ELECTRICAL AND INFORMATION TECHNOLOGY, FACULTY OF ENGINEERING | LUND UNIVERSITY

Algorithms in Signal Processors - Project Course Automatic Speech Recognition

Hanna Runer ael10hru@student.lu.se
Filip Gummesson dat11fgu@student.lu.se
Viktor Stevander elt11vst@student.lu.se

March 10, 2015

Supervisor: Mikael Swartling, mikael.swartling@eit.lth.se

CONTENTS

1 Introduction

In this report the project in "Algorithms in Signal Processors - Project Course" - ETIN80[1] is presented. The project considers an automatic speech recognition(ASR)problem which will be implemented on a DSP. The goals are to be able to detect if speech is present, decide if the word is in the library and what word was being said.

The problem solving approach was to first test the functionality of the algorithms and build high level libraries in MATLAB. Then, when it was clear that the algorithms were working as expected, they were implemented on the DSP commenced. The digital signal processor used is the 32-Bit Floating-Point SHARC DSP, ADSP-21262. The program VisualDSP++ was used to program the DSP.

2 Theory

2.1 Speech

Speech is created when air flows through the vocal tract which compresses the air and sound waves are formed [2]. All human speech can be categorized into two subcategories. These are voiced- and unvoiced speech.

A voiced speech originates from vocals which are folded in the larynx, and an unvoiced speech comes mostly from consonants. If these two speeches would be plotted, the voiced speech would be very periodic while the unvoiced speech would look noisy. This makes it more valuable to use statistics from voiced speech than unvoiced.

Speech also has the inconvenient property of being non-stationary, but it is however true that it is stationary for about 20 ms. When having this insight, the prediction of speech is easier to realize.

2.2 Levinson-Durbin and Schur

Voiced speech is measured by calculating the reflection coefficients, using either Schur- or Levinson-Durbin algorithm.

The Levinson-Durbin algorithm is a recursive algorithm that calculates a lattice-filter in order to find both the IIR-filter coefficients and the reflection coefficients.

The Schur algorithm is calculating the coefficients based on auto correlation and is by that avoiding computation of inner products. It is hence an effective algorithm to use if only the reflection coefficients are needed. Since it instead requires matrix computation to solve the equation it does not make up for an easy implementation in C-code, but is efficient in MATLAB.

2.3 Identification and Validation

When talking about speech recognition, one has to make difference between identification and validation. Identification is made when software can determine which word most likely

was spoken from an predefined library.

3 IMPLEMENTATION

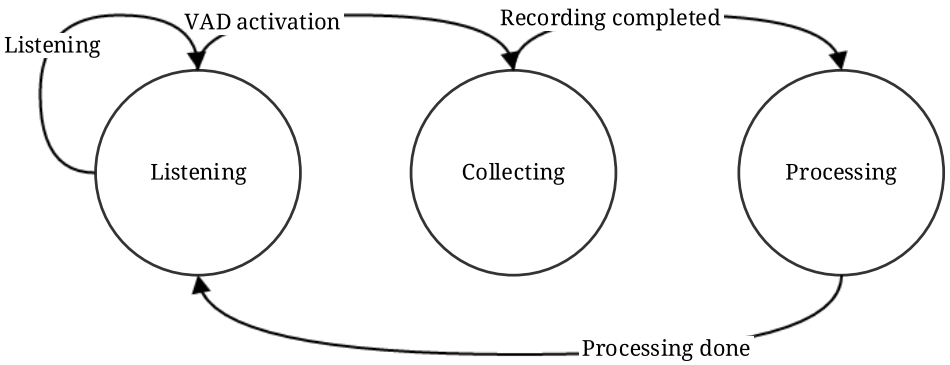


Figure 3.1: This speech recognition algorithm is implemented as a state machine.

The speech recognition algorithm consists of three states, see figure 3.1. The first state, *Listening*, samples the the input source and runs a speech detecting algorithm. If there is a speech, the state changes to the state *Collecting*. In this state the algorithm collects samples in blocks and store the in internal memory[1] and the state changes. Next the *Processing* state becomes active. This state performs the signal processing and matching of a signal. Identification and validation is also a part of the processing state.

In the different states there is a total of seven parts - level detection, recording, filtering, cutting, calculation of coefficients, dividing into subsets and matching. The implementation method of these parts differ from the implementation in MATLAB and on the DSP, but the principle is the same.

The first five parts handle blocks of 160 consecutive sample. After the reflection coefficients are calculated and put into a feature vector, the signal is represented by a matrix of feature vectors.

A feature vector is retrieved through the following steps. IIR - filter removes some noise, pre-emphasis filter boosts higher frequencies and Hamming window removes the effect of transients. See figure 3.2.

[1] The DSP is not able to store a complete signal, instead the signals features are exracted.

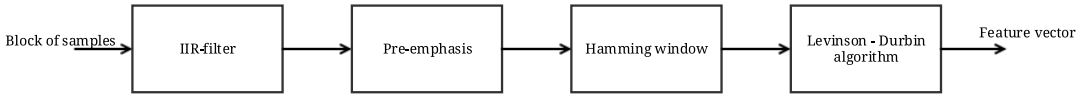


Figure 3.2: Transformation steps of a signal block.

3.1 Level Detection

To determine if speech is present and recording should commence a Voice Activity Detection (VAD) algorithm. The algorithm is based on a dynamic noise detection which adapts in accordance to its surroundings. That is, in a constantly noisy environment the algorithm will raise the threshold on which speech can be detected, thus minimizing the risk of an *insertion* [2]. If the VAD- algorithm is activated, a sound signal will be recorded.

3.2 Recording

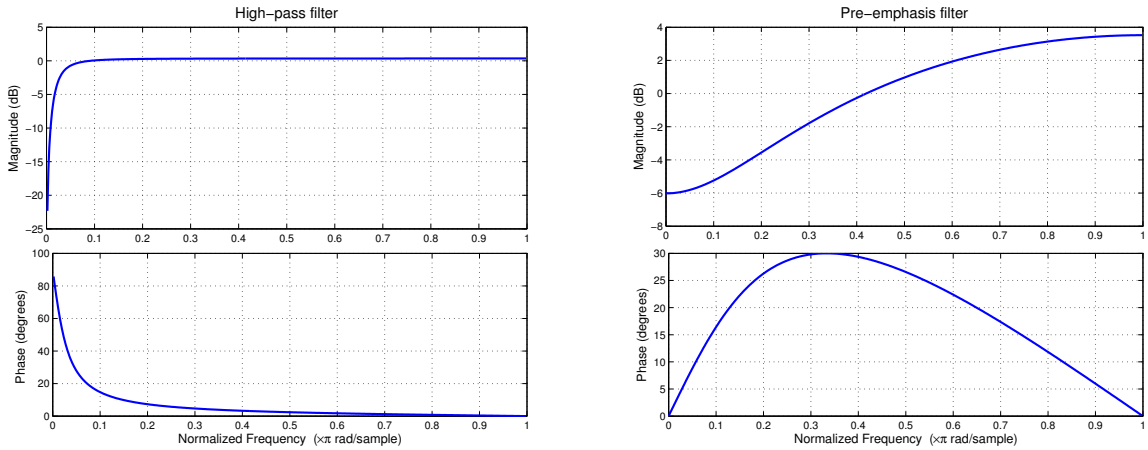
When speech is detected recording of the following 1.5 seconds follows. A sample rate of 8 kHz was chosen to keep amount of data down and to prevent disturbance from high frequency components. Speech has usually a maximum frequency of around 4 kHz and because high frequency consonants, such as k, t, s, f, does not give any information to the reflection coefficients, there is no need to sample at a higher rate. The recording outputs a block of 160 consecutive samples. Next, the recorded block of samples enters the first step seen in figure 3.2.

3.3 Filtering

Two types of filters are applied to the recorded signal, a high-pass and a notch filter, in that specific order. Low frequencies has normally a higher effect then the higher ones, which is why the recorded signal is filtered with a high-pass filter. The high-pass filter removes low frequency signals such as vibrations from table and floor.

Following after is the notch filter, the pre-emphasis filter, which removes low frequency disturbances and boosts higher frequencies. The high-pass filter is a FIR-filter and pre-emphasis an IIR-filter.
See figure 3.3 below for the characteristics of the two filters.

[2] insertion - when a recognizer hypothesize a word that was not spoken[2]



(a) High-pass filter characteristics

(b) Pre-emphasis filter characteristics

Figure 3.3: The filters applied to the signal

3.4 Cutting

The recorded signal is often longer than it has to be, thus the signal is cut. As this is done, recordings will start and end with vocal speech, and no unnecessary samples, containing noise, will be saved. See figure 6.2 for an example of a signal being processed by the use of filters and cutting of the signal.

3.5 Calculation of coefficients

As stated before, the Schur and Levinson-Durbin algorithm are used to calculate the reflection coefficients. The two algorithms are applied to extract feature coefficients from a stationary signal. It is in this part the choice of block size becomes apparent as speech is non-stationary in general, but if considering only a block size representing around 20ms, the speech can be considered stationary.

The two algorithms extracts the same feature vectors from a block, but in different ways. Schur is more complex and uses matrix multiplications to obtain the coefficients while the Levinson-Durbin is a recursive algorithm. The former is quick but hard to implement on DSP. The latter extracts the feature vector as a by-product and that makes it slower, but the Levinson-Durbin algorithm is much easier to implement i C code.

Schur and Levinson-Durbin algorithm is applied on each block to calculate the reflection coefficients. Each set of reflection coefficients, calculated from one block of 160 samples, is called a feature vector.

3.6 Dividing into subsets

When every block of samples from the recording have been processed and a feature vector from each block have been extracted, a matrix of K feature vectors have been produced. Then,

for both memory saving properties and robustness of the characteristics of the speech, the feature vectors are divided into subsets by taking a mean value row wise along the feature vectors. After this averaging, the set of M subsets is considered a database containing the characteristics of the recorded spoken word. See figure 3.4[3].

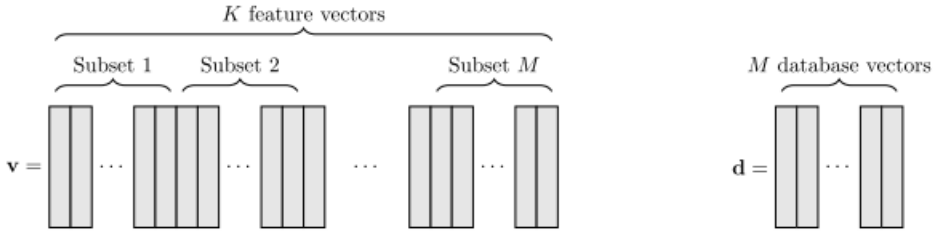


Figure 3.4: Dividing the feature vectors into subsets

3.7 Database

The database consists of two types of words, "Höger" and "Vänster". The two words have several recordings stored, that is, there exist several versions of the two words. The number of versions of each word increases the chance of finding a better match. For example for two types of words, that have five versions each, there exists 10 database items to match the recorded word against.

A method of testing the words in the library was to plot the Euclidean distance between different versions of a word and a test signal with a different word. See figure 3.5 The point was to note the diversity in the points and therefore find a few versions of the word which works together and define a threshold for validation.

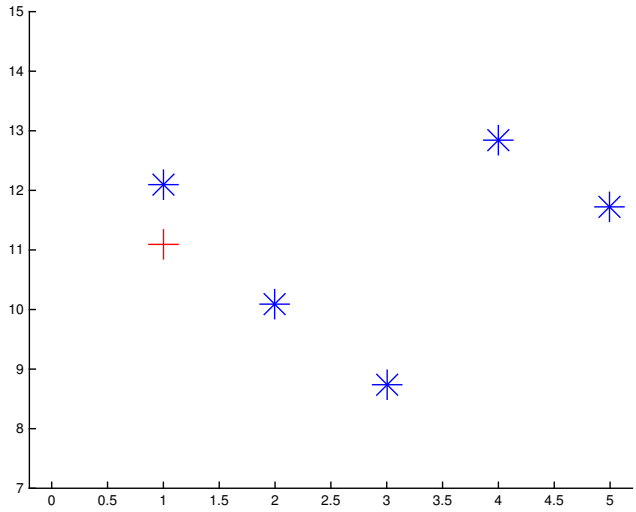


Figure 3.5: Plot of the sum of the error in Euclidean distance between different versions of the same word (blue stars) and a test signal. The red plus sign is the mean error.

3.8 MATCHING

To match a recorded word against the database the Euclidean distance for between the recorded subsets reflection coefficients and the database subsets reflection coefficients are calculated and summarized. See figure 3.6. The recorded word is tested against every word and every version of a word in the database. Two types of error are saved used in the matching decision:

ϵ_{min} which is the smallest ϵ of all versions.
ϵ_{tot} which is the is the total error for all versions for a type of word.

For **identification**, the word which produced the smallest ϵ_{tot} is the recognized word.

But if wanting **validation** for a word much harsher constraints are needed. To decide on a specific word both ϵ_{min} and ϵ_{tot} have to belong to the same type of word, for example "Vänster", to give a decision that the recognized word is "Vänster". If the two errors do not belong to the same type of word, the decision states that no match was found. For greater accuracy a threshold was also added, that is, alongside the two error needing to belong to the same type of word, the error must lie beneath the threshold to decide upon a certain word.

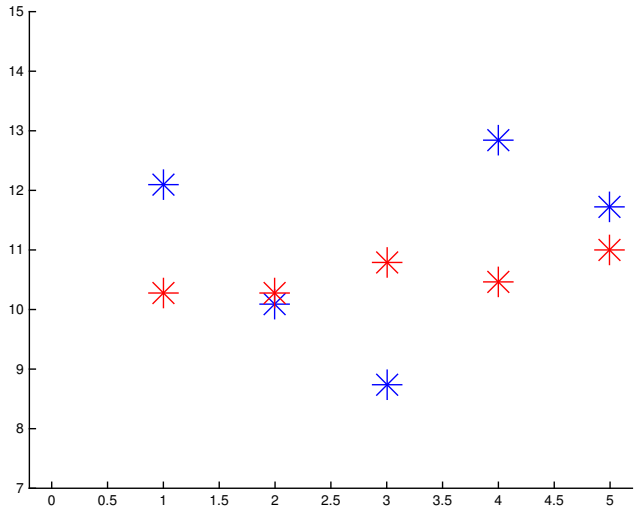


Figure 3.6: The Euclidean distance is the distance between the red and the blue dot in the same vertical line, where each dot represents a reflection coefficient.

4 MATLAB

MATLAB was used to build high level libraries. These could be tested with prerecorded signals in an offline manner. This made it easy to test our algorithms and change parameters the obtain a desired behavior before starting to write C code for the DSP. MATLAB was also used during the DSP implementation as tool for testing if the DSP implemented algorithms produced the same output as MATLAB algorithms.

5 ADSP - 21262

On the DSP the program runs in a bit different order, due to the continuous data stream. The program is build in three stages – listening, collecting and processing, as shown in Figure(5.1). This is due to the online process which interrupts our process each time a new block of samples is available.

As an extra feature, an buffer is implemented to be able to start the record in a few samples before the VAD is activated. This to make sure no important information is dropped.

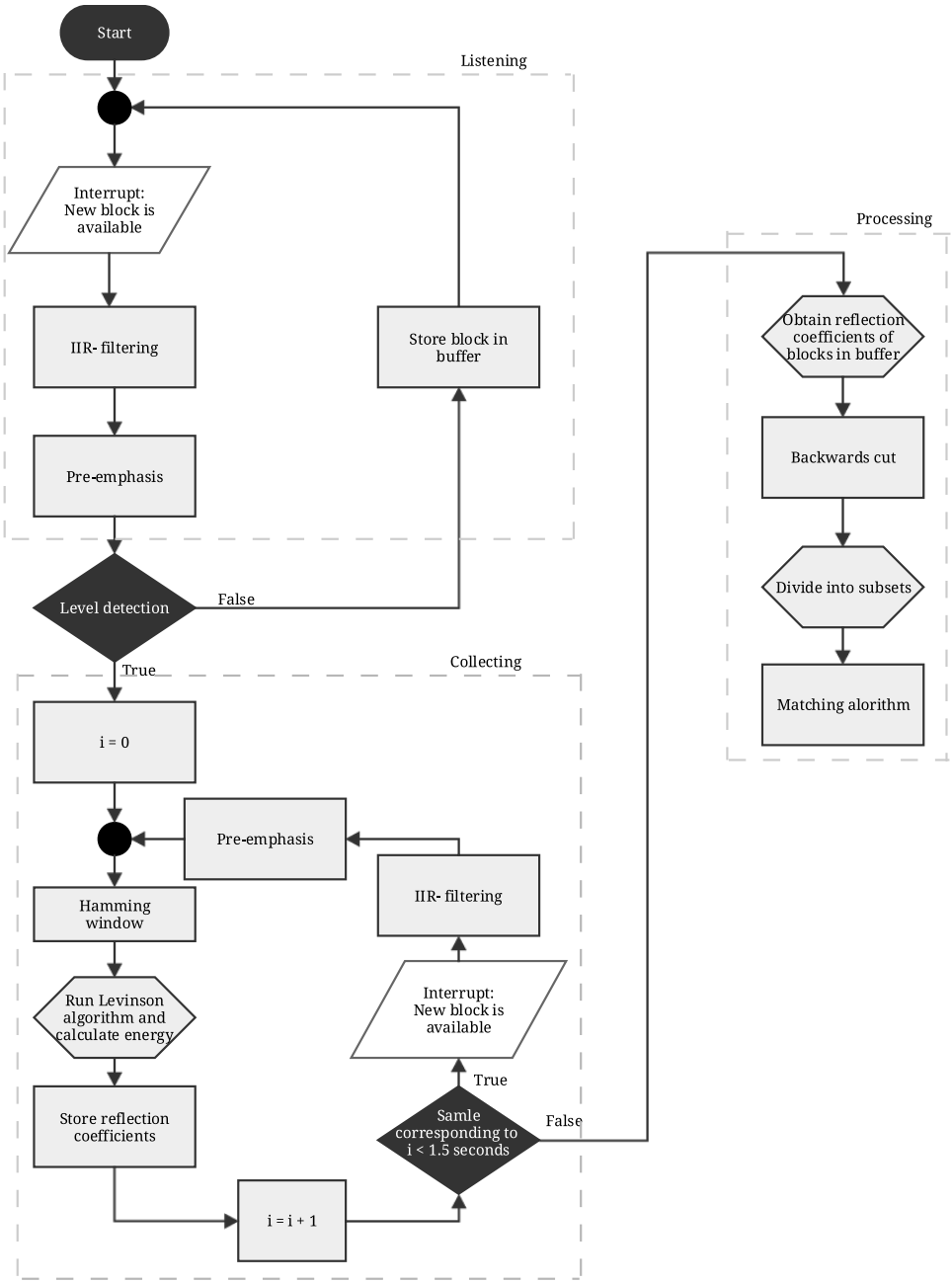


Figure 5.1: A flowchart over the process implemented on the DSP.

5.1 Listening

The AD-converter saves samples in a buffer and therefore produces blocks of 80 samples to be handled, one a time. Each block is then going through the same procedure with filters and pre-emphasis as above, before entering a level detection.

In the level detection, the energy of each block is compared to the energy of the previous block. When enough blocks are changing enough energy if level detection is not recognizing any speech, the block is placed in a buffer with a capacity of three blocks. The blocks on the buffer will later be processed to make sure the whole signal is covered.

5.2 Collecting

When speech is detected the program is then entering stage two where it collects data. The largest problem with the DSP is the limited amount of data that can be stored in the data memory. The first step is therefore to calculate the reflection coefficient to reduce the number of data to a tenth of the collected. The DSP collects a block, filters it, calculates the energy and the reflection coefficients, that is, the feature vector. To calculate the reflection coefficients the Levinson-Durbin algorithm was used. The energy of the block was also calculated and stored to be used later to determine where to cut the signal.

This way of collecting data was looped for the amount of blocks corresponding to 1.5 seconds, before entering the third stage of process.

5.3 Processing

The first step in the third stage is to take care of the blocks in the buffer, by calculating the coefficients. The energy stored from the blocks in stage two is then used to cut from the end of the recorded signal. The remaining feature vectors are then divided into subsets and matched against the library in the same way as described in section 3.8.

6 Results

6.1 MATLAB

The MATLAB implementation worked as it was supposed to. The FFT of a signal, spoken by a man, as seen in Figure 6.1, shows that the signals frequency spectra is leveled by the notch-filter and that the higher frequencies are boosted by the pre-emphasis.

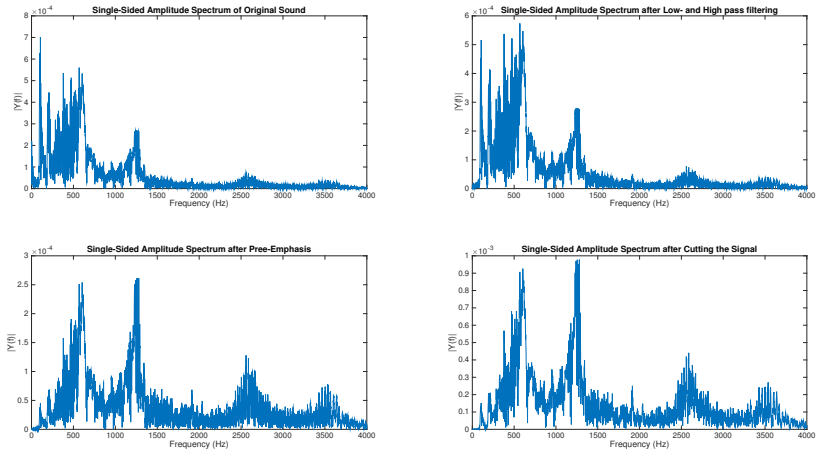


Figure 6.1: One sided Fourier-transform of a male saying the word "Höger". The spectra is leveled out and boosted. The only notable change after cutting the signal is that the high level noise is reduced.

Figure 6.2 displays how the filtering and pre-emphasis affects the signal. This can, however, be hard to see in the time domain and better shown in in the frequency domain (Figure 6.1). What is interesting is the signal is cut to a length that makes sure only the speech is within the region of interest (down right in Figure 6.2).

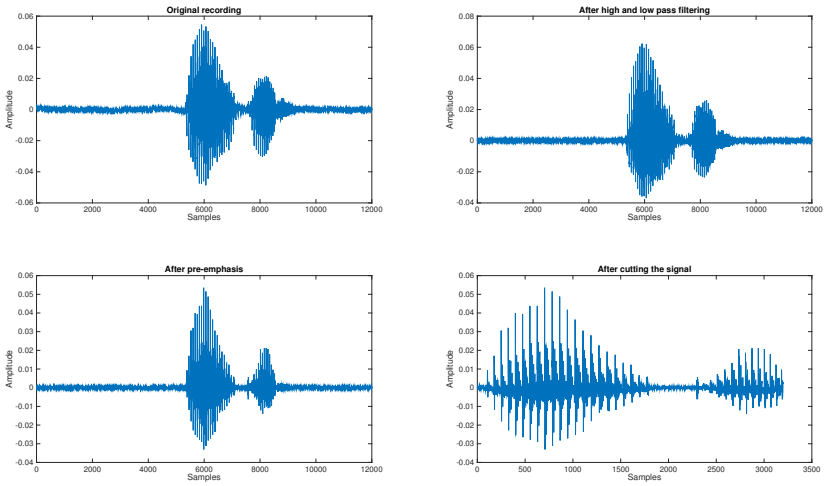


Figure 6.2: Speech pattern in the time domain of a male saying the word "Höger". Note that the cut-function (down-right) reduce the number of samples by 8600, from 12000 to 3400.

When tests was performed in MATLAB, like the one described in Figure 3.5, it showed that the program could identify the right word in 100 % of the cases and validate a right answer in around 80 %. These tests was performed using a different microphone and algorithm settings than the used with the DSP in the end. These results was achieved early on in project, before implementation on the DSP was done, and gave a hint that the algorithms should work.

6.2 ADSP-21262

Two different test were performed on the DSP. The first test considered the two words "Höger" and "Vänster" being said 100 times each and counting the number of times the correct word was successfully matched, see table 6.1 for results. The desired result is a low Word error rate(WER).

The second test was to see how the DSP would interpret normal speech, without mentioning the word "Höger" or "Vänster", the results from the second test is seen in table 6.2. Here the number of occasions "Höger" and "Vänster" was detected is desired small and the number of occasion no word in database was found, desired high.

Word	"Höger"	"Vänster"
Word error rate (%)	9	27

Table 6.1: Word error rate of the two words, each spoken 100 times

Recognized word	"Höger"	"Vänster"	No word i database
Number of occasions	3	27	70

Table 6.2: Detection and matching speech without mentioning "Höger" or "Vänster", 100 occasions speech was detected

7 Conclusions och Discussion

The implemented speech recognition software can identify two different words and validate if it is one of the words in the database, or if the spoken word is not in the database.

The results in table 6.1 and 6.2 are quite satisfactory, considering the lack of finesse the algorithms have. It is clear that the word "Vänster" is more tricky to extract features from since it was both hard to recognize when the word was spoken, and that the word was mistakenly recognized at a high number of occasions. The word "Höger", on the other hand, obtained far better results in both tests. This indicates that a proper, or proper enough, feature extraction was not achieved. This recognition system is rather primitive and there is a lot of ideas and tricks that be implemented that will give a more robust ASR system.

The current implementation holds some restrictions:

- The same person who recorded the words in the database, is the person using the system for best recognition
- The database contains several versions of the two word, but they are not too different in pronunciation.
- The distance to the microphone is crucial in matching, that is, the algorithms do not take the amplitude of the speech in consideration
- The words should to be chosen such that they differ from each other, to increase the accuracy of the algorithms

Summing up, in this implementation, the speaker must speak at a steady volume and quite similar to the recorded versions in the database to achieve high recognition rates.
These point are all matter of improving if continuing this project. There is already some ideas and thoughts around improving, for example:

- Filtering of the reflection coefficients to increase robustness
- Testing different numbers of subsets and reflection coefficients that characterize the signal
- Using the energy of the signal to scale and use as a score when comparing against a threshold
- Adding different pronunciations of word, and with that, switch matching strategy. This since, the matching algorithm also checks the sum of the error of all the version for one type of word. Different pronunciations will most likely increase this summed error, but decrease the smallest error.
- Trying to extract the features of a block of samples in an other way than with reflection coefficients, for example, mel frequency cepstral coefficients[2](MFCC).

REFERENCES

[1] The course webpage: http://www.eit.lth.se/index.php?ciuid=821&coursepage=kursfakta&L=1

[2] Woelfel W., McDonough J., 2009 "Distant Speech Recognition", Wiley

[3] Picture from document provided by Mikael Swartling, teacher in the course [1]