human Neurons to Artificial Neuronesther aspect of learning concerns the distinction or not of a separate phase, during which the network is trained, and a subsequent operation phase. We say that a neural network learns off-line if the learning phase and the operation phase are distinct. A neural network learns on-line if it learns and operates at the same time. Usually, supervised learning is performed off-line, whereas unsupervised learning is performed on-line.

NEURAL NETWORK ALGORITHM.

(A) Back propagation Algorithm:

In the employment of the back propagation algorithm, each iteration of training involves the following steps:

I. A particular case of training data is fed through the network in a forward direction, producing results at the output layer,

II. Error is calculated at the output nodes based on known target information, and the necessary changes to the weights that lead into the output layer are determined based upon this error calculation,

III. The changes to the weights that lead to the preceding network layers are determined as a function of the properties of the neurons to which they directly connect (weight changes are calculated, layer by layer, as a function of the errors determined for all subsequent layers, working backward toward the input layer) until all necessary weight changes are calculated for the entire network.

Then the speech will be matched and identified.

(B) K means algorithm

***K-means* clustering** is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining. *K-means* clustering aims to partition *n* observations into *k* clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi cells.

The problem is computationally difficult (NP-hard); however, there are efficient heuristic algorithms that are commonly employed and converge quickly to a local optimum. These are usually similar to the expectation-maximization algorithm for mixtures of Gaussian distributions via an iterative refinement approach employed by both algorithms. Additionally, they both use cluster centers to model the data; however, *k*-means clustering tends to find clusters of comparable spatial extent, while the expectation-maximization mechanism allows clusters to have different shapes.

The algorithm has nothing to do with and should not be confused with *k*-nearest neighbor, another popular machine learning technique.

3.1.3 FEATURE MATCHING:

In the recognition phase an unknown command/speech, represented by a sequence of feature vectors, is compared with the codebooks in the database. For each codebook a distortion measure is computed, and the command with the lowest distortion is chosen.

3.2 COST ANALYSIS:

In this table 3.2 shows cost of various module and component used in implanting the project.

Table 3.2:Cost analysis

Sr. No.	Particulars	Cost(Rs.)
1	Zigbee S2 modules (x2)	2600
2	Xbee USB Explorer	800
3	Easy Cap	800
4	Wireless Camera	2600
5	Battery Li Po	450
6	DC Motors(12V)	330
7	Robot mechanism	310
8	Motor Controller ICs	240
9	Arduino Uno	1300
10	Castor Wheels	110
11	GPB & other components	450
12	Wires and drills	100
	TOTAL	10,090

3.3 PROCESS MODEL:

The figure shows the block diagram of the VOICE CONTROLLED CAMERA ENABLED ROBOT. As shown in the figure 3.9, the first step in the whole process is to provide the system with the voice signals and in our case we have stored the signals like LEFT, RIGHT, STOP, FORWARD and REVERSE. After recording the voice signals, signal processing is done using

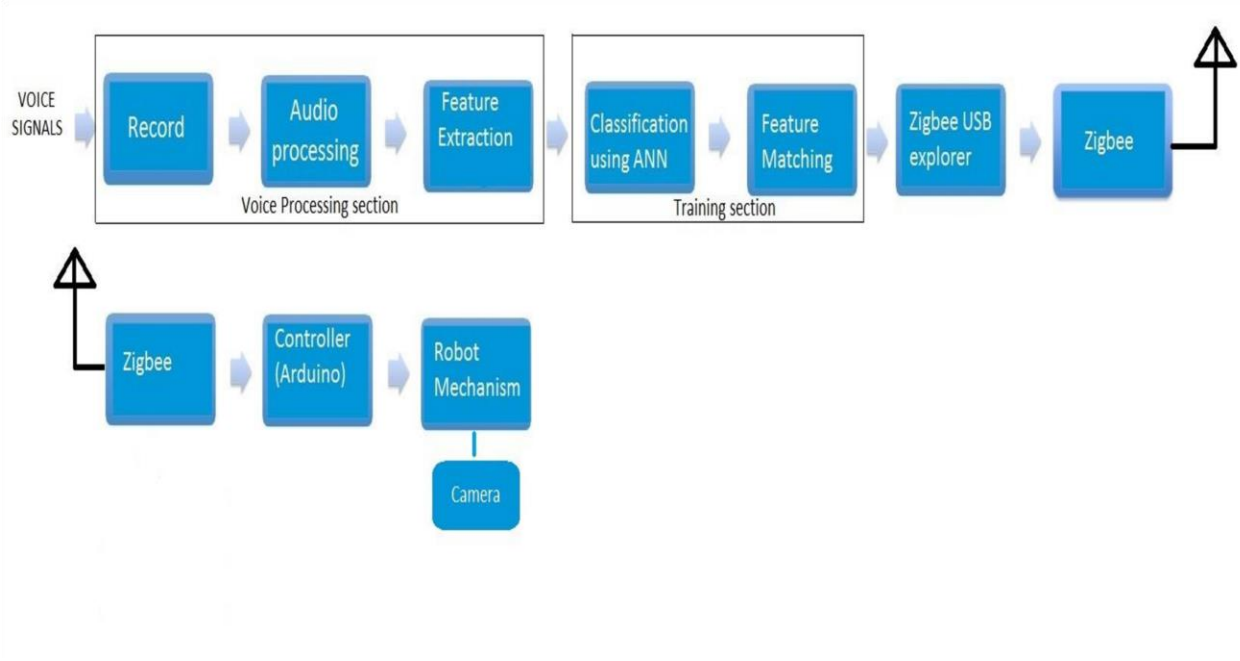


Fig3.9: Process model

MATLAB which includes Feature Extraction. Feature Extraction is be done by using MFCC and Classification using Artificial Neural Network. Artificial Neural Network includes many algorithms but being specific enough we have used Feed Forward Back Propagation method. After extracting the features of the input commands, the corresponding features are stored in a variable and the variable is transmitted serially from the computer to the Zigbee Module. At the Receivers end, Zigbee Module receives the variable character and sends the same to the Arduino. The Arduino controller is loaded with a program containing difference cases to drive the motor according to the character (For ex. F=forward, B=reverse, L=left, R=right, S=stop). It will give instructions to the motor controller IC L298 to take the desired action. The camera on the robot will accordingly show its position. The output of the camera, which is a real-time video is shown in MATLAB.

3.4 DATA FLOW DIAGRAMS:

Training:

Fig 3.10 shows the training procedure of bot.

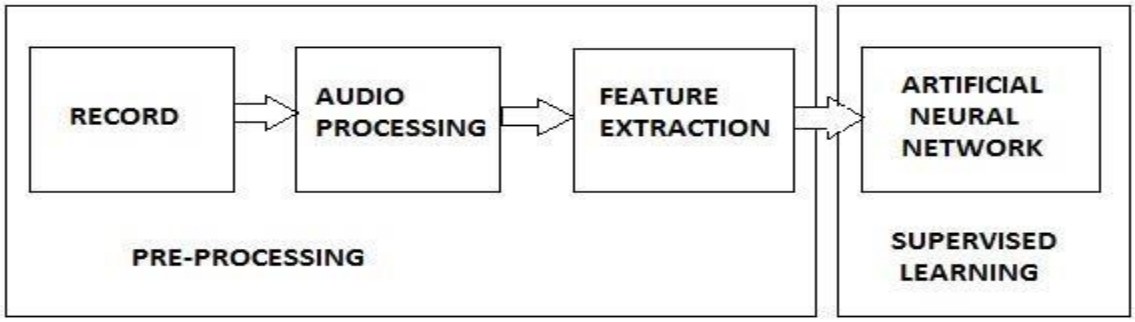


Fig 3.10: Block diagram of training

Testing:

Below figure 3.11 shows the testing procedure.

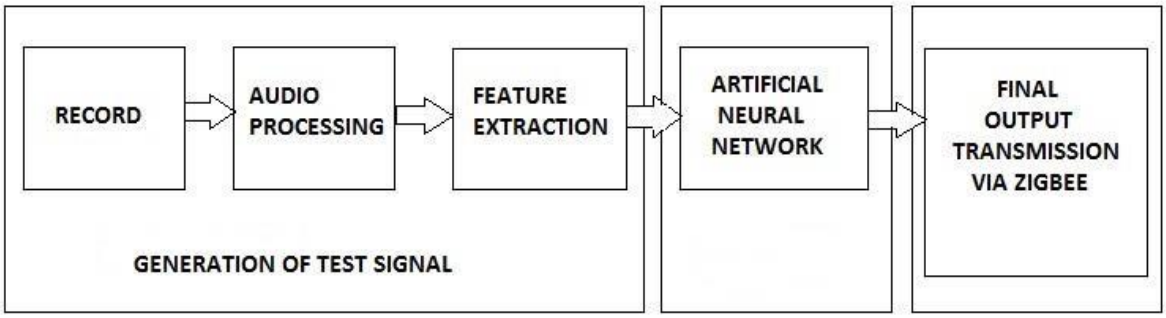


Fig 3.11: Block diagram of testing

3.5 TECHNOLOGIES USED:

3.5.1. Hardware requirements:

3.5.1.1 Arduino IC

Fig 3.12: Arduino IC

In figure 3.12 The Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

It has the following features:

1) 1.0 pin out: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible with both the board that uses the AVR, which operates with 5V.

2) Stronger RESET circuit.
3) Atmega 328P.

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the

latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

Table 3.3 shows the specification of Arduino IC.

Table 3.3: Arduino specifications

Microcontroller	ATmega328
Operating Voltage	5V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by boot loader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16Mhz

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC to DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector. The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

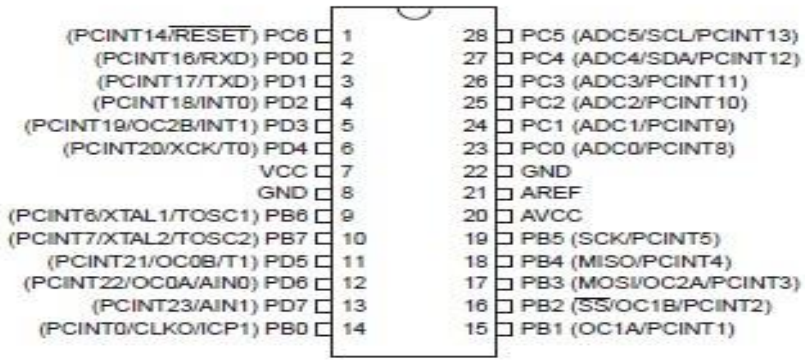


Fig 3.13: Pin diagram of Arduino IC.

3.5.1.2 Zigbee S2 Module:

The XBee Series 2 OEM RF Modules were engineered to operate within the ZigBee protocol and support the unique needs of low-cost, low-power wireless sensor networks. The modules require minimal power and provide reliable delivery of data between remote devices. The modules operate within the ISM 2.4 GHz frequency band. We can see in below fig 3.14 the pin diagram of zigbee and fig 3.15 shows zigbee module.

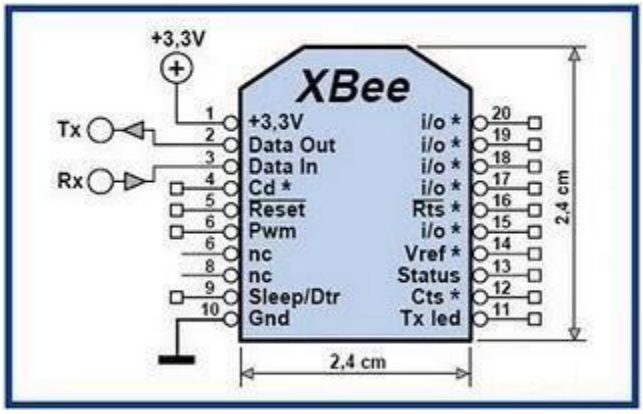


Fig 3.14: Pin Diagram ZIGBEE.

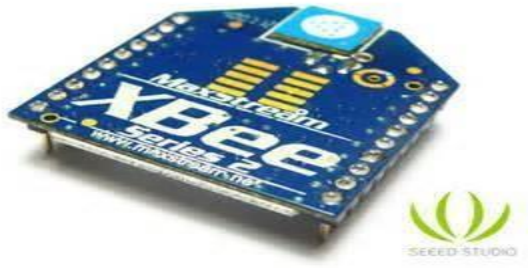


Fig 3.15: Zigbee Module

1. Key Features:

High Performance, Low Cost:

• Indoor/Urban: up to 133’ (40 m)

• Outdoor line-of-sight: up to 400’ (120 m)

• Transmit Power: 2 mW (+3 dBm)

• Receiver Sensitivity: -95 dBm

• RF Data Rate: 250,000 bps

XBee Series 2

• TX Current: 40 mA (@3.3 V)

• RX Current: 40 mA (@3.3 V)

• Power-down Current: < 1 µA @ 25 degree Celsius.

Advanced Networking & Security:

•Retries and Acknowledgements

•DSSS (Direct Sequence Spread Spectrum)

•Each direct sequence channel has over 65,000 unique network addresses available

•Point-to-point, point-to-multipoint and peer-to-peer topologies supported

•Self-routing, self-healing and fault-tolerant mesh networking.

2. Specifications:

Table 3.4 shows the specification of zigbee module.

Table3.4: Specifications of Zigbee

SPECIFICATION	XBEE SERIES2
PERFORMANCE	
Indoor/Urban Range	Up to 133ft(40m)
Outdoor RF line-of-sight Range	Up to 400ft(120m)

Transmit output power (software selectable)	2mW(+3db)
RF Data Rate	250,000bps
Serial Interface Data Rate (software selectable)	1200-230400 bps (nonstandard baud rate also supported)
Receiver sensitivity	-95 dBm(1% packet error rate)
POWER REQUIREMENTS	
Supply Voltage	2.8 – 3.4v
Operating Current(Transmit)	40mA(@3.3v)
Operating Current(Receive)	40mA(@3.3v)
Power-down Current	< 1uA @ 25 C
GENERAL	
Operating Frequency Band	ISM 2.4GHz
Dimensions	0.960'*1.087
Operating Temperature	-40 to 85 C
Antenna Options	Integrated with chip, RPSMA, UFL connector
NETWORKING & SECURITY	
Supported Network Topologies	Point-to-point, point-to-multipoint, peer-to-peer & Mesh
Number of Channel (software selectable)	16 direct sequence channel
Addressing Options	PAN ID addresses, Cluster IDs & Endpoint's (optional)
AGENCY APPROVELS	

United States	Pending
Industry Canada	Pending
Europe	Pending

3. RF Module Operation:

3.1 Serial Communication.

The XBee Series 2 OEM RF Modules interface to a host device through a logic-level asynchronous serial port. Through its serial port, the module can communicate with any logic and voltage compatible UART; or through a level translator to any serial device (For example: Through a Max Stream proprietary RS-232 or USB interface board).

3.2 UART Data Flow:

Devices that have a UART interface can connect directly to the pins of the RF module as shown in the figure below. Fig 3.16 shows UART data flow.

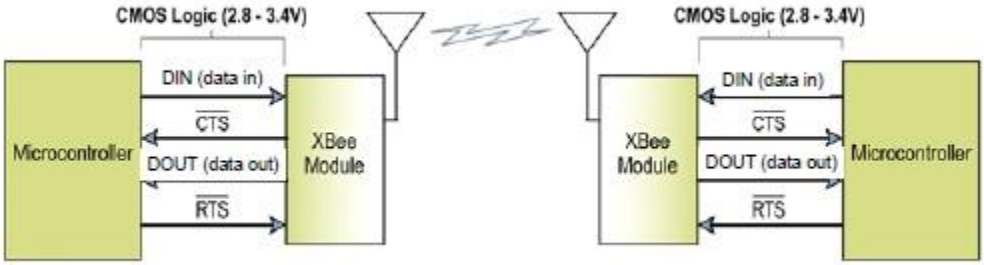


Fig 3.16: UART Data Flow

Serial Data

Data enters the module UART through the DIN (pin 3) as an asynchronous serial signal. The signal should idle high when no data is being transmitted. Each data byte consists of a start bit (low), 8 data bits (least significant bit first) and a stop bit (high). The following figure illustrates the serial bit pattern of data passing through the module.

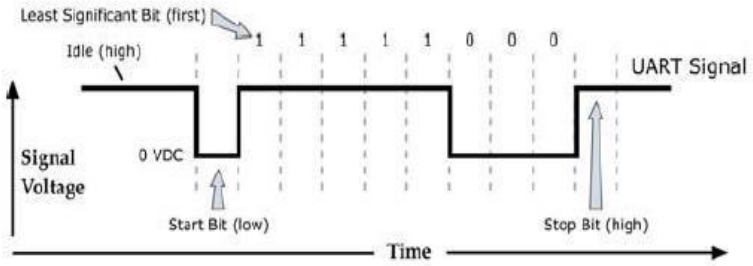


Fig 3.17: Serial Data flow through RF Module

The module UART performs tasks, such as timing and parity checking, that are needed for data Communications. Serial communications depend on the two UARTs to be configured with compatible settings (baud rate, parity, start bits, stop bits, data bits.

Serial Buffer:

The XBee Series 2 modules maintain small buffers to collect received serial and RF data. The serial Receive buffer collects incoming serial characters and holds them until they can be processed. The serial transmit buffer collects data that is received via the RF link that will be transmitted out the UART.

Serial Receive Buffer:

When serial data enters the RF module through the DIN Pin (3 pin), the data is stored in the serial receive buffer until it can be processed.

Hardware Flow Control (CTS):

When the serial receive buffer is 17 bytes away from being full, by default, the module de-asserts CTS(High) to signal to the host device to stop sending data [refer to D7 (DIO7 Configuration) parameter]. CTS are re-asserted after the serial receive buffer has 34 bytes of memory available.

Cases in which the serial receive buffer may become full and possibly overflow:

I. If the module is receiving a continuous stream of RF data, any serial data that arrives on the DIN pin is placed in serial receive buffer. The data in the serial receive buffer will be transmitted over-the-air when the module is no longer receiving RF data in the network.

II. When data is ready to be transmitted, the module may need to discover a Network Address and/or a Route in order to reach the destination node. Discovery overhead may delay packet transmission.

Serial Transmit Buffer:

When RF data is received, the data is moved into the serial transmit buffer and is sent out the Serial port. If the serial transmit buffer becomes full enough such that all data in a received RF Packet won't fit in the serial transmit buffer, the entire RF data packet is dropped.

Hardware Flow Control (RTS).

If RTS is enabled for flow control (D6 (DIO6 Configuration) Parameter = 1), data will not be sent out the serial transmit buffer as long as RTS (pin 16) is de-asserted.

Cases in which the serial transmit buffer may become full resulting in dropped RF packets:

I. If the module is receiving a continuous stream of RF data, any serial data that arrives on the DIN pin is placed in the serial receive buffer. The data in the serial receive buffer will be transmitted over-the-air when the module is no longer receiving RF data in the network.

II. When data is ready to be transmitted, the module may need to discover a Network Address and/or a Route in order to reach the destination node. Discovery overhead may delay packet transmission.

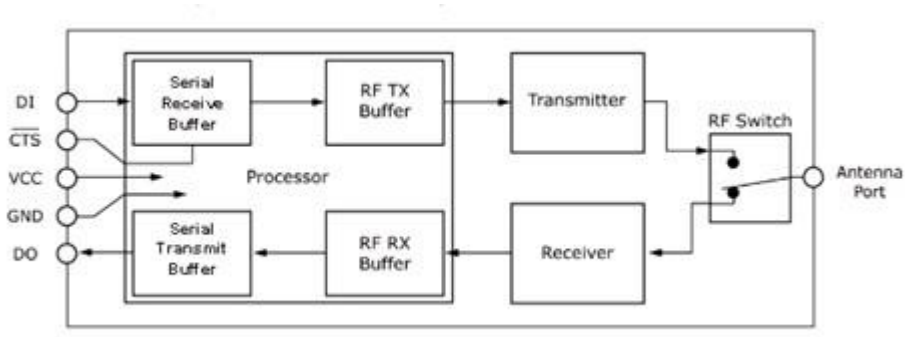


Fig 3.18: Internal Flow Diagram.

Modes of Operation of Zigbee:

1. Idle Mode.

When not receiving or transmitting data, the RF module is in Idle Mode. During Idle Mode, the RF module is also checking for valid RF data. The module shifts into the other modes of operation under the following conditions:

• Transmit Mode (Serial data in the serial receive buffer is ready to be packetized).

• Receive Mode (Valid RF data is received through the antenna).

• Sleep Mode (End Devices only).

• Command Mode (Command Mode Sequence is issued).

2. Transmit Mode.

When serial data is received and is ready for packetization, the RF module will exit Idle Mode and attempt to transmit the data. The destination address determines which node(s) will receive the data. Prior to transmitting the data, the module ensures that a 16-bit Network Address and route to the destination node have been established. If the 16-bit Network Address is not known, Network Address Discovery will take place. If a route is not known, route discovery will take place for the purpose of establishing a route to the destination node. If a module with a matching Network Address is not discovered, the packet is discarded. The data will be transmitted once a route is established. If route discovery fails to establish a route, the packet will be discarded.
When data is transmitted from one node to another, a network-level acknowledgement is transmitted back across the established route to the source node. This acknowledgement packet indicates to the source node that the data packet was received by the destination node. If a network acknowledgement is not received, the source node will re-transmit the data.

3. Receive Mode.

If a valid RF packet is received and its address matches the RF module's MY (16-bit Source Address) parameter, the data is transferred to the serial transmit buffer.

4. Command Mode.

To modify or read RF Module parameters, the module must first enter into Command Mode - a state in which incoming serial characters are interpreted as commands.

5. Sleep Mode.

Sleep modes are supported on end devices only. Router and coordinator devices participate in routing data packets and are intended to be mains powered. End devices must be joined to a parent (router or coordinator) before they can participate on a ZigBee network. The parent device does not track when an end device is awake or asleep. Instead, the end device must inform the parent when it is able to receive data. The parent must be able to buffer incoming data packets destined

for the end device until the end device can awake and receive the data. When an end device is able to receive data, it sends a poll command to the parent. When the parent router or coordinator receives the poll command, it will transmit any buffered data packets for the end device. Routers and coordinators are capable of buffering one broadcast transmission for sleeping end device children. The SM, ST, SP, and SN commands are used to configure sleep mode operation.

3.5.1.3 Motor Driver IC L298:

The L298 is an integrated monolithic circuit in a 15- lead Multi watt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the connection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

Pin diagram:

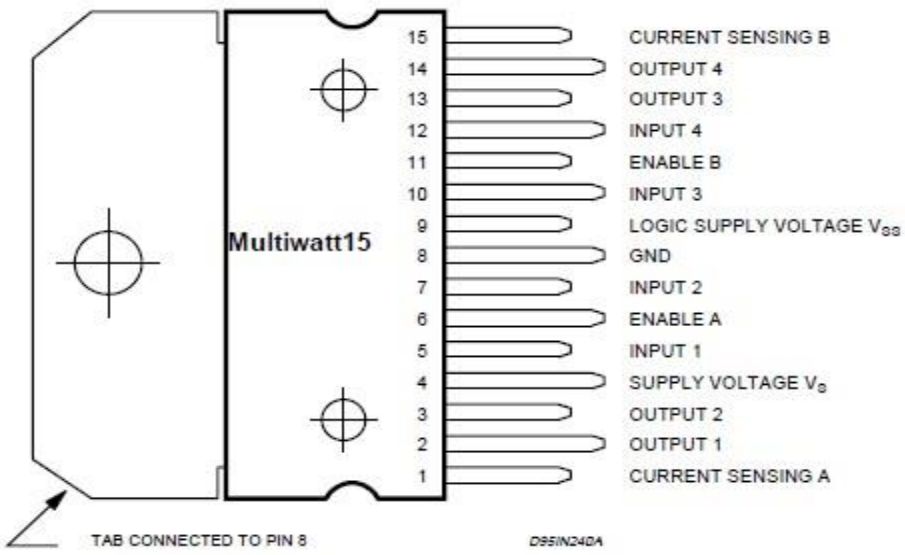


Fig 3.19: Pin Diagram L298

Specifications:

- Operating supply voltage up to 46V.
- Total DC current up to 4A.
- Low saturation voltage.
- Over temperature protection.

3.5.2 Software Requirements:

3.5.2.1 MATLAB Software

MATLAB® is the high-level language and interactive environment used by millions of engineers and scientists worldwide. It lets you explore and visualize ideas and collaborate across disciplines including signal and image processing, communications, control systems, and computational finance.

- High-level language for numerical computation, visualization, and application development
- Interactive environment for iterative exploration, design, and problem solving
- Mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, numerical integration, and solving ordinary differential equations
- Built-in graphics for visualizing data and tools for creating custom plots
- Development tools for improving code quality and maintainability and maximizing performance
- Tools for building applications with custom graphical interfaces
- Functions for integrating MATLAB based algorithms with external applications and languages such as C, Java, .NET, and Microsoft® Excel®

Functions:

1. Numeric Computation

MATLAB provides a range of numerical computation methods for analyzing data, developing algorithms, and creating models. The MATLAB language includes mathematical functions that support common engineering and science operations. Core math functions use processoroptimized libraries to provide fast execution of vector and matrix calculations.

2. Data Analysis and Visualization.

MATLAB provides tools to acquire, analyze, and visualize data, enabling you to gain insight into your data in a fraction of the time it would take using spreadsheets or traditional programming languages. You can also document and share your results through plots and reports or as published MATLAB code.

3. Visualizing Data.

MATLAB provides built-in 2-D and 3-D plotting functions, as well as volume visualization functions. You can use these functions to visualize and understand data and communicate results. Plots can be customized either interactively or programmatically. The MATLAB plot gallery provides examples of many ways to display data graphically in MATLAB. For each example, you can view and download source code to use in your MATLAB application.

4. Documenting and Sharing Results.

You can share results as plots or complete reports. MATLAB plots can be customized to meet publication specifications and saved to common graphical and data file formats. You can automatically generate a report when you execute a MATLAB program. The report contains your code, comments, and program results, including plots. Reports can be published in a variety of formats, such as HTML, PDF, Word, or LaTeX.

5. Application Development and Deployment.

MATLAB tools and add-on products provide a range of options to develop and deploy applications. You can share individual algorithms and applications with other MATLAB users or deploy them royalty-free to others who do not have MATLAB.

Development Tools.

MATLAB includes a variety of tools for efficient algorithm development, including:

- **Command Window** - Lets you interactively enter data, execute commands and programs, and display results
- **MATLAB Editor** - Provides editing and debugging features, such as setting breakpoints and stepping through individual lines of code
- **Code Analyzer** - Automatically checks code for problems and recommends modifications to maximize performance and maintainability
- **MATLAB Profiler** - Measures performance of MATLAB programs and identifies areas of code to modify for improvement.

Matrices.

Matrices can be defined by separating the elements of a row with blank space or comma and using a semicolon to terminate each row. The list of elements should be surrounded by square brackets: []. Parentheses: () are used to access elements and sub arrays (they are also used to denote a function argument list).

3.5.2.2 ZigBee X-CTU:

XCTU is a free multi-platform application designed to enable developers to interact with Digi RF modules through a simple-to-use graphical interface. It includes new tools that make it easy to set-up, configure and test XBee® RF modules.

Other highlights of XCTU include the following features:

- You can **manage and configure multiple RF devices**, even remotely (over-the-air) connected devices.
- The **firmware update** process **seamlessly** restores your module settings, automatically handling mode and baud rate changes.
- Two specific **API** and **AT consoles**, have been designed from scratch to communicate with your radio devices.
- You can now **save your console sessions** and load them in a different PC running XCTU.
- XCTU includes a set of embedded tools that can be executed without having any RF module connected:
    - **Frames generator**: Easily generate any kind of API frame to save its value.
    - **Frames interpreter**: Decode an API frame and see its specific frame values.
    - **Recovery**: Recover radio modules which have damaged firmware or are in programming mode.
    - **Load console session**: Load a console session saved in any PC running XCTU.
    - **Range test**: Perform a range test between 2 radio modules of the same network.
    - **Firmware explorer**: Navigate through XCTU's firmware library.
- An update process allows you to **automatically update the application itself and the radio firmware** library without needing to download any extra files.
- XCTU contains **complete and comprehensive documentation** which can be accessed at any time.
- XCTU is a **free, multi-platform** application compatible with Windows and MacOS
- **Graphical Network View** for simple wireless network configuration and architecture
- **API Frame Builder** is a simple development tool for quickly building XBee API frames
- **Device Cloud** integrated, allowing configuration and management of XBee devices anywhere in the world.

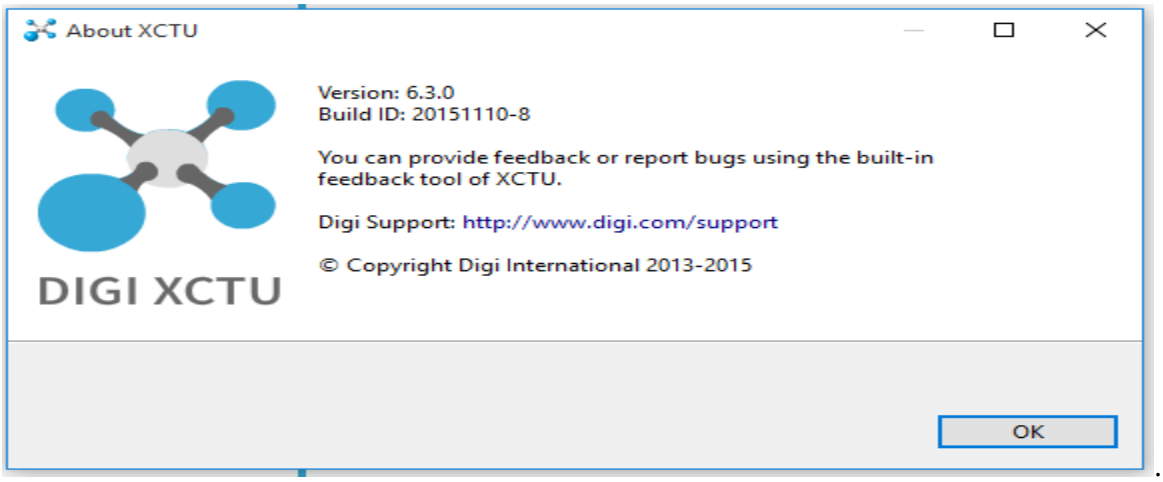


Fig 3.20: About XCTU

Before you can talk to an XBee, you will need a XBEE adapter. With the USB adapter, you can communicate with Xbee through **"USB Serial Port"**. You may have more than one device on serial port, for example, you want to test the wireless communication connect two XBee to your PC. So you will more devices on serial port

This is an easy way to test and check if an XBee is working and configuring properly. After parameter modification and firmware upgrading, you need to do this for checking if everything is ok. You will have a very little chance to get a problem Xbee. If you got problem to Test / Query an XBee , usually it is due to the wrong parameter setting. For a successful two way communication, the most primary principle is the **"Baud Rate"** should match each other. When everything comes to the right place, this window will appear. If you see any other kind of message window, it means something wrong even you see an OK on it. It is not ok without this message and window. With the same setting, if you only got this occasionally, it means the communication is unstable. You may need to use an XBee adapter with a better quality, or try another XBee. The modem type and firmware version means the firmware is programmed in this XBee. It is possible to have other types of firmware, depends on your usage for XBee.

If something goes wrong, then you will get this window. For most cases, it can be solved by just changing the Baud Rate and un/check API Mode. The wrong firmware in XBee will also result in this kind of message. If the setting and firmware is correct, then the hardware may have problem. In this case, check the adapter first then XBee. A handmade, bad quality Xbee adapter

may result in this. The next section will discuss the four main tabs in X-CTU and information about using XBee with Arduino.

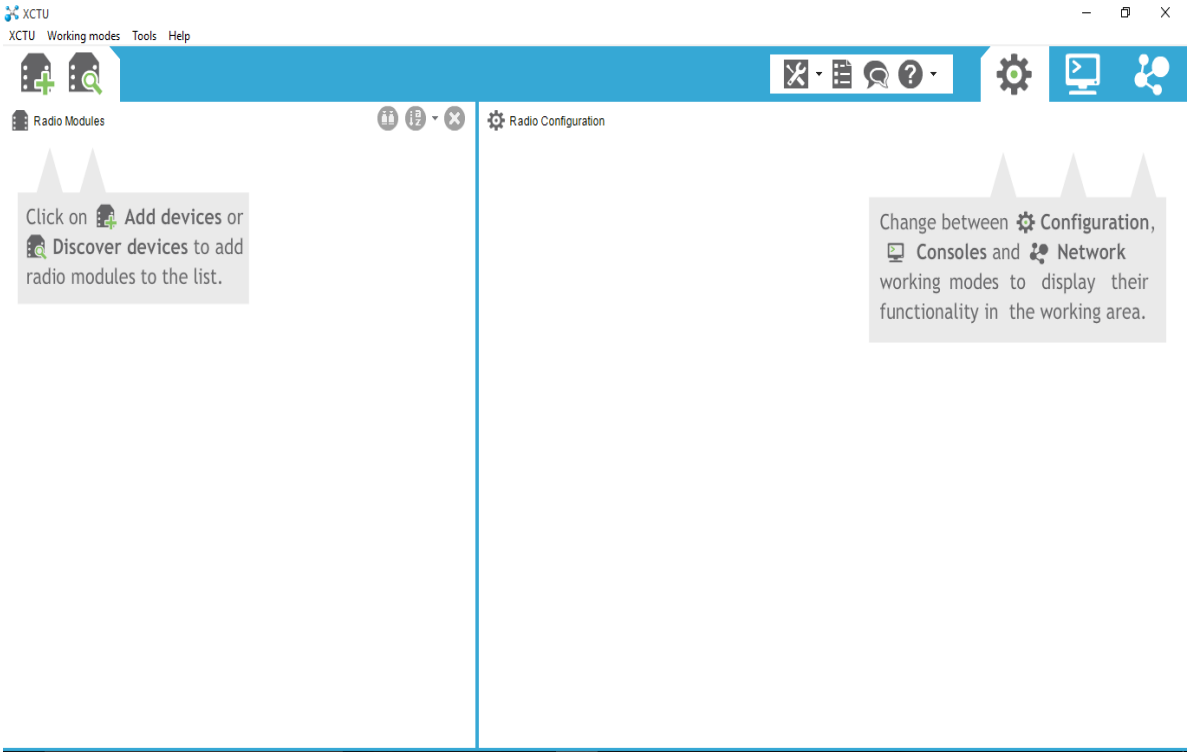


Fig 3.21: User Interface

X-CTU has four main tabs, the brief descriptions are described below:

1. PC Settings:

You can configure the PC to talk to XBee or other devices using serial port interface. You can also test and query if an XBee is working properly in here.

2. Range Test:

After the configuration in PC Settings are correct, AND, the XBee has connected to another one, then you can use this function here. It gives you the ideas about how is the signal strength, how is the successful rate for sending / receiving data.

3. Terminal:

After the configuration in PC Settings are correct, then you can use this simple terminal talk to that device , such as XBee.

4. Modem Configuration:

A very good user interface for reading or writing the parameters of an XBee. You can also update the firmware here. Even you can change the modem type and function set base on how you are going to use it.

X-CTU is intended to upload firmware to XBee radio modules. This is needed to change the firmware between router and co-coordinator of the Zigbee mesh network, and between the different protocol variants that the XBee radios can support. One limitation of X-CTU is that it only works on Windows: if you're running Linux, X-CTU will run under Wine. You can download the latest X-CTU from Digi's X-CTU page; alternatively, there's a version installed on the Citizen Sensing VM.To use X-CTU you need to connect your XBee module to your computer.

CHAPTER 4

PROJECT TIME

&

TASK DISTRIBUTION

CHAPTER 4

PROJECT TIME & TASK DISTRIBUTION

4.1 Time line chart:

In below table 4.1 which provide information on various timeline for doing the work.

Table 4.1: Time line chart

WORK OF PROJECT	DATE
Topic selection	11-AUGUST TO 18-AUGUST 2015
Literature survey on the topic	25-AUGUST & 1-SEPTEMBER 2015
Referring papers & journal's	15 TO 29 SEPTEMBER-2015
Studying the basic vocal tract & its characteristics	6 TO 13 OCTOBER-2015
Study different feature extraction techniques & selecting MFCC techniques	20 TO 27 OCTOBER-2015
Studying ANN & selecting FFB propagation	12 TO 19 JANUARY-2016
Training the robot with different test classes	9 TO 16 FEBRUARY-2016
Testing robot & calculate its efficiency	23-FEBRUARY TO 22-MARCH 2016
Troubleshooting the project	29-MARCH TO 5-APRIL 2016
Deriving conclusion	13-APRIL-2016

CHAPTER 5

TEST CASES

CHAPTER 5

TEST CASES

In below table 5.1 test cases of different sample are taken.

Table 5.1: Table test cases

Input Commands	No. of Samples for testing	Speaker Dependent (Known speaker)		Speaker Independent (Unknown speaker)	
		No. of samples properly recognized	**Recognition rate (%)**	**No. of samples properly recognized**	**Recognition rate (%)**
Forward	15	15	100	2	13
Reverse	15	14	93	0	0
Left	15	6	40	0	0
Right	15	15	100	4	26
Stop	15	14	93	3	20
Total	**75**	**64**	**85.3**	**9**	**11.8**

Thus, the system is capable of giving an efficiency of 85.3% for one known user. The system doesn't respond well to the unknown user, which in turn means that it is secured. To improve the efficiency, more number of samples can be given as input to train the network. Also samples in noisy conditions can be considered so that the system responds well.

CHAPTER 6

CONCLUSION

&

FUTURE SCOPE

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

6.1 Conclusion:

In this project, a speech-control robot system has been developed with Back-propagation Neural Network which has achieved reasonable results for most of the commands. This speech-control system, though quite simple, shows the ability to apply speech recognition techniques to the control application. The system can be used to recognize the commands given to it and has achieved 85.3% recognition accuracy for single speaker. Our robot can understand control commands spoken in a natural way, and execute the corresponding action. The method is proved efficient enough for real-time operation.

With properly trained speakers and noise free environment, the developed system will produce better recognition results. The variability in various parameters, like speed, noise, and loudness will properly handle in our future research.

6.2 Future Scope:

Day by day, the field of robotics is blooming and the robots are having great impact on human beings. The project which is implemented is a non-flying robot mechanism due to certain economic causes but it has a huge scope for future development.

The robot can be transformed into a quad-copter which can fly with a camera on it. Also instead of using voice signals for controlling it, the robot can be made autonomous.

In case of disaster management, the robot can automatically move and scan the disaster-prone area when triggered and can get the video information of the people trapped without any user commands.

Also, multiple User inputs can be applied to the same by getting samples from different users, to get it controlled from different users at a time.

BIBLIOGROPHY

Book,

[1] *"Speaker Diarization of News Broadcasts and Meeting Recordings"* by Koh Chin Wei, *Eugene,Nanyang Technological University.*

[2] *'A Spectral Clustering Approach to Speaker Diarization'* by Huazhong Ning, Ming Liu, Hao Tang, Thomas Huang, from *Beckman Institute, U. of Illinois at Urbana-Champaign.*

[3] *'A Study of New Approaches to Speaker Diarization'* by Douglas Reynolds1 , Patrick

[4] Kenny2, Fabio Castaldo3, 1MIT Lincoln Laboratory, USA 2CRIM, Canada 3Politecnico di Torino, Italy.

[5] *'Segmentation, Diarization And Speech Transcription: Surprise Data Unraveled'* by prof. Dr. W.H.M. Zijm.

[6] *'New Attempts in Sound Diarization'* Ciprian Costin*, Mihaela Costin**, * "Al. I. Cuza" University, Department of Computer Science, Iasi, Romania ** Institute of Computer Science, Romanian Academy, Iasi Branch.

[7] *'Techniques for feature extraction in speech recognition system : a comparative study'* by Urmila Shrawankar,Research Student, SGB Amravati University.

[8] *'Linear Predictive Coding'* by Jeremy Bradbury.

Websites,

[9] *http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency* read on 29-November-2015.

[10] *http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html* viewed on 18-August-2015.

[11] *http://www.webpages.ttu.edu/dleverin/neural_network/neural_networks.html* viewed on 15-December-2015.

[12] *http://www.psych.utoronto.ca/users/reingold/courses/ai/cache/neural2.html* viewed on 30-December-2015.

Journal Paper,

[13] *"NEURAL NETWORKS USED FOR SPEECH RECOGNITION" ,Journal of automatic control, university of Belgrade, vol.20:1-7,2010*

[14] *'MFCC and its applications in speaker recognition' by Vibha Tiwari,Deptt. of Electronics Engg., Gyan Ganga Institute of Technology and Management, Bhopal, (MP) INDIA. International Journal on Emerging Technologies* ***1****(1): 19-22(2010), ISSN : 0975-8364.*

[15] *"DESIGN & IMPLEMENTAION OF A SYSTEM FOR WIRELESS CONTROL OF A ROBOT" , IJCSI International Journal of Computer Science Issues, Vol. 7, Issue 5, September 2010, ISSN (Online): 1694-0814.*

[16] *"Speech recognition from spectral dynamics", Sadhana Vol. 36, Part 5, October 2011, pp. 729–744. Indian Academy of Sciences.*