Analyzing Theremin Sounds for Touch-Free Gesture Recognition

Svilen Dimitrov

A thesis submitted in partial satisfaction of the requirements for the degree Bachelor of Science at the Department of Computer Science at Saarland University

SAARLAND UNIVERSITY

Supervisor

Dipl.-Inform. Christoph Endres

German Research Center for Artificial Intelligence, DFKI

Reviewers

Prof. Dr. Dr. h.c. mult. Wolfgang Wahlster

German Research Center for Artificial Intelligence, DFKI

Dr. Christian Müller

Head of automotive IUI group, Intelligent User Interfaces Department
German Research Center for Artificial Intelligence, DFKI

Autor

Svilen Dimitrov

Bruchwiesenanlage 4,

D-66125 Saarbrücken, Germany

Submission Date

28. October 2010

Statement in Lieu of an Oath

I hereby confirm that I have written this thesis on my own and that I have not used any other media or materials than the ones referred to in this thesis.

Saarbrücken / 28. October 2010

(Svilen Dimitrov)

Declaration of Consent

I agree to make both versions of my thesis (with a passing grade) accessible to the public by having them added to the library of the Computer Science Department.

Saarbrücken / 28. October 2010

(Svilen Dimitrov)

Abstract

The increasing number of car accidents caused by distraction of the drivers, mostly by the increasing amount of devices in every car, leads to rising interest in innovative human computer interaction technology. The two most studied methods are gesture and speech recognition. In the gesture recognition field, numerous techniques are using different expensive devices and complex algorithms or combination of them to increase robustness. This thesis investigates a new method for two dimensional gesture recognition of index finger movement, having one-dimensional input by using magnetic fields, called Geremin Approach. The gestures, that are to be recognized, vary from single straight moves to more complex geometric figures. The way to do this is straightforward, flexible and low cost. The core input component studied in this work is an early electronic musical instrument named Theremin after its founder. It is placed in immediate proximity of a steering wheel providing gesture based interaction without requiring the driver to take off hands. Despite all benefits that this method has, it is still in early development phase and can be further improved by adding more antennas and by using modern classification algorithms to obtain greater accuracy.

Acknowledgements

I would like to express my gratitude to Christian Müller and Christoph Endres for challenging me with this great and innovative work.

Secondly I want to thank Stefan Popov and Yavor Kaloyanov for backing me up with useful programming tips and fruitful advices. At this place I am much obliged also to everyone, who shared his work and experience over the internet.

Finally I am deeply grateful to my Family for their unconditional support, patience and comprehension ☺

Contents

1 Introduction 1

2 Basic Concepts 3

2.1 Theremin 3

2.2 WAVE Format 5

2.3 Gesture Recognition 6

2.4 Dynamic Time Warping 7

3 Related Work 9

3.1 Accelerometer 9

3.2 Wearable Sensors 11

3.3 Camera 12

3.4 Infrared 13

3.5 Combinations 14

3.6 Electric Fields 15

3.7 Automotive Design 15

4 Geremin Approach 17

4.1 Motivation 17

4.2 Design 18

4.3 Implementation 18

4.3.1 Source Code Architecture and Usage 20

4.3.2 Dynamic Link Library Usage 24

4.3.3 TCP Communication 24

4.4 Hardware Set-up 26

4.5 Gesture Recognition 29

4.6 Evaluation 33

5 Conclusion & Outlook 37

Chapter 1

Introduction

In the automotive field, an increasing effort is made to improve the safety of driving. One of the major causes for driving accidents is distraction. On the other hand, consumer electronics devices, providing multimedia entertainment and enabling communication, have become ubiquitous in daily life. Due to the increasing number of these modern devices in the cars the drivers are forced to interact somehow with them while driving [1]. Traditionally, in-car user interfaces consist of remote control and keypad. With digital contents becoming more and more complex and interconnected, drivers are expecting more natural and powerful user interfaces [2]. So an important challenge for the modern consumer electronics industry is the design of user interfaces enabling natural interactions that are convenient, intuitive and enjoyable. The most recent and promising ways of interaction, also subject to a high research interest, are speech and gesture recognition, where gestures have the benefit to be unaffected by noisy conditions. Particularly in the gesture recognition field there are various methods that recognize and interpret human gestures. All of them however have different requirements such as wearable hardware or having a clear and static lighted background. Further technological requirements are expensive hardware or long computational time. This work will investigate another method of touch-free finger gesture recognition by using electric fields. The gesture sets consists of lines and simple geometric shapes, with respect to their direction, drawn with the index finger of the driver, while he is holding the steering wheel. This is realized using a musical instrument named Theremin after its creator who found it by observing

tone change when moving his hands around his device, which supposed to be a “radio watchman”. Exactly this feature will be digitalized by processing the theremin signal with computer. This method leads to inexpensive installation costs and could be applied straightforward in every car. Besides gesture recognition, devices of this kind can be used further for motion detection or even as a simple hand tracking tool. Furthermore alerting the driver when he puts his hands away from the steering wheel could also improve the driving safety.

Chapter 2

Basic Concepts

This chapter introduces the basic concepts, which mostly affected this work, like the history and capabilities of the theremin instrument and the WAVE format. Then the concept of gesture recognition and the classification method used for collected data are explained.

2.1 Theremin

Born in 27 August 1896, Leon Theremin was a famous inventor from Russia. His most popular invention, in which we are currently interested in this work, is the theremin (Fig. 2.1).

Abram Fedorovich Ioffe, a Russian professor, made a proposition to Theremin to come to his newly founded Physical Technical Institute in Petrograd, and the next day he invited him to start work at developing measuring methods for high frequency electrical oscillations. The next day Theremin already started working there. He built a high frequency oscillator to measure the dielectric constant of gases with high precision and shortly made the first motion detector for use as a "radio watchman" [3].

While adapting the dielectric device by adding circuitry to generate an audio tone, Theremin noticed the pitch changed when his hand moved around. In October 1920 he first demonstrated this to Ioffe who called in other professors and students to hear. Theremin recalled trying to find the notes for tunes he remembered from when he played the cello, such

as the Swan by Saint-Saëns. By November 1920 Theremin had given his first public concert with the instrument, now modified with a horizontal volume antenna replacing the earlier foot-operated volume control. He named it the "etherphone", to be known as the "Termenvox" in the Soviet Union, as the "Thereminvox" in Germany, and later as the "theremin" in the United States [3]. Although not the first electronic synthesizer, the theremin had the biggest influence among all early electronic instruments.

Figure 2.1: *Leon Theremin with his gadget*

In detail, the theremin is controlled without any contact from the player and consists of two metal antennas. Moving the hand towards or away from an antenna alters the capacity of an oscillating circuit, because the hand and the antenna form the both conductors of the constructed capacitor. To control the theremin, the player uses one of his hands for the first antenna, which controls the volume circuit. With his other hand the second antenna, which alters the frequency and is used in this work.

2.2 WAVE Format

In this work, the sound signal produced by the theremin is caught by a computer and processed as an uncompressed LPCM WAVE (Linear Pulse Code Modulation Waveform Audio File Format) sound. It is the pure form of digital sound that the computer captures. The most important settings in this format are the number of channels, as well as the sample rate and the bits per sample (Figure 2.2 shows detailed format layout). By its design and characteristics the WAVE format permits to be used for digital storage of any kind of waves with random number of channels and precision [4] [5].

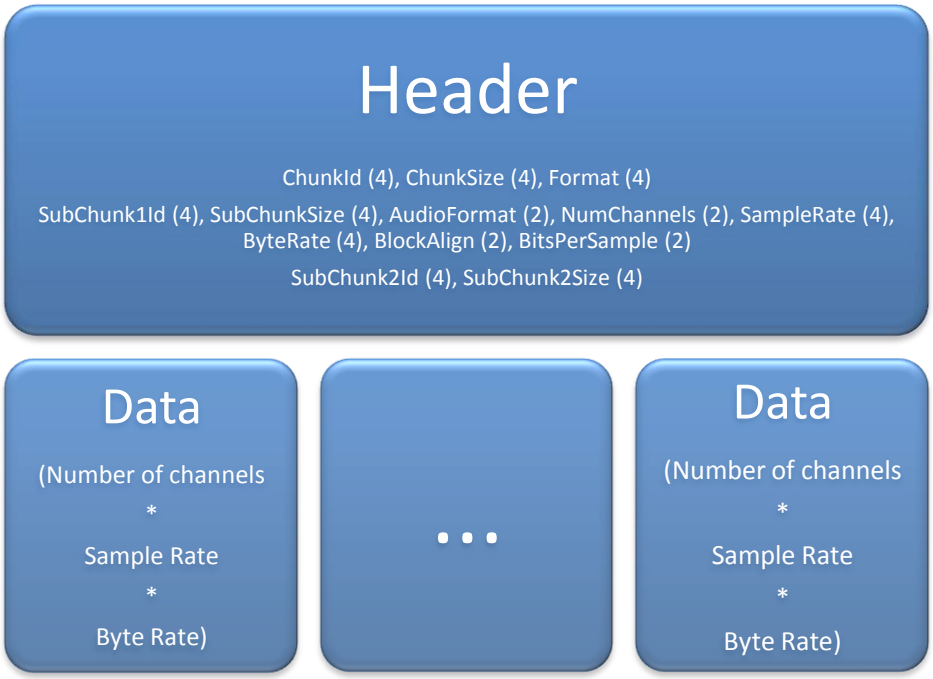

Figure 2.2: *WAVE format overview. Each segment of the both Header and Data chunks is presented with its size in braces*

2.3 Gesture Recognition

Consumer electronics devices, providing multimedia entertainment and enabling communication, have become ubiquitous in daily life. An important challenge for the modern industry is the design of user interfaces for consumer electronics products that enable natural interactions that are convenient, intuitive and fun [6]. As many products are supplied with microphones and cameras, the exploitation of both audio and visual information for interactive multimedia is a growing field of research, because the user interfaces of consumer electronics have been limited to devices such as remote control and keypad for a long time. With digital contents becoming more and more complex and interconnected, consumers are expecting more natural and powerful user interfaces. Automatic recognition of human gestures provides a promising solution for natural user interfaces. There has been an increasing interest in the last years in gesture control by the consumer electronics industry [2] (Fig. 2.3).

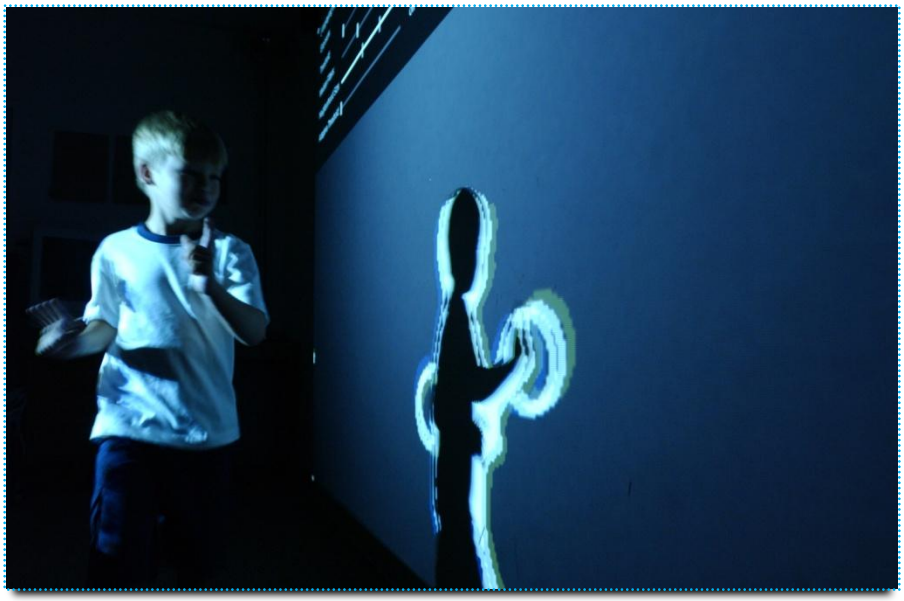

Figure 2.3: *Commercial show displaying kid moving around motion sensing huge screen*

To be more precise, gesture recognition is the desire to recognize and encode in some way the gestures produced by the human with computer technology. So on the one side there are the gestures to recognize and on the other side the different technologies to do this.

The gestures, in which the research is mostly interested, are the face gestures to guess his mood or his lips position, body gestures to see where someone is turning, hand gestures to see where someone points or what is he showing or drawing with his hands. This work will present finger movement recognition.

The different technologies available to recognize someone's gestures are depth-aware cameras, single or stereo cameras, controller-based gestures, electric fields, wearable sensors using accelerometers, gyros and compasses. There are also technologies that combine those mentioned above.

2.4 Dynamic Time Warping

In general, there are two well-studied non-linear sequence alignment (or pattern matching) algorithms - dynamic time warping and hidden Markov models. The research trend transited from dynamic time warping to hidden Markov models in approximately 1988-1990, since dynamic time warping is deterministic and lack of the power to model stochastic signals. A comprehensive study of this transition literature by [7] shows that stochastic and nonstochastic dynamic time warping and hidden Markov models are actually sharing the same idea of dynamic programming.

Dynamic time warping is an algorithm for measuring similarity between two signals. For instance, similarities in gesture patterns would be detected, even if in once the does the gestures slowly and more quickly or slightly different in another. It was firstly proposed in 1978. Their benefit over the standard brute force algorithm is a significant lower

consumption of computational power [8]. Specifically, dynamic time warping is a method that allows a computer to find an optimal match between two given sequences with certain restrictions. The sequences are "warped" non-linearly in the time dimension to determine a measure of their similarity independent of certain non-linear variations in the time dimension. As an example we can see in Figure 2.4, where the very same gesture, of moving finger left and right, performed with different speed should be rightly recognized. This sequence alignment method is often used in the context of hidden Markov models, which are statistical models intending to guess unknown parameter based on current observations.

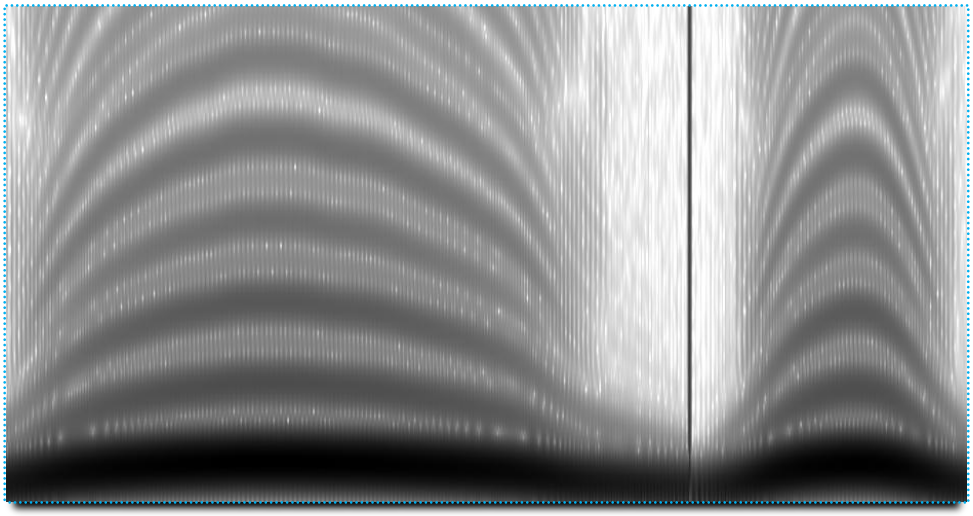

Figure 2.4: *The spectrum analysis of the same finger movement to the left and right but with different speed. The slope curve represents firstly rising and then falling frequency, which maps to finger distance*

This work uses dynamic time warping because the advantages of hidden Markov models come with bigger training data and for cases like ours it has been outperformed [9]. In this case will be made only a single training and based on that training attempt to recognize the gestures.

Chapter 3

Related Work

The history of automobile interior design has been one of incorporating nonessential features that subsequently became indispensable. Since broadcasting became common, it opened up the new era of automotive interior design, which by then was considered unnecessary [10]. From that time on, the number of in-car devices and mechanisms raised radically. This increased the distraction, which is considered one of the major causes for car accidents [1]. In order to interact with these devices there is a need to substitute "the old-fashioned automobile design culture, which proclaims that the engineer knows best, and considers studies of real people driving their vehicles irrelevant" [11]. The first human computer interactions were firstly demonstrated in 1963 and used light-pen and sketch pad to manipulate objects. In 1964 the first trainable gesture recognizer was developed. A gesture-based text editor using proof-reading symbols was developed in 1969. Gesture recognition has been used in commercial CAD systems since the 1970s, and came to universal notice with the Apple Newton in 1992 [6] [12]. The following sections provide a brief look at recent popular gesture recognizers and the different techniques used there, as well as some exemplification of their importance for the automotive field:

3.1 Accelerometer

There are numerous devices that have accelerometers in its hardware. Exploiting this hardware by different means has become very popular.

One of most beloved device is the Wii remote (Fig. 3.1). It is the primary controller for Nintendo's Wii console. A main feature of the Wii remote is its motion sensing capability, which allows the user to interact with and manipulate items on screen via gesture recognition and pointing through the use of accelerometer and optical sensor technology (infrared). Another feature is its expandability through the use of attachments.

Figure 3.1: *Nintendo Wii Remote*

There have already been successfully performed tests to recognize pantomimic gestures using the Wii remote by exploiting the accelerometer signal. The test users controlled a table calculator by writing the digits and operators into the air which is the same movement as writing on the black board [13] [14].

Another popular trend is the cell phones, where the manufacturers are seeking new functionality to differentiate their mobile handsets and update relatively primitive interfaces with too many buttons and menus. Motion processing by exploiting the embedded accelerometers is

emerging as the only solution that can deliver a new user interface and experience that will make a specific mobile handset stand out. There are already commercial products that bring about next generation gesture-based user interface for menu navigation, enable mobile authentication, enhance location based services, deliver an immersive gaming experience and improve camera image stabilization. In order to do this credibly, existing solutions based purely on 3-axis accelerometer motion sensing are not sufficient so there rose a need for full, six degrees of freedom motion processing that can precisely translate human motion for the various applications [15].

3.2 Wearable Sensors

There is a wide range of tools used in human-computer interaction. The wearable sensors motion capture suits are well used as input devices. Their functioning is based on different types of sensors like gyroscopic, magnetic, opto-electrical, etc. Using these numerous sensors to measure the movements of the points of interest is complex and the use if their data too [21]. They require also as the name suggests, the subject wearing different sensors, like gloves for arm gesture recognition (Fig. 3.3). The most significant wearable sensor is the accelerometer, which was already discussed separately above. Further liability is their cost and practical application of exploiting the current existing consumer technologies.

Figure 3.2: *Wearable sensor "CyberGlove"*

3.3 Camera

Many approaches have been made using cameras and computer vision algorithms to recognize human gestures since 1997 [16] [17] [18] (Fig. 3.3). However these methods are usually very expensive in terms of computational time [19] and only easier tasks like hand tracking can be performed relatively fast [20]. More liabilities to mention are their requirements of distance between the object of interest and the cameras, small field of view and having static backgrounds in terms of light and motion. Further issues here are the higher hardware costs, which include cameras (one, two or more) and the necessary fast processing units.

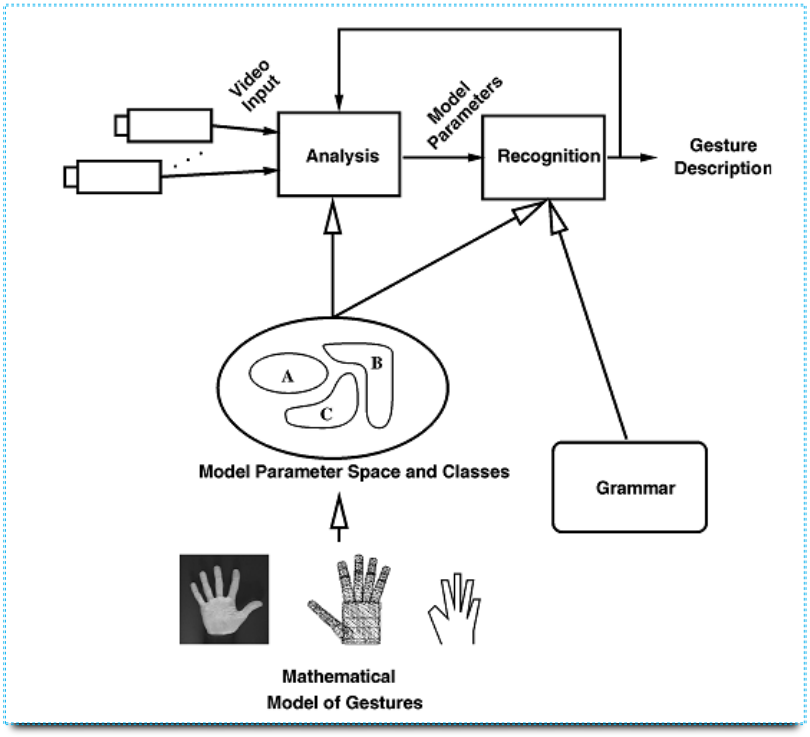

Figure 3.3: *Typical camera based recognition system*

3.4 Infrared

Hand tracking and gestures recognition using infrared is another trend in human computer interaction technology. They use three dimensional capturing devices with infrared sensors called time-of-flight depth aware cameras (Fig. 3.4). They work like a radar and measure the distance between each pixel from the camera and the object. They are very robust and can recognize complex gestures in 3d [23] [24]. To enable the infrared camera clearly to distinguish between points of interest and surrounding it requires the subjects, either to wear LEDs or some other optical markers (like Wii remote [22]), or to avoid wearing them (depending on the recognition approach). Otherwise the method is unstable. It also requires enough distance between the object and the sensors (so that the object can fit in the field of view). Both requirements cannot be met in our testing conditions of trying to recognize gestures with small pointing finger movements in car. Further issues are the low resolution of the depth aware cameras and their high costs.

Figure 3.4: *Commercial product of infrared time-of-flight depth aware camera produced by Softkinetic*

3.5 Combinations

There are also numerous of applications that combine the hardware techniques mentioned above by using for example infrared in combination with normal cameras (Fig. 3.5) [25]. Other studies suggest combinations like face and hand gesture recognition to improve the human machine interaction [26]. So there are numerous ways to improve the current technologies or to combine them in order to reduce their liabilities and to increase their benefits. This however increases their installation costs.

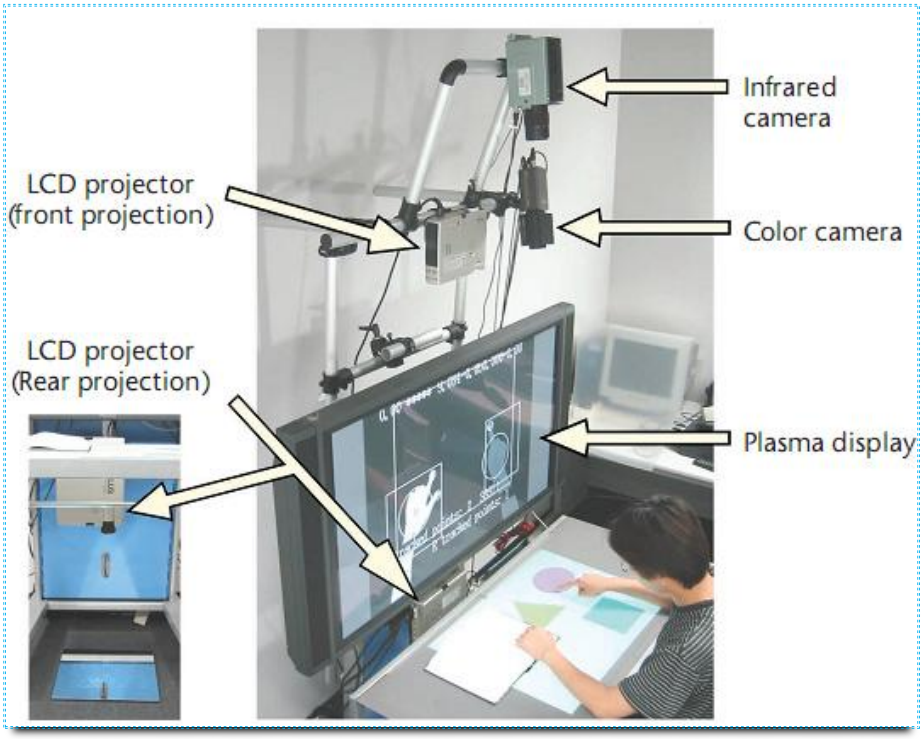

Figure 3.5: *Example of infrared and color camera combination used for finger tip tracking and gesture recognition system*

3.6 Electric Fields

As concluded by a recent comprehensive overview article about the application of the current gesture recognition systems in a car for human vehicle interaction “no publication on a working system has been identified”, although there are different technologies, like lasers and capacitive infrared technologies, that are supposed to cover the automotive field [27].

Electric fields sensing techniques are firstly introduced in 1995 [28]. Their benefits for human vehicle communication over the discussed techniques include; fast response times, line-of-sight- and touch-free, low power consumption and they are safe to use and remain unaffected by light and dynamic backgrounds [29]. Exactly this will be investigated in this work by trying to use electric fields for finger gesture recognition.

3.7 Automotive Design

In the automotive design there is a rising need of new generation of human machine interaction technology. The increasing number of devices makes the designer’s task to add known controls impossible without harming the driver’s safety. The current solutions are just improvement of the older ones like replacing buttons with joysticks or scrolls. Their improvement is limited, since they don’t introduce something qualitative new. In consequence, the interior of the modern cars is overwhelmed by different controllers and every of them leads to drivers distraction (Fig. 3.6).

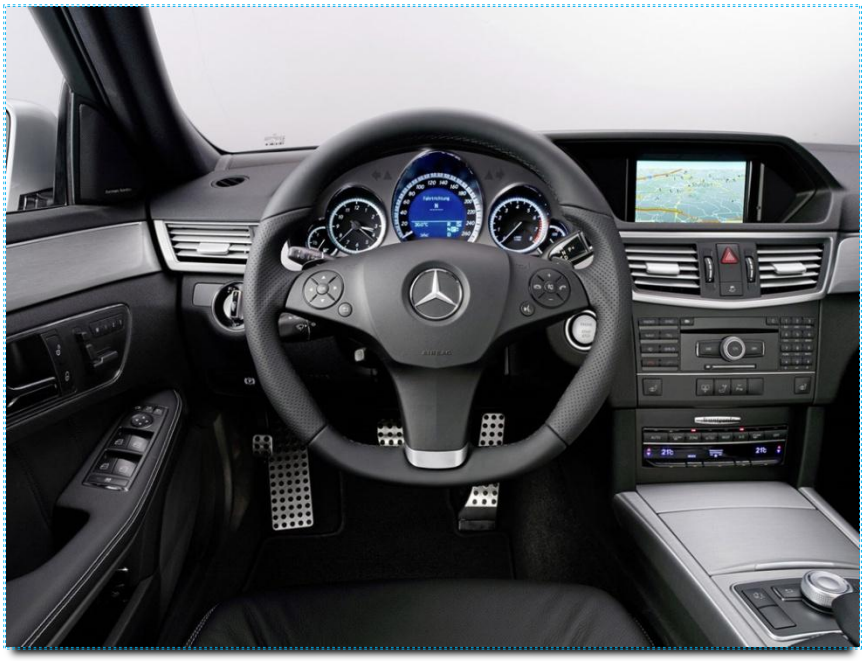

Figure 3.6: *Drivers view inside Mercedes E class shows the potential of drivers distraction*

There is high research interest to improve the interaction by numerous ways. The two most modern techniques being subject of intense research are the speech and gesture recognition. Speech recognition has the big disadvantage by being seriously affected by background music or noise, or several passengers talking at same time to each other. Also the current gesture recognition methods, as shown in this chapter, cannot provide context specific solutions. So the research is looking for new techniques and solutions.

Chapter 4

Geremin Approach

This chapter describes the Geremin approach for touch-free gesture recognition. Geremin stays for “Gesture Recognition using Theremin”. The chapter starts with the motivation and the objectives, which were set in the beginning, followed by the design and the implementation. After this, is provided a detailed information about the technical set-up. The last two sections explore the experimental environment and evaluate the approach.

4.1 Motivation

Creating a new consistent and reliable recognition system is not the only motivation for its design. The users and manufacturers are tending to avoid installing new products unless being convinced of its benefits. In this case, the easy human machine interaction, which gesture recognition provides, increases the most important factor in the automotive field, namely the driver’s safety [1]. Further motivation is to reduce the installation costs of the new system. This work shows that both these factors are elaborated at highest level. Moreover, gestures are expected to have numerous automotive applications until 2020 [1], although recent studies concluded that gestures are not self revealing and therefore need explanation and visual reminders, which the researchers acknowledge to reduce safety.

4.2 Design

There are different attempts to select a gestures set for auto installation. They revealed that mapping all controls in vehicle to gestures is practically impossible, as there are not that many natural gestures. According to researchers, the most promising way in the practice are the gestures drawn with the index finger like circles or lines that map to controls like sound volume or accepting incoming calls [30] [31].

Another important decision is the placement of the sensors, which could vary in places like the windshield areas or central stack area. This approach allows placement to the immediate proximity of the steering wheel, providing gesture recognition without requiring driver to take off hands.

After the designing phase and the study of the existing technologies and commercial products, the main goal of this work was to write software that analyzes the sound output of several theremins and to test if these devices can be used together to recognize gestures and to visualize this in two-dimensional grid. The project was opposed by different problems and limitations like the one that two antennas are disturbing each other and there must be special prepared circuit to avoid this. In consequence was decided to try with one theremin, which turned out to work better than the expectations. Hence, this work also answers the interesting question how much the recognition of linear two dimensional gestures suffers from having only one-dimensional input signals.

4.3 Implementation

Writing software for analyzing the theremin sounds was the main part of this work. On the theremin side, only the antenna responsible for the pitch was used, because of the low precision of the sound input amplitude encoding (not enough bit depth). So after choosing to connect the theremin analog sound input with the line-in of the sound card it

remained to decide in which platform and programming language to write the software. The main choice was between Java and C#, with leaving the platform selection at second place. Due to the promising audio related functions of the DirectX library for .NET it was decided to write the code using Visual Studio 2010 and C# programming language [32]. It was also important task to visualize the results and build an intuitively useable demonstrator. An overview of the whole developed process is shown below in Figure 4.1.

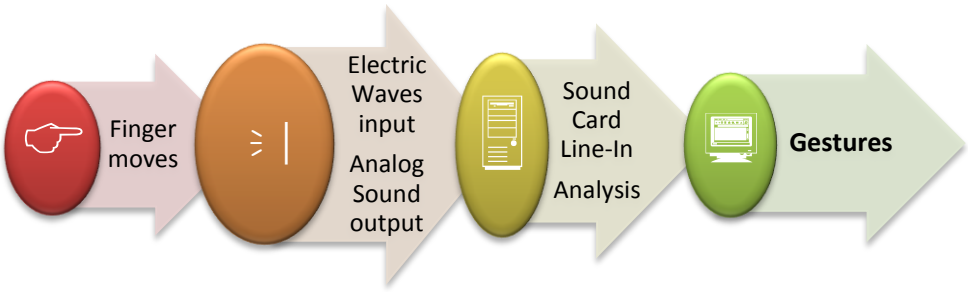

Figure 4.1: *Graphic overview of the main steps from moving the finger to displaying its recognized moves*

After connecting the theremin with the sound card line-in it is important to adjust its amplitude using the volume properties to avoid problems with strong signal input. The sound is captured as single channel (mono) in its pure WAVE form. This is realized using the DirectX CaptureBuffer class. Having this done, the distance between peaks is measured using simple algorithm that runs once through the captured buffer. The exact distance is calculated by using a peak variable indicating the maximum sound level the device can produce. In details, if we have 8000 Hz sample rate (8000 samples per second) and array distance between two positive peaks of 80, gives us 100 Hz current frequency.

The retrieved frequencies are stored for each gesture. Then the application opens a connection to a Java application, which uses the Weka collection of machine learning algorithms for a classification of the gathered dataset [33]. Since the connection is established via TCP, the classifier could as well run on another machine. The experiments were performed on local host with playing the both codes at same time. Further information about the tests and the gesture recognition is provided in the last sections of this chapter. Detailed code structure, usage and architecture can be found in the next section.

Besides real time gesture recognition, there are few many software capabilities, which were written during the implementation phase, like extracting frequency from already recorded WAVE files or logging the session, which aren't discussed here, because they are not relevant to the real time system. They remain included in the user interface and give the opportunity to the user to evaluate already recorded tests or store his current tests.

The next subsections describes in details the reuse and development of the software, which was written to record and evaluate the theremin signal. To use and interact with the program in another applications there are three general ways. The first one is to use it as project for Visual Studio 2010. The second is to use its classes and functions from the dynamic link library. The third is to connect a classifier with the software via TCP at runtime.

4.3.1 Source Code Architecture and Usage

The code is written using Visual Studio 2010 with .NET 3.5 libraries in C# programming language. The architecture consists of four main classes and is written with respect to the Model View Controller design pattern. The most important and responsible for the theremin input analysis is the Recorder class. Then is the Gesture class to define what information to store for one gesture. The Server class is responsible for the TCP communication and the Mainform for the user interface. Example for

important Mainform event is the FrequencyUpdateTimer, responsible for the occurrence of all of the regular operations like asking the Recorder for the current frequency and storing it to the corresponding Gesture member (Figure 4.2 shows example class diagram). Other example for notable function is the ComputeFrequency from Recorder class. It computes the frequency of given array and information about the channels number and bytes per sample (Fig. 4.4).

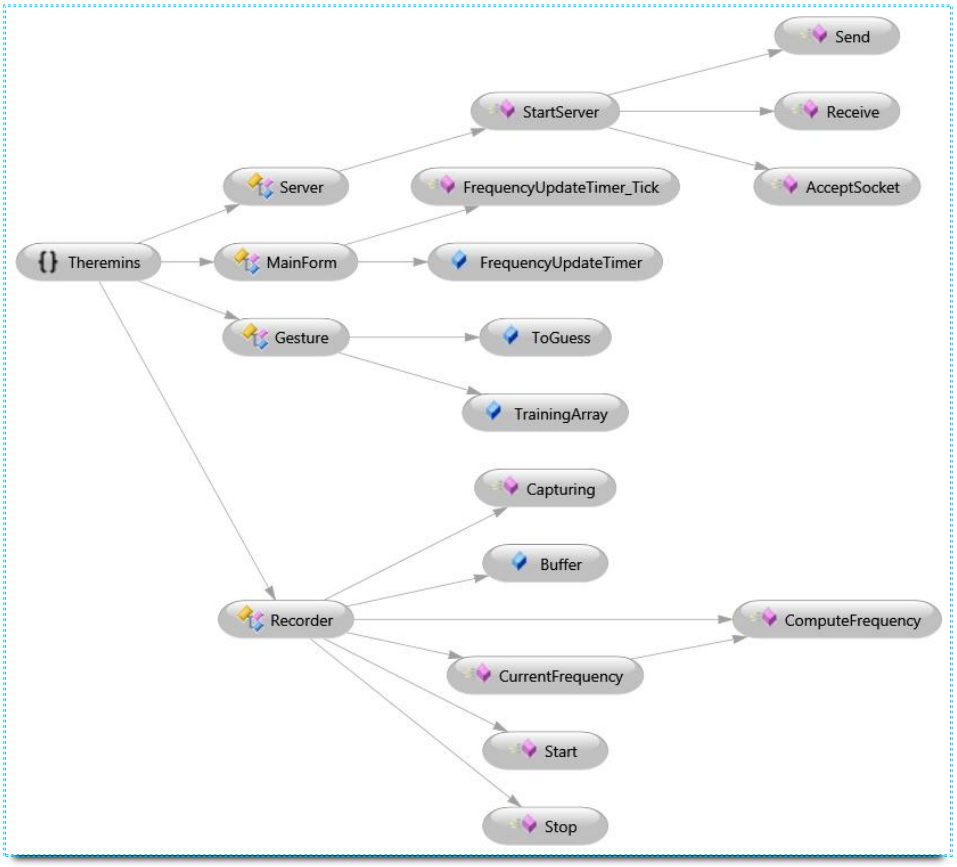

Figure 4.2: *Partial class diagram of the Visual Studio solution with some examples of the most important functions and members from all of the classes*

The software solution starts with “Theremins.sln”. Note that VS 2010 may ask for retargeting the project to .NET 4.0 as default operation (Fig. 4.3).

The user should avoid this, because the code uses DirectxSound.dll, which is still not implemented in 4.0, and use .NET 3.5 instead (may have been checked earlier to retarget without asking again, which will lead to error message that the DirectxSound.dll cannot be found).

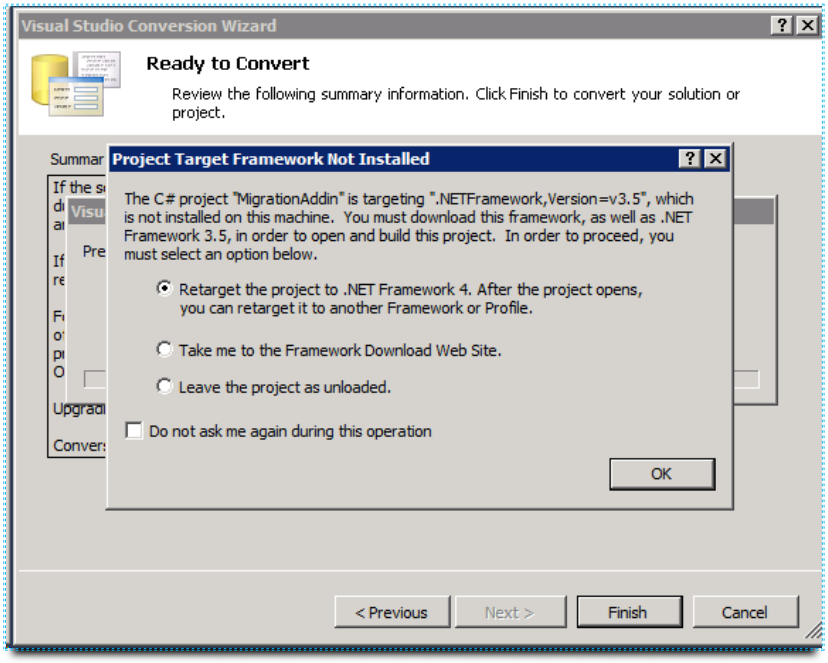

Figure 4.3: *The visual studio conversion proposition dialog*

During the project rose different problems and ideas. All of them affected the code and its functionality. So in the process of writing the source, its reusability was placed first (see code example in Figure 4.4). That means that the code is:

- Rich commented – including links to literature or code sources.
- Very Flexible – adding more theremins (e.g. channels) would not produce new programming job. Just the editing of channels parameters and the labels displaying this on the GUI would be enough. Changing the theremins is also easy to handle just by calibrating the peak levels.
- Easy to understand – using one and the same naming convention and indenting with understandable variable names.

```
public int[] ComputeFrequency(byte[] samples, bool stereo, bool bit16)
{
    int stereoDiv = stereo ? 2 : 1;
    int bitDiv = bit16 ? 2 : 1;

    int sampleIterator;                 // stop iterating when second peak on each channel has been found
    int[] firstPeaks = { 0, 0 };        // the first peak positions in the sample array
    bool[] cleanPeaks = { false, false };// check whether we started the sample in the middle of some peak
    int[] returnFrequency = { 0, 0 };

    for (sampleIterator = 0; sampleIterator < SAMPLE_SIZE; sampleIterator++)
    {
        byte[] sampleData = { samples[stereoDiv * bitDiv * sampleIterator], 0 };

        if (stereo)
        {
            sampleData[SECOND_CHANNEL] = samples[stereoDiv * bitDiv * sampleIterator + bitDiv];
        }

        for (int channel = 0; channel < stereoDiv; channel++)
        {
            // works for both int and byte because we want to check out we passed the middle
            if (sampleData[channel] < 150 && sampleData[channel] > 20)
            {
                cleanPeaks[channel] = true;
            }

            if (sampleData[channel] >= PEAK_LEVEL_8 && cleanPeaks[channel])
            {
                if (firstPeaks[channel] == 0)
                {
                    cleanPeaks[channel] = false;
                    firstPeaks[channel] = sampleIterator;
                    // make assumption about the frequency that is not lower than the sampling / 2000
                    sampleIterator += SAMPLING_RATE / 2000 * stereoDiv * bitDiv;
                }
                else // we found second peak, which is at position [k]
                {
                    if (returnFrequency[channel] == 0)
                    {
                        // pitch as the sample frequency divided by the distance between two peaks
                        returnFrequency[channel] = SAMPLING_RATE / (sampleIterator - firstPeaks[channel]);
                    }

                    if (returnFrequency[FIRST_CHANNEL] > 0 && returnFrequency[SECOND_CHANNEL] > 0)
                    {
                        return returnFrequency;
                    }
                }
            }
        }
    }
    return returnFrequency;
}
```

Figure 4.4: *Code example of the function that computers frequency from the sample array displaying how the code is written. The meaning of the instructions is self explaining and backed up by few comments. Changing the sampling rate or the input channels are handled intuitive and also extending the function can be done without struggling to change the whole structure consisting of naming convention or code style*

4.3.2 Dynamic Link Library Usage

All the functionality of the recorder is nested in “Recorder.cs”. Is has been compiled to “Recorder.dll”, which can be used in other projects. The same holds for the TCP and the Gestures classes, which are in “Server.cs” / “Server.dll” and “Gesture.cs” / “Gesture.dll”.

To add and use these classes in your Visual Studio 2010 project go to the solution explorer, right click to references and add new one. Browse to the location of the desired library (Fig. 4.5).

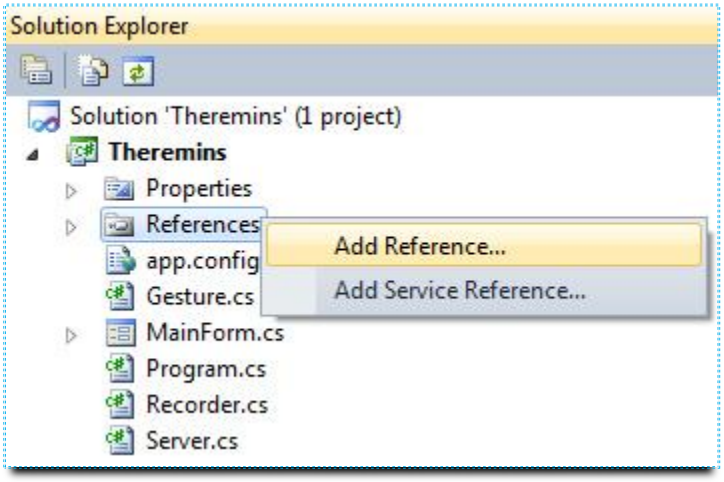

Figure 4.5: *Adding reference to a project in Visual Studio 2010*

For different windows applications please lookout in internet using the keywords “import COM DLL”.

4.3.3 TCP Communication

The TCP server starts with starting the theremin listening state of program. The connection between the server (main program) and the client (classifier) initializes with the keyword START and expects OK answer. Then the server initiates training and sends the training data to the client, which responds after successful record. The training information consists of gesture name, time needed to perform it, and

different frequencies logged in between. Afterwards with the RECOGNIZE keyword together with duration and pitch list the software can call the classifier in order to receive response containing the recognized gesture. Detailed view of the connection protocol is shown in Figure 4.6.

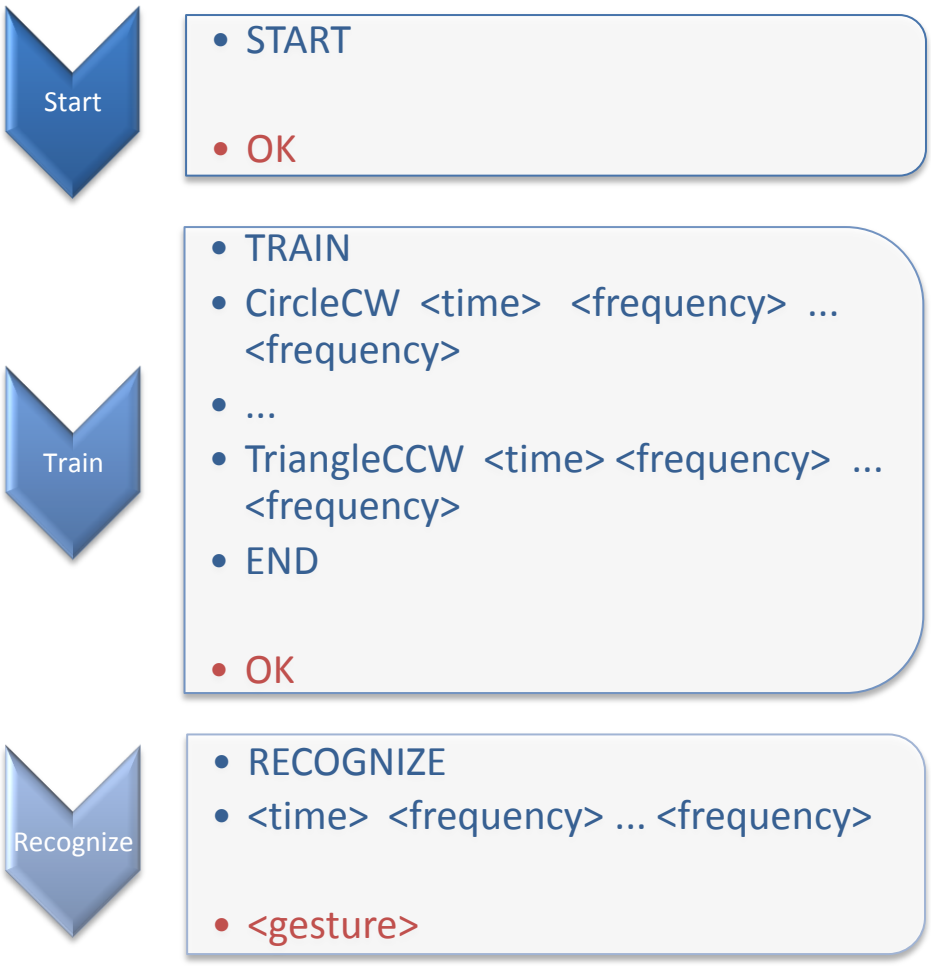

Figure 4.6: *Detailed view of the communication protocol between the program (server in blue font) and the classifier (client in red font)*

4.4 Hardware Set-up

During the work were used three theremins and there was no need of significant calibration when switching them. They all produce a simple sound in form of single frequency tone with clearly readable peaks (Fig. 4.7), which means they are not made of combined sounds.

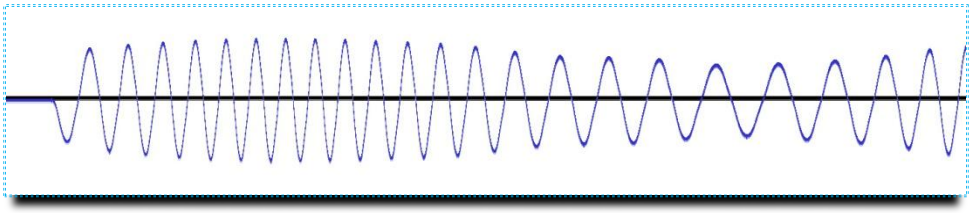

Figure 4.7: *Picture of the sound input. Shows half a second after starting the theremin. The signal gets wider, which means the finger of the subject is moving away from the antenna*

If the theremin has 100 Hz minimal frequency this would mean 50 Hz refresh rate, because in the worst case frame listening will start right after some peak and will need two more to measure the distance between them. This doubles the normal cameras frame rate which is around 25 frames per second. Of course, by calibrating the theremin with different minimum frequency like 500 Hz leads to 250 Hz (4 ms) refresh rate (which is 10 times better than the normal cameras). Recording with the smallest sample rate of 8000 is sufficient, because the exact quality of the sound is unimportant, where even low end sound card and theremin would work. The same goes for the byte rate, where we need only one byte per channel (theremin). So with one theremin we have a data flow of around 8000 bytes per second. The algorithm of finding the frequency of the sound simply parses the frames linearly, so the hardware requirements for this approach can be satisfied by the most of the embedded processors in cars now. It is important to mention again, that the gesture recognition systems are not supposed to occupy the whole computational power and energy of the car's computer.

Figure 4.8: *The first tested theremin bought from eBay as the cheapest offered there worked out well. The figure shows how the output signal and the audio cable are connected with crocodile clips. On the theremin side, there is only single channel output, and on the soundcard line-in side, only one channel is used from two (second one remains free and can be used to connect one more theremin). The audio cable in the picture has 3.5mm stereo jack plugs at his both sides. The image displays also the simplicity of the theremin hardware*

The pitch extraction worked out well with all three tested theremins (from cheapest "made in china" theremin from eBay shown in Figure 4.8 to self produced theremin circuit). It consists of measuring the distance between the peaks by simple single reading of the current sound buffer (C# code shown in Figure 4.4). The final tests of this work are done using a Jupiter 4 (Fig. 4.9).

Figure 4.9: *Jupiter 4 theremin. The left antenna is for the volume and the right, which was used, is for the frequency. The three scrolls below are to calibrate the pitch height and sensitivity, and the volume of the output*

As the waves are propagating quadratic it will normally make sense to filter the incoming frequencies with the inverse formula to calculate the precise hand distance from the sensor [34] [35]. Otherwise the classifier will have problems recognizing data input with distance of 30 cm between the finger and the antenna with having training data with 10 cm distance, because the first signal will look more like linear function and the second more like quadratic function, so comparing them wouldn't make any sense. In our particular case such filter is not so important because training and gesture recognition start always at the same frequency (position) and the sensitivity of the theremin is calibrated knowing this issue. However it will be important to study the exact behavior, when adding more antennas, to make visualization of the hand or finger movement on the screen, like in two dimensional grid with cursor.

4.5 Gesture Recognition

For the gesture recognition is used a set of 10 gestures (Fig. 4.10). This set has been selected in order to have simple gestures like upward and downward movements, together with some complex gestures, in geometrical meaning, like square or circle with respecting the drawing direction (clockwise or counterclockwise). Similar sets have been used in the related works, already discussed in the last chapter. The gesture recognition can be performed after single training session, where the user trains every gesture once. The distance between the index finder and the theremin is around 10 cm. This reveals another advantage of this recognition approach, its field of view, which is at least four times greater than the normal cameras. This allows installation of the recognizer to the immediate proximity of the steering wheel and hence avoids different possible disturbance factors, like moving objects close to the sensor.

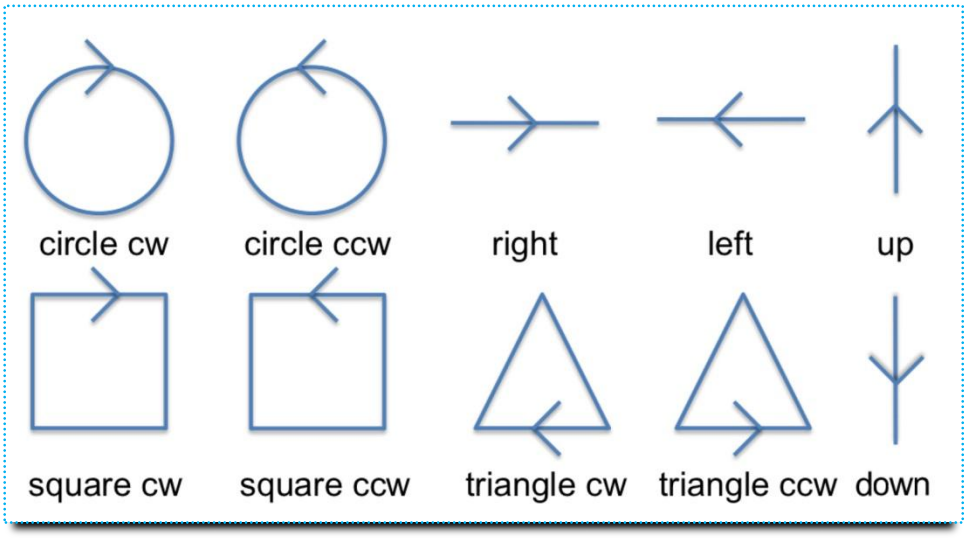

Figure 4.10: *The gesture set of simple straight lines and more complex clockwise and counterclockwise geometric figures*

While the subject is moving his finger (Fig. 4.11), during the training of some gesture or attempting to recognize, the program computes every

millisecond the frequencies, which the theremin outputs. Note that the frequency doesn't change with that rate, so only different consecutively frequencies are processed. This doesn't harm the process because even without moving the finger there is also some slightly difference of the frequency. After the training phase or the recognition phase the program connects with the classifier, sends him the training data or the unknown data, and accepts response regarding of the result (Fig. 4.12).

Figure 4.11: *The experimental phase represents a typical car situation of holding the steering wheel with left hand and making finger movements, which are to be recognized by the system*

At first place the duration time of the gesture is sent, since the time to perform some gesture also indicates the complexity of the gesture (drawing square is about five times slower than just moving the finger to the right). However, this information is not evaluated by the classifier at the moment. Then the recorded frequencies during the training or

recognizing are being sent. After this the recognizer says he has been trained or gives as output the recognized gesture together with the possibility of any other gesture (Fig. 4.6).

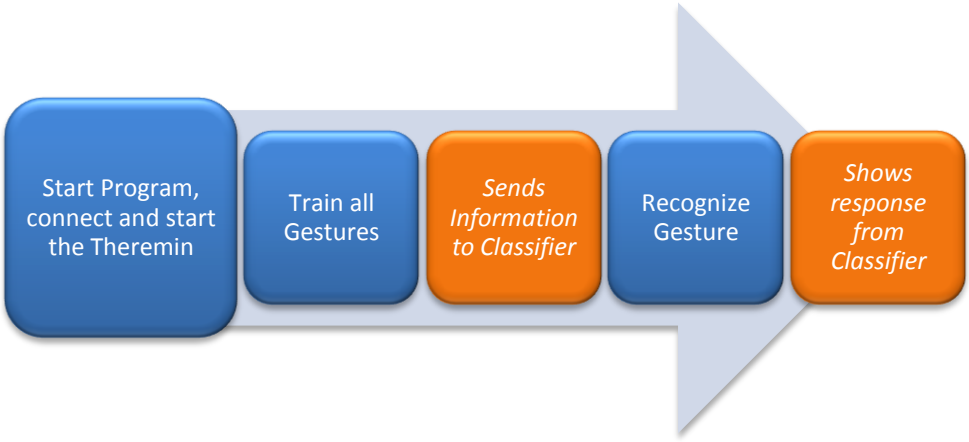

Figure 4.12: *Represents the process of recognizing a gesture by our current system*

From the test user perspective there is button to start device listening (real-time theremin sound output capturing). After successful start the user can initialize gesture training and afterwards start the recognizer (Fig. 4.13). For each state there are corresponding command buttons and the actions are supposed to be done in consequential order. Also the buttons remain not enabled until the previous actions are not done. As an example, after starting the device listening mode the frequency is displayed under the button, and the training state will be enabled (both buttons that indicate training mode and start and stop time for each gesture, which also can be done convenient by holding a shortcut button). After training all gestures, the information is transmitted to the classifier automatically or by clicking the transmit button (if the connection doesn't persists for the automatic data transmission). Afterwards the Recognize button becomes enabled to enter the recognition mode and gestures can be recognized (in the same way by clicking start and stop button or by using shortcut). The recognition

happens immediately after the recognize event (the program connects with the classifier and displays the result). The picture panel displays, which gesture should be trained now, or which gesture is recognized.

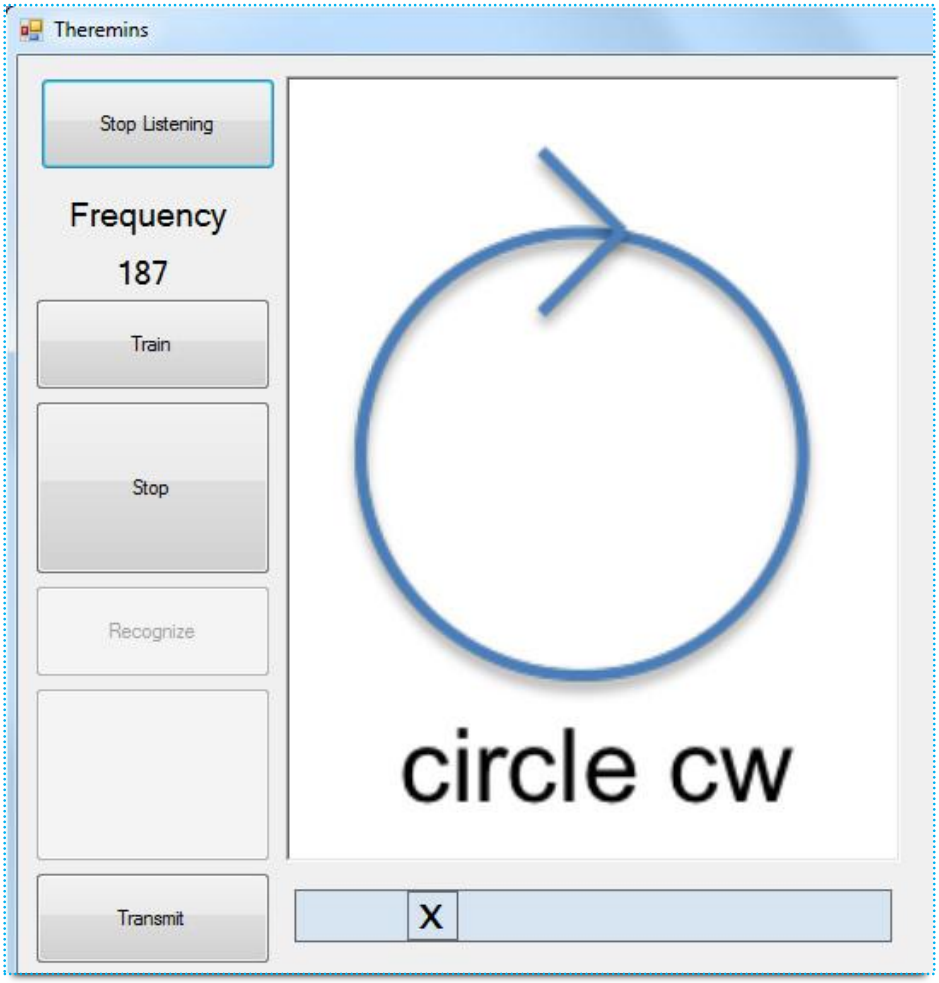

Figure 4.13: *Partial screenshot while the software is running. At the top left there is button to stop the theremin device listening. Then it shows also that we are currently in training mode and we are training the circle gesture. As we are in training mode the recognition phase buttons are still not enabled. The small cross under the picture panel shows the hand distance from the antenna*

4.6 Evaluation

After the implementation and set-up phases, which shown the program to run correctly, three small tests were performed with one subject to check its recognition accuracy. Each test is done with one training session and ten attempts to recognize each gesture. Firstly, the gestures were done relatively slow, then faster and then with maximum precision that the subject could obtain. All gestures were performed with the left index finger. Important note is that no parameter tuning or adaption of the classification algorithm was made. The results are shown in Figure 4.14.

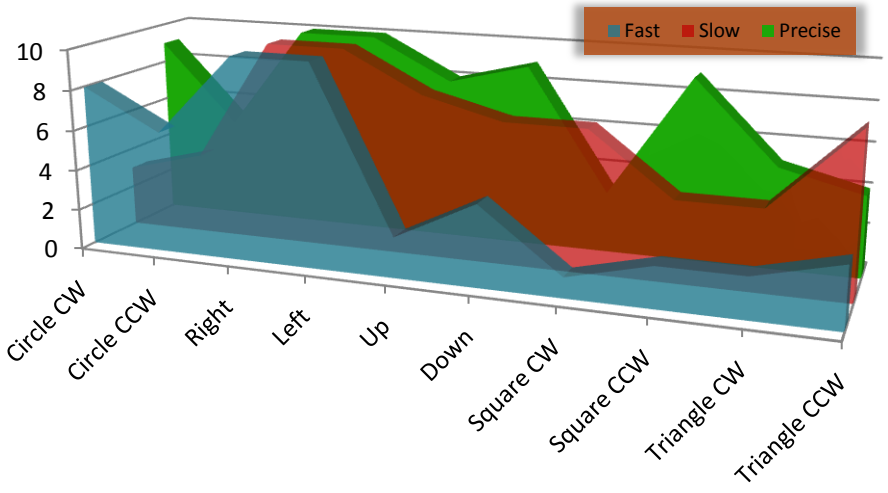

Figure 4.14: *The results of the three performed tests with left pointing finger. On the Y axis is the number of correct recognitions for the different gestures in X axis*

In the fast experiment there is big difference in recognition result between the circle and the rest of the complex gestures (triangle and square). The slow experiment showed similar differentiation, but this time between the triangle and the rest of the complex gestures. By looking at the detailed output of the classifier we can see, that there are difficulties in distinguishing the complex shapes, because they all were assigned with almost the same recognition chance. This is caused also by

the difficulty level of drawing these shapes in the air. So the recognizing algorithm “favors” one of them and outputs it (Fig. 4.15).

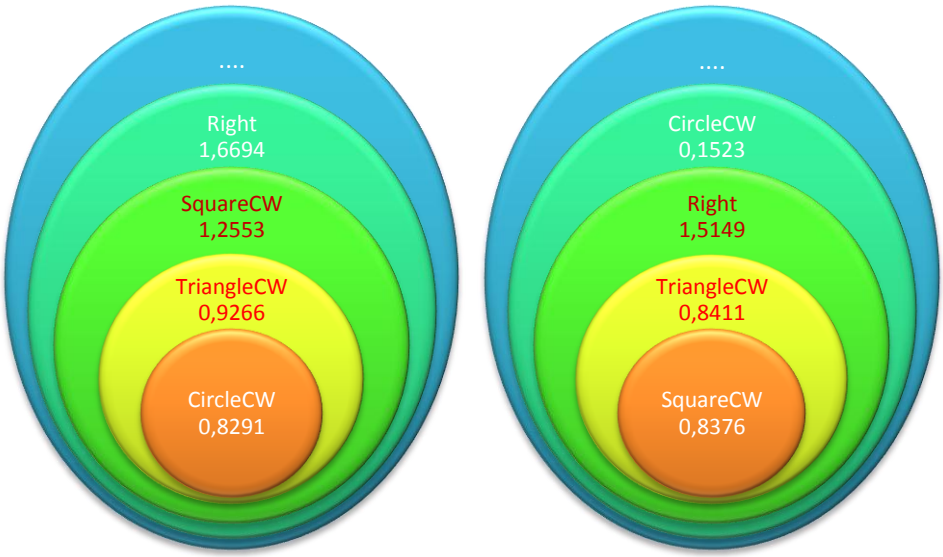

Figure 4.15: *Displays two different outputs of the classifier after training. After dimension reduction, the algorithm assigns a real number to every shape regarding of the received training data. For recognition the same process is done and then the closest number is chosen for output. This demonstrates how closely and difficult is to predict some of the shapes*

At this place, we can see the need of replacement of some gestures from the set, like adding slope lines and removing of some of the complex gestures. In recognizing the direction of these shapes however, the recognizer was robust. We can see the perfect result when moving the finger left and right. This results from the change of frequency by the vertical placed antenna. One more interesting thing is that up and down gestures are recognized in two of the tests very good but in the third one very poor. This can be explained by the fact that moving the finger up and down isn’t that perfect and it has some slope and angle. By the fast

performed gestures this slope wasn't that clear and in the training the gestures may have been done pretty good so the recognizer had trouble finding out what happens. In the precise test we can see slightly improvement compared to the slow test. By counting all of the successful recognitions from the 10 tries with the 10 gestures set we can achieve briefly percentage rate of the accuracy. It is 48% for the fast test, 65% for the slow and 72% for the precise. There was also a try to do precise test with mobile phone between subject's finger and the antenna. As a result of that the recognizer could not recognize even his best case of moving left and right. Here antenna frequency tuning will be important to avoid such disturbance. There are also some difficulties, when working with larger sensitivity and when other objects are moving near the antenna (not necessarily in close distance or between the finger and the sensor). In the case of in-car installation, this would cause no troubles as no one besides the driver should use the steering wheel and his place. The exact result output is shown in Table 4.1.

	Circle CW	Circle CCW	Right	Left	Up	Down	Square CW	Square CCW	Triangle CW	Triangle CCW
Slow	3	4	10	10	8	7	7	4	4	8
Fast	8	6	10	10	2	4	1	2	2	3
Precise	9	5	10	10	8	9	3	9	5	4

Table 4.1: *Three tests of slow, fast and precise finger movement. The numerical data represents the number of correct recognitions made by the system from 10 total tries*

Chapter 5

Conclusion & Outlook

This work provided finger gesture recognition, using a theremin musical instrument connected to a sound card of a computer, the so called Geremin Approach. This allows in-car gesture recognizer embedding, without expensive installation costs or complicated and computationally exhaustive algorithms. The developed software showed flexibility by successfully running with three different theremins. The implementation of real-time signal processing system was excellent in terms of running time and robustness. The recognition accuracy was also satisfying and showed great results when detecting the directions of finger movements. The recognized finger gestures are two-dimensional, despite the one-dimensional input information, which reveals promising growth of this concept. All other minor goals have been achieved; also many underlying problems and aspects were enlightened during the whole development process and later evaluation.

There are numerous of improvements, which the future work on the subject should consider. One of them is to create frequency filter for the input, important to bypass the theremin liability of being often disturbed by any kind of magnetic waves around, also by moving objects. Further improvement is modifying the hardware by creating a circuit with more antennas and outputting regular numbers about the object distance instead of sound. The tests showed also the need to calibrate the sensitivity and the frequency of the air waves component to achieve maximum accuracy rate and low disturbance by side effects. Moreover, state-of-the-art classifying algorithms can be used and calibrated for this

special case in order to improve the recognition success percent. Using a bigger set of training data is also essential for the recognizer to generalize sufficiently. A task for the designers is to find the optimum gesture set for the special purpose of in car installation and respecting the hardware capabilities. Finally there is a need of comprehensive tests with larger number of different subjects and also to study the efficiency under noisy conditions in terms of presence of other electrical devices.

In the near future, recognizers based on electric fields may appear in different domains of our surrounding world, whether as an addition to existing systems or stand-alone, because of the growing information environment, in which we are living our daily life. Research trends go toward making the devices around us more intuitive and smart. However, giving a big computational power to the embedded systems is neither an easy nor a cheap task for the engineers. Hence, exploiting simple existing gadgets or designing less complex circuits will yield a solution for optimizing the installation costs or a way to solve different upcoming problems.

References

1. C. Pickering. “Gesture Recognition Could Improve Automotive Safety”, *Asian Engineer – Automotive-Design, October 2006.*
2. L. Shao, C. Shan, J. Luo, M. Etoh. “Multimedia Interaction and Intelligent User Interfaces”, *1st Edition, Springer, 2010, pp. 107-128.*
3. A. Glinsky. “Theremin: Ether Music and Espionage”, *University of Illinois Press, 2000.*
4. Microsoft Corporation. “Multiple Channel Audio Data and WAVE Files”, *December 2001*.
5. IBM Corporation and Microsoft Corporation. “Multimedia Programming Interface and Data Specifications 1.0”, *August 1991, pp. 56-65*.
6. B. Myers. "A Brief History of Human Computer Interaction Technology.", *ACM interactions. Vol. 5, no. 2, March, 1998, pp. 44-54.*
7. C. Fang. “From Dynamic Time Warping (DTW) to Hidden Markov Model (HMM)”, *Final project report for ECE742 Stochastic Decision, March 2009.*
8. H. Sakoe, S. Chiba. “Dynamic programming algorithm optimization for spoken word recognition”, *Transactions on Acoustics, Speech and Signal Processing, 1978, pp. 43- 49.*
9. R. Nesselrath, J. Alexandersson. „A 3D Gesture Recognition System for Multimodal Dialog Systems“, *6th IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems, July 2009, pp 46-51.*
10. H. Petrosky. “How US car designers came to include cup holders”, *Slate Magazine, March 2004.*
11. D. Norman. “Emotional Design: Why We Love (or Hate) Everyday Things”, *Basic Books, December 2003.*

12. I. Sutherland. "SketchPad: A Man-Machine Graphical Communication System", *AFIPS Spring Joint Computer Conference. 1963. 23. pp. 329-346.*
13. R. Neßelrath. „TaKG - Ein Toolkit zur automatischen Klassifikation von Gesten“, *Master Thesis at University of Saarland, March 2008, pp. 27-34.*
14. T. Schlömer, B. Poppinga, N. Henze, S. Boll. “Gesture Recognition with a Wii Controller”, *Proceedings of the 2nd International Conference on Tangible and Embedded Interaction*, *February 2008, pp. 11-14.*
15. S. Nasiri, S. Lin, D. Sachs, J. Jiang. „Motion Processing: The Next Breakthrough Function in Handsets“, *InvenSense Inc., July 2009.*
16. Y. Wu and T. Huang. "Vision-Based Gesture Recognition: A Review", *Gesture-Based Communication in Human-Computer Interaction, Springer, 1999, pp. 103-116.*
17. R. Cipolla and A. Pentland. “Computer Vision for Human-Machine Interaction”, *Cambridge University Press, 1998.*
18. V. Pavlovic, R. Sharma, T. Huang. "Visual interpretation of hand gestures for human-computer interaction: A review", *IEEE Trans. Pattern Analysis and Machine Intelligence, July 1997, pp. 677 – 695.*
19. Department of Computer Engineering Boğaziçi University. “Hand Gesture Interface for Windows Operating Systems”, *2006.*
20. M. Kölsch, M. Turk. “Fast 2D Hand Tracking with Flocks of Features and Multi-Cue Integration*”, IEEE Workshop on Real-Time Vision for Human-Computer Interaction, 2004, pp. 158-166.*
21. S. Lovell. “A System of Real-Time Gesture Recognition and Classification of Coordinated Motion”, *Bachelor and Master Thesis, MIT, February 2005*
22. G. Hackenberg. Master Thesis, *Fraunhofer FIT, July 2010.*
23. J. Lee. “Wiimote Project”, *www.wiimoteproject.com.*
24. P. Breuer, C. Eckes, S. Müller. “Hand Gesture Recognition with a novel IR Time-of-Flight Range Camera – A pilot study”, *Proceedings of the 3rd international conference on Computer vision/computer graphics collaboration techniques, 2007, pp. 247-260.*

25. K. Oka, Y. Sato, H. Koike. “Real-Time Fingertip Tracking and Gesture Recognition”, *IEEE Computer Graphics and Applications, December 2002, pp. 64-71.*
26. M. Yang. “Face Detection and Gesture Recognition for Human-Computer Interaction”, *Springer, 2001.*
27. C. Pickering, K. Burnham. “A Research Study of Hand Gesture Recognition Technologies and Applications for Human Vehicle Interaction”, *third Institution of Engineering and Technology Conference on Automotive Electronics, June 2007, pp. 1-15.*
28. T. Zimmerman, J. R. Smith, J. A. Paradiso, D. Allport, N. Gerschenfeld. „Applying Electric Field Sensing to Human-Computer Interfaces”, *Proceedings of CHI, 1995, pp. 280-287.*
29. C. Pickering. “Human Vehicle Interaction Based On Electric Field Sensing”, *Advanced Microsystems for Automotive Applications, 2008, pp. 141-154.*
30. U. Reissner. “Gestures and Speech in Cars”, *Electronic Proceedings of Joint Advanced Student School, March 2007.*
31. K. Whitfield. “Gesture Interface for Automotive Control: Beyond Digital Expletives”, *7th Journal f*or *Automotive Manufacturing and Production, July 2003, pp. 50-56.*
32. Microsoft Corporation. “Visual Studio 2010”, *www.microsoft.com/visualstudio.*
33. M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. Witten “The WEKA Data Mining Software: An Update”, *SIGKDD Explorations Volume 11, 2009.*
34. A. Koga. “Study on a Theremin”, *www.thekoga.com, December 2006.*
35. F. Nachbaur. “On Theremin Sensitivity”, w*ww.dogstar.dantimax.dk, September 1997.*