Chapter 1

Introduction
1.1 Introduction - Why is speech recognition needed?

Speech recognition or recognition of human speech by machines has long been an elusive goal of scientists all over the world. Speech is the most natural way to communicate and many applications are possible if we can speak to machines. For example, we can command a robot to do certain tasks; we can query a database using speech commands from a telephone. We can dictate letters, dial telephone numbers while driving automobiles. We can use speech recognition to translate between languages. We can have a computer recognize spoken Kannada, for example, translate it to Japanese language, and have the computer speak out Japanese. Several applications beyond the above listing are possible. Speech recognition enables ease of use, convenience, productivity, and safety depending on the specific application.

However, human speech has a lot of variation. A person may utter the same word in different ways depending on the context in which that word is used. His vocal expression may alter when he has a cold. In addition to this speech variation, the background noise may vary depending on where the speaker is situated. The ambient noise or background noise in a quiet office environment will be quite different from that of a conference room or an airport. In noisy environments, Lombard effect causes the person to increase his vocal effort, which in turn alters the formants. Similarly, the medium in which the speech signal is transmitted also determines the nature of the signal at the receiving end. For example, a speech signal transmitted over a telephone channel is band limited to 3300Hz only, whereas the signal can have frequencies up to 20000Hz. Some important recognition cues may not be available at the receiving end.
1.2 Challenges and problems of realizing speech recognition systems

Developing and deploying a successful speech recognition based product is very challenging. A speech recognition system that works well under different environmental conditions has remained an elusive goal. There are several reasons for this and span a broad spectrum of diverse disciplines including board design to assure a good signal-to-noise ratio, signal processing algorithms, search algorithms, linguistic processing, efficient real time processing with minimal resources, and man-machine interface. It is typical to find electrical engineers, computer scientists, silicon designers, and system designers, and linguists in a speech research team and indicate the complexity of the speech recognition problem. These challenges can be grouped into the following main areas:

- Input signal conditions & Analog front end design,
- Algorithms,
- Real time processing with minimal resources, and
- Man-machine interface or application design.

In this thesis, Algorithms and Real time processing areas are addressed in detail since these are considered to be relevant in the traditional sense of a thesis. However, one should not underestimate the importance of both Application design as well as Analog front-end design towards realizing a successful speech recognition based system. If the front end digitizing circuitry is not designed properly, then no amount of advanced signal processing techniques further down the stream can bring significant improvement in performance.

In the area of application design, one has to consider other alternative inputs and weigh their benefits and cost. It is only when speech recognition offers a significantly better value proposition compared to other alternatives that a speech recognition based product will be successful. Some of the deficiencies of a speech recognition based product will
be tolerated by the user when other alternatives are not as viable. One example is dialing a phone by voice while driving. Here speech recognition offers convenience and safety and other alternatives are not as attractive as speech recognition.

1.2.1 Input speech-signal conditions

Speech from the user is captured by a transducer, filtered by an analog front-end circuit, digitized and then sent further to a digital signal processor. Various factors contribute to the signal conditions before being processed and these are discussed in the succeeding paragraphs.

1.2.1.1 Transducer characteristics

The transfer characteristics of the transducer affect the quality of the signal. There are different types of transducers such as electret microphone, dynamic microphone, carbon microphone, headset microphone, and etcetera. Some of these such as an electret microphone require an electrical bias to operate and some like the dynamic microphone do not. Electret microphones are significantly cheaper than dynamic microphones but their transfer characteristics are not as good. For electret microphones that require a bias, coupling with the 50/60Hz power line frequency has shown to affect the performance significantly.

1.2.1.2 Analog Front End (AFE) design

Once the transducer captures speech, the analog signal needs to be filtered and digitized. Analog ground needs to be carefully separated from the digital ground and the analog circuitry needs to be carefully laid out to reduce spurious noise. The analog gain of the amplifier needs to be carefully set such that the input signal to the analog-to-digital converter is not saturated. Speech signals have a very large dynamic range and many a time, the board designer inadvertently sets the gain of the analog amplifier at a large value resulting in clipping. Clipping significantly affects the performance of the recognizer. The board designer also needs to filter out the power line hum if an electret
microphone is used. As previously stated, if the input signal to the analog-to-digital converter is corrupted in the first place, advanced signal processing techniques further down the stream cannot bring significant improvement in performance.

1.2.1.3 Signal-to-Noise-ratio (SNR)

The surrounding environment of the user invariably affects the performance of the recognizer. An office environment is best since the signal-to-noise-ratio (SNR) is high. Factory floors, airports, bus terminals, highways, etc. are the worst since the ambient noise is high and the resultant SNR is very poor. Telephony networks fall in between these two extremes.

Depending on whether a close talking microphone or a handsfree kit is used, the SNR can vary significantly. Typically, the SNR in an office environment is between 15dB to 30dB. When SNR falls below 15dB, the task of recognizing becomes quite challenging.

1.2.1.4 Bandwidth of the signal – Narrowband and Broadband

The human ear is capable of hearing up to 20KHz. Historically, telephone companies drove speech-processing applications. Telephony networks transmit speech in the 300Hz to 3300Hz region and as a result speech has always been sampled at 8KHz. Speech recognition of telephone network speech should then utilize the information in this limited bandwidth. With the recent proliferation of the Internet this narrowband utilization of the speech spectrum can be alleviated. We can now sample the signal at higher rates such 10KHz or perhaps even as high as 48KHz as audio and utilize information in this broader spectrum.

1.2.2 Algorithms

This area influences the performance or the accuracy of the speech recognizer as well as the resources (MHz and Mbytes) required. What are the main characteristics that should be observed? What similarity measure should be
used? How should the vocabulary templates be created? We discuss some of these in the following paragraphs.

### 1.2.2.1 Signal representation

What features do we use to represent the input speech? Are these orthogonal? What is the minimal set that is required. What parameters should we measure or observe in order to create these features. How many parameters should we use? What parameters we observe, influences the performance and the number of parameters influences the memory required to store the templates? Since the size of the templates or models is proportional to the features used, it is important to define a minimal and orthogonal set.

### 1.2.2.2 Distance metric

A distance metric computes a similarity measure between an unknown vector and a stored, reference vector. What is the best metric that is appropriate for the application? Should we use L1 norm, or Euclidean distance metric? The choice again influences the performance as well as the storage requirements.

### 1.2.2.3 Segmenting speech – a fundamental problem

Sampled speech is buffered before being processed. This buffered speech is called a frame. A frame duration ranges from about 10ms to 30ms in duration. Segmenting speech means the ability to identify the frame indices that correspond to the boundaries of speech. Reliable identification of these boundaries is key to both, speaker dependent as well as speaker independent recognition systems.

In a speaker dependent recognition system, input speech needs to be end pointed for word boundaries for enrolling a vocabulary word. This is done by prompting the user to say the specific phrase containing the word, and the system then attempts to find the begin-time and end-time of the word using frame energy as a criterion. However, as shown in Figure 1-1, energy based end-point detection has its limitations. Often extraneous segments of the signal are
incorporated with the word to be enrolled. For example, fricative portions may be truncated or a segment of input signal that is actually background noise is added. When this input utterance with faulty word boundaries is used to create a template, recognition performance will suffer. A need, therefore, exists to reliably detect word boundaries.

Figure 1-1 Problems with word-boundary detection using energy based detectors

This problem can be further posed as a segmentation problem as shown in Figure 1-2. As shown in this figure, if we can classify the input speech frames accurately and reliably, we will have solved the speech recognition problem. Hence reliably segmenting speech is a fundamental problem in speech recognition.
In this thesis, several techniques are developed that improve the reliability of accurately detecting word boundaries using novel approaches. These include the use of anchor words, and recognizing non-speech segments such as lipsmack, inhalation, exhalation etc.

1.2.2.4 Averaging templates in speaker dependent systems

In speaker dependent systems, once the word boundaries are detected, then a recognition template has to be created. To improve the recognition performance, it is better to ask the speaker to repeat the specific word a few times. The second and subsequent utterances may be of different time durations from the initial template created from the first utterance. One needs to allow for this variation and then average the templates. Dynamic time warping is used to align and average the appropriate frames.

Since averaging tends to improve recognition performance, one might tend to allow each word to be averaged several times, say 3 to 4 times. However, when the vocabulary size increases, this increases the training time and one has to
strike a balance between training time and recognition performance.

### 1.2.2.5 Collecting large speech databases

In order to reduce the training time for each vocabulary word, as well as eliminate incorrect segmentation of speech, researchers have collected large databases of speech and offline, marked the boundaries, transcribed the segments, and then created speaker independent models. However, there are many problems with collecting such large databases. These are related to the design of the database, logistics of collecting them, and the time and effort associated with transcribing them. What is the minimal set of utterances that needs to be recorded in order to create reliable speaker independent models? How many dialects should one sample for a given language? How many speakers should be allocated to each dialect? What signal to noise ratios should these samples be collected in?

### 1.2.2.6 Apriori knowledge for improved recognition performance

Instead of just recognizing the input utterance using a similarity measure, can we incorporate some apriori knowledge of the application in which the recognizer will be used? Such application specific knowledge will narrow the search space and improve the recognizer's performance. In the thesis, we will explore two techniques as follows:

**Bottom-up approach**

In this approach, the word hypothesizer is open-ended in the sense that every frame is scored against all words in the vocabulary space. The recognized words are then bound later by a sentence hypothesize to form a sentence. The sentence hypothesize does not guide or control the operation of the word hypothesizer.

**Top-down approach**

In this approach the sentence hypothesize controls or guides the operation of the word hypothesizer by allowing or searching only those words that make sense in that context. As a result, recognition performance improves since we do
not include out-of-context vocabulary words, which might be similar and cause misrecognitions. Reducing the branching factor, in a finite state automaton, will improve the performance. Preventing or reducing the number of similar words branching from a single node can obtain another step increase in performance.

1.2.3 Real time implementation using minimal resources

Real time processing introduces several constraints in both the design as well as the implementation of the algorithms. The size of the memory required, computation in fixed-point notation, task partitioning in multiprocessing systems are example challenges.

1.2.3.1 Fixed point arithmetic representation

Since floating point arithmetic processing is expensive from a system cost, most digital signal processors implement fixed point arithmetic processing. Historically, floating-point multipliers occupy a larger silicon area than fixed-point multipliers and require wider word lengths (32bit versus 16bit) resulting in a more expensive system cost. Today the silicon area of the multipliers is no longer an issue; however, memory width and size is still a significant issue to be addressed. Cost effective solutions therefore require fixed point arithmetic processing.

However, variables represented with fixed-point notation have a smaller dynamic range than numbers represented with floating point notation. As a result, one needs to have a thorough understanding of the nature of computations and the range of interest of the variables. What Q-point do we choose for the autocorrelation function, linear predictive coefficients, etc.? We need to appropriately assign Q-points to all variables used in the speech processing system.

In HMM based scoring techniques, one typically requires floating point representation since the HMM based score is a monotonically decreasing function. In this thesis, we present techniques that we have developed that allow this computation to be performed on a fixed point DSP.
1.2.3.2 Minimal resource usage – MHz and MB

By resources, we mean memory required to realize the solution and the MHz or cycles required by the system. Speech recognition systems tend to more memory limited than MHz limited. Memory dominates the overall system cost. Hence, it is imperative that we design and implement a solution that requires minimal memory. This constraint needs to be addressed at both the algorithm domain as well as in the real time implementation domain. The number of features, the width of the variable for representing the features, using pointers instead of indices etcetera all have a significant impact on the size of memory required.

In HMM based recognition systems, a scoring buffer of several hundred kilobytes is required in order to store the scoring information. When the vocabulary size increases, this buffer can grow in size preventing a cost effective implementation of this technology. In this thesis, several algorithms to reduce the large buffer requirements are explored. These techniques significantly reduce the size of the buffer without compromising the recognizer’s performance.

1.2.3.3 Task partitioning and allocation in Multiprocessor systems

The number of vocabulary words that can be addressed by a single processor, such as a TMS320C10/C25 is about 25 to 50 words. Then, how do we build a recognition system that can recognize several hundred or several thousand words? One approach is to build a multiprocessor system. How do we partition the recognizer and allocate the tasks to individual processors? How can we partition such that the bus-loading is minimal or optimal?

1.2.4 Man-machine interface or application design

A good and thorough understanding of the application is key to the success of the speech recognizer solution. What and how many vocabulary words should be enabled at a specific time? How confusabale are the vocabulary words at a
specific node in the grammar? What is the branching factor? How should the prompts be designed? Should the recognizer be enabled along with the system prompts? All of these are crucial questions that the system designer needs to address.