CHALMERS

SoundCRAB

Automatisering av hörselbaserade tester

Examensarbete inom Data- och Informationsteknik

Samuel Leckborn
Karl Svensson

Institutionen för Data- och Informationsteknik
CHALMERS TEKNISKA HÖGSKOLA
GÖTEBORGS UNIVERSITET
Göteborg, Sverige 2021

Examensarbete 2021

SoundCRAB

Automatisering av hörselbaserade tester

Samuel Leckborn
Karl Svensson

CHALMERS

Institutionen för Data och Informationsteknik
Chalmers Tekniska Högskola
Göteborgs Universitet
Göteborg, Sverige 2021

SoundCRAB

Automatisering av hörselbaserade tester

Samuel Leckborn
Karl Svensson

© Samuel Leckborn, Karl Svensson, 2021.

Examinator: Peter Lundin, Institutionen för Data och Informationsteknik

Institutionen för Data- och Informationsteknik
Chalmers Tekniska Högskola / Göteborgs Universitet
SE-412 96 Göteborg
Telefon +46 31 772 1000

The Author grants to Chalmers University of Technology and University of Gothenburg thenon-exclusive right to publish the Work electronically and in a non-commercial purpose make it accessible on the Internet. The Author warrants that he/she is the author to the Work,and warrants that the Work does not contain text, pictures or other material that violates copyright law.

The Author shall, when transferring the rights of the Work to a third party (for example a publisher or a company), acknowledge the third party about this agreement. If the Author has signed a copyright agreement with a third party regarding the Work, the Author warrants hereby that he/she has obtained any necessary permission from this third party to let Chalmers University of Technology and University of Gothenburg store the Work electronically and make it accessible on the Internet.

Omslag: Visuell representation av ett ljud som analyseras. Producerad av rapportens författare.

Institutionen för Data- och Informationsteknik
Göteborg, Sverige 2021

Sammanfattning

I och med popularisering av agila arbetsmetoder har efterfrågan på automatiserade tester ökat inom fordonsbranschen. Ett typ av test som vanligtvis sker manuellt i dagsläget är verifikation av återkopplingsljud, alltså ljud som fordonet spelar upp för att bekräfta en interaktion från användaren. För att utforska sätt att automatisera sådana testmoment beskriver denna rapport ett framtaget prototypsystem och utvärderar dess förmåga att ersätta en människa i testprocessen. Genom diskussioner om vad människan lägger fokus på vid testning av ljud sattes delmål upp som guidade dels utvecklingen av systemet, dels tidsplaneringen. System utvecklades i programmeringsspråket Java. För att underlätta utvecklingen integrerades systemets kod med existerande kodbibliotek för hantering av ljudfiler och signalanalys. Systemets olika delar verifierades under arbetets gång genom praktiska experiment i en simulerad testmiljö. Resultatet visar att då systemet har en fördel mot människan när det gäller att med robust precision avgöra objektiva tids- och volym-aspekter hos ljud har människan i stället fördel när det kommer till att identifiera missljud i en ljudbild. För att kunna ersätta en människas hörsel helt i detta sammanhang krävs vidarearbete där känsligare metoder kalibrerade efter människans hörsel utvecklas.

Nyckelord: testautomatisering, ljud, analys.

Abstract

With the popularity increase of of agile methods comes a higher demand for automated tests within the automotive industry. One type of test that normally is done manually currently is verification of auditory feedback in vehicles. Auditory feedback is a sound that is being played to confirm interaction from a user. To explore ways to automate these types of tests this paper describes a developed prototype system and evaluates its ability to replace the human hearing in a test setting. Through discussion about what is important to a human when testing audio a number of goals was defined that guided both development and timeline planning. The system was developed in the program language Java. To help with the development, already existing codelibraries were integrated with the systems code for handling of audio files and signal analysis. The different parts of the system was verified continuously through practical experiments in a simulated testing environment. The result shows that while the developed software has an advantage when it comes to robust and precise identification of time and volume aspects of audio, the human hearing is better at detecting noise. To be able to completely replace the human hearing in this type of scenario more sensitive methods calibrated to the human subjectivity would have to be developed.

Keywords: Audio, Test automation, Sound, Analysis.

Förord

Denna rapport är resultatet av ett examensarbete vid Institutionen för data- och informationsteknik som utförts under vårterminen 2021 på Chalmers Tekniska Högskola. Vi vill först och främst tacka vår handledare på Diadrom Systems AB Henrik Fagrell. Han är grunden till arbetets idé och har ständigt givit oss värdefull feedback under arbetets gång. Vi vill även tacka vår handledare från Chalmers Sakib Sistek som under arbetets gång guidat oss i rapportskrivning och delat med sig av tips och erfarenheter för att arbetet ska flyta på utan problem.

Samuel Leckborn, Karl Svensson, Göteborg, 2021

Ordlista

AD omvandlare Analog till Digital signalomvandlare. 4

ASCII En teckenkodning som används för att representera bokstäver och andra tecken binärt. 5

Java Ett objektorienterat programspråk. 9

JUnit JUnit är ett ramverk för enhetstestning av programmeringsspråket Java. 10

kvantifieras En process för att approximera en kontinuerlig värdemängd till en relativt liten målmängd. 4

MATLAB Ett datorprogram och programspråk som kan användas vid matematiska och tekniska beräkningar. xv, 9, 18

oscillera Rörelse kring ett jämnviktsläge. 3

Förkortningar

API Application Programming Interface. 9
ASCII American Standard Code for Information Interchange. 5

FFT Fast Fourier transform. 7, 18, 19

RIFF Resource Interchange File Format. xvii, 5
RMS Root Mean Square. 8, 18, 24

UML Unified Modeling Language. 10, 20

Innehåll

Figurer **xvii**

Tabeller **xix**

1 Introduktion **1**
1.1 Bakgrund . . . 1
1.2 Syfte . . . 1
1.3 Mål . . . 1
1.4 Frågeställning . . . 2
1.5 Avgränsningar . . . 2

2 Teknisk bakgrund **3**
2.1 Ljud . . . 3
2.1.1 Människans hörsel . . . 3
2.1.2 Analog representation . . . 4
2.1.3 Digital representation . . . 4
2.1.4 Akustiskt fingeravtryck . . . 5
2.2 Signalanalys . . . 6
2.2.1 Korrelation . . . 6
2.2.2 Frekvensspektrum . . . 7
2.2.3 Volym . . . 8

3 Metod och genomförande **9**
3.1 Verktyg för utveckling . . . 9
3.1.1 Musicg . . . 9
3.1.2 MATLAB . . . 9
3.1.3 Utrustning . . . 9
3.2 Planering . . . 10
3.2.1 Modellen . . . 10
3.3 Utveckling . . . 10
3.4 Testning . . . 10

4 Systembeskrivning **13**
4.1 Systemflöde . . . 13
4.1.1 Exposition . . . 13
4.1.2 Konfiguration . . . 14
4.1.3 Testresultat . . . 14

4.2 Systemmodeller 14
4.2.1 Specifikation 15
4.2.2 Inspelning 15
4.2.3 Analys 16
4.2.4 Testresultat 19
4.3 Systemstruktur 20

5 Resultat **21**
5.1 Ljudidentifikation 21
5.2 Likhetsanalys 22
5.2.1 Missljud 23
5.2.2 Frekvensanalys 24
5.2.3 Ljudets uppmätta volym 24

6 Diskussion **27**
6.1 Frågeställningar 27
6.2 Etiska aspekter 28
6.3 Miljöaspekter 29

7 Slutsats och framtida rekommendationer **31**
7.1 Förslag för vidarearbete 31

A Bilagor **I**
A.1 JUnit klasstester I
A.2 UML - Klassdiagram IV

Figurer

2.1 Kvantisering av en analog signal(grön linje) till digital representation(röd linje) . 4
2.2 Visuell beskrivning av en Wave fils uppbyggnad enligt RIFF specifikationen . . . 5
2.3 En ögonblicksbild av en korrelationssekvens 7
2.4 Vy av en ljudsignal i tid och frekvensdomänen 8

3.1 Mikrofonens placering vid ljudkällan 11

4.1 Specifikation av två alternerande ljud 13
4.2 Den mest övergripande modellen av systemet. 14
4.3 Modell av ett testobjekt 15
4.4 Modell av ett SoundCrab objekt 16
4.5 Genererad ton med frekvenser 400 Hz och 1500 Hz 19
4.6 Frekvenskomponenter från en den genererade tonen med frekvens 400 Hz och 1500 Hz 19
4.7 Text som specificerar testets resultat 20

5.1 Inspelad ljudsignal av ett blinkerstillslag, frekvenskoefficient = 0,97 24
5.2 Editerad inspelning av blinkerstillslag, frekvenskoefficient = 0,75 24

Tabeller

5.1 Resultat av ett identifikationstest med brus 21
5.2 Resultat av identifikationstest med relativ volymskillnad 22
5.3 Resultat av test med olika avstånd till ljudkälla 23
5.4 Resultat av korrelationskoefficient experiment med missljud 23
5.5 Resultat av frekvenskoefficient experiment med missljud 23
5.6 Resultattabell för test att RMS värdet ökar med en högre förstärkning 24
5.7 Resultattabell för test att med högfrekvent bakgrundsljud 25

1
Introduktion

I detta inledande avsnitt av rapporten presenteras grunden till arbetet. Här beskrivs rapportens bakgrund, mål, syfte och frågeställning.

1.1 Bakgrund

Projektet genomförs i samarbete med Diadrom Systems AB som är ett konsult- och produktbolag inom diagnostik av autotech. En utmaning de står inför är att det på senaste tiden blivit mycket populärt med agila metoder. Man utvecklar i korta iterationer för att förbättra produktionen. För att detta skall fungera effektivt måste man i möjligaste mån automatisera flertalet testmoment. En särskilt svår sak att automatisera är tester av användargränssnitt, där manuell testning fortfarande är dominerande. I dagsläget sker tester inom fordonsindustrin som kräver en människas hörsel vid verifiering av, till exempel, återkopplingsljud vid aktivering av blinkers eller andra funktioner i fordonet.

1.2 Syfte

Syftet med rapporten är att utforska möjligheter att ersätta den mänskliga hörseln vid testning av återkopplingsljud. Ljud spelas in med en mikrofon och analyseras på olika sätt av systemet för att avgöra om det låter korrekt eller inte. Systemet förmåga ska utvärderas i relation till sin mänskliga motsvarighet.

1.3 Mål

Målet är att utveckla ett system som automatiserar tester av ljud genom att bedöma ljudets korrekthet. Med korrekthet menas likhet till ett förväntat referensljud med avseende på timing, volym och frekvensomfång.

För att avgöra om ett ljud är korrekt kommer följande delmål att behöva uppnås:

- Integrera metoder för att kunna skapa och editera ljudfiler.
- Integrera en metod för att med hög tillförlitlighet avgöra när ett specifikt ljud börjar och slutar i en inspelning.
- Implementera en metod att mäta tiden mellan två specifika ljud i en ljudfil.

- Avgöra den upplevda volymen med hjälp av rå ljudfilsdata.
- Integrera en metod för att kunna läsa av frekvensinnehåll i en ljudfil.
- Implementera jämförelser mellan inspelning och referens.

Användaren ska ha kontroll över specifikationen systemet använder för att värdera analysdatan och leverera ett testresultat.

1.4 Frågeställning

- Hur väl står sig det framtagna systemet mot människans förmåga att avgöra timing?
- Hur väl står sig det framtagna systemet mot människans förmåga att avgöra volymskillnader?
- Hur väl står sig det framtagna systemet mot människans förmåga att upptäcka missljud?

1.5 Avgränsningar

För att underlätta projektets arbete och utveckling så gjordes följande antaganden och avgränsningar.

- Antalet ljud som systemet testar begränsades till två ljud av olika karakteristik.
- Alla inspelningar sker i en kontrollerad miljö med minimala störningar
- Referenser om ett ljud låter bra eller inte bedömdes endast av rapportens författare.
- Valet av inspelningsutrustning gjordes med begränsad budget.
- Systemet testades endast i hemmamiljö och inte i en verklig testmiljö.

2

Teknisk bakgrund

I följande avsnitt presenteras nödvändig bakgrundsinformation för att läsaren enklare ska förstå resterande avsnitt av rapporten.

2.1 Ljud

Att tala om ljud är att tala om den mänskliga upplevelsen av det fysiska fenomen som är longitudinella vågor som propagerar genom ett medium, till exempel luft. En sådan våg skapas genom att energi tillsätts till ett objekt på ett sådant sätt att det fysiskt börjar oscillera. Detta objekt ger då upphov till en longitudinell våg som skapar variationer i omgivningens lufttryck. Det är dessa variationer människan uppfattar som ljud och det är vågens fysiska egenskaper som avgör tillexempel upplevd styrka och tonhöjd.[1][2]

2.1.1 Människans hörsel

Som ett resultat av människans förmåga att översätta fysiska variationer av lufttryck till upplevda ljud har ett antal koncept introducerats för att lättare kunna analysera och kommunicera denna upplevelse.[2]

Tonhöjd

Beroende på avståndet mellan ljudvågens förtätningar och förtunningar av partiklar i mediet den propagerar över upplever människan att ljudet har olika tonhöjd. Desto tätare variationerna kommer, det vill säga högre frekvens, upplevs också en högre tonhöjd. Tvärtom ger längre avstånd mellan variationerna en upplevelse av låg tonhöjd. En människa uppfattar ljudvågor i frekvensområdet 20 Hz-20 kHz. Detta frekvensområdet är dock väldigt individberoende och blir också mindre med åldern. [1]

Volym

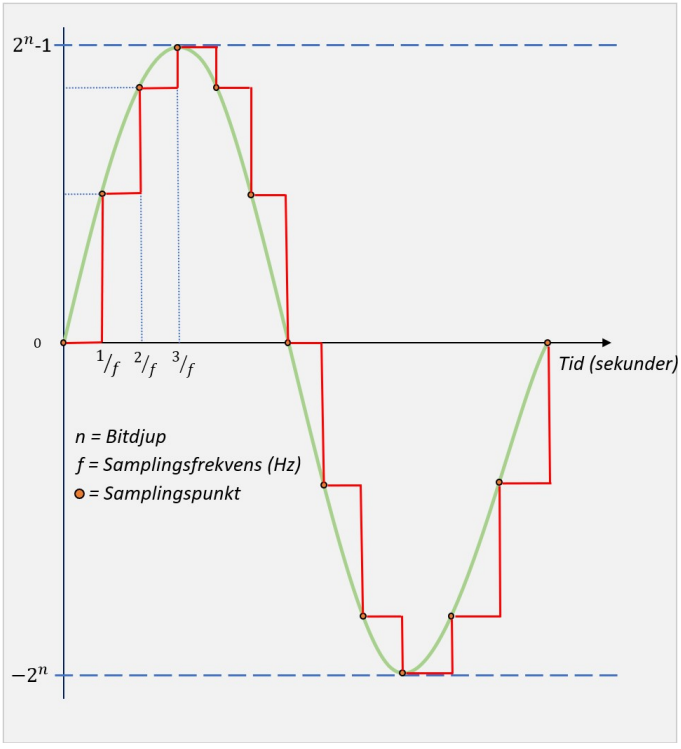
Den upplevda styrkan av ljudet beror på till vilken grad ljudvågen ändrar trycket i relation till det statiska trycket i mediumet. En större relativ förändring av trycket uppfattas ha starkare volym än en mindre.[1]

2.1.2 Analog representation

Ljud kan representeras som analoga elektriska signaler för vidare hantering eller lagring. Detta görs med hjälp av någon typ av mikrofon, ofta i kombination med en förstärkare då den resulterande elektriska signalen från en mikrofon är relativt svag. I mikrofonen sitter ett väldigt tunt och känsligt membran som kan röra sig både framåt och bakåt i relation till sin neutrala mittpunkt som den befinner sig i opåverkat läge. När en ljudvåg når mikrofonens membran reagerar den på tryckförändringarna genom att röra sig framåt och bakåt. Denna mekaniska svängning omvandlas till en elektrisk signal som sedan kan förstärkas och användas för vidare processering eller lagring. I en analog signal representeras alltså den fysiska vågens variation av lufttrycket som en variation av spänning över tid och tonhöjden beror således på signalens periodicitet.[3]

2.1.3 Digital representation

Analoga signaler kan omvandlas till en digital representation med hjälp av en analog-till-digital signalomvandlare(AD omvandlare). En analog signal som matas in i en A/D omvandlares kretsar kvantifieras med en förinställd regelbundenhet. Denna frekvens, med vilken signalen läses av, kallas för samplingsfrekvens och anger alltså hur många gånger per sekund A/D omvandlaren mäter och lagrar amplituden på den inkommande signalen. Således är den digitala representationen alltid en mer eller mindre grov approximation av den analoga signalen beroende på samplingsfrekvensen. En ögonblicksbild av denna process är illustrerad på Fig. 2.1 där den röda signalen är en digital approximation av den gröna analoga signalen.[3]



Figur 2.1: Kvantisering av en analog signal(grön linje) till digital representation(röd linje)

Figuren visar även hur en tidpunkt T går att uttrycka som en kvot mellan samplingspositionen från start och samplingsfrekvensen.

Wave fil

En Wave-fil är ett filformat för att lagra digital ljuddata i form av en bitström som representerar vågormen av ett ljud. Formatet är utvecklat är Microsoft och IBM och använder sig av Microsofts Resource Interchange File Format (RIFF) specifikation [4]. Enligt RIFF så delas filen upp i olika delar för att beskriva filens innehåll. De 4 första bitarna av filen innehåller American Standard Code for Information Interchange (ASCII) kod för att representera ordet "RIFF" som beskriver att det är en fil enligt RIFF specifikationen, nästa block av 4 bitar representerar filens storlek och sista blocket av 4 bitar är ASCII kod som representerar ordet "WAVE" för att specificera att filen innehåller ett ljud av formatet Wave [5]. Wave formatet består i sin tur av två delar: "fmt" och "data". Fmt delen beskriver ljudets format i form av bitdjup, mono eller stereo, samplingsfrekvens etc. Data biten består av den faktiska ljuddatan vilken är en siffra som representerar amplituden hos varje sampling. [4] [5] Fig. 2.2 visar hur en Wave-fil är uppbyggd enligt RIFF specifikationen.

WAVE formatet enligt RIFF specifikationen

Fil offset	Fältets namn	Fältstorlek	Beskrivning
0	Chunk ID	4	
4	Chunk Size	4	RIFF beskrivning, filen är av formatet WAVE och innehåller två delar: fmt-del och data-del.
8	Format	4	
12	SubChunk1 ID	4	
16	SubChunk 1 Size	4	
20	Audio Format	2	
22	Channel Number	2	"fmt" delen: Innehåller nödvändig information om Wave filens format så som samplingsfrekvens, mono/stereo etc.
24	Sample Rate	4	
28	Byte Rate	4	
32	Block Align	2	
34	Bits Per Sample	2	
36	SubChunk 2 ID	4	
40	SubChunk 2 Size	4	"data" delen: Fältet "SubChunk2 Size" indikerar ljudfilens storlek.
44	Data	SubChunk 2 Size	

Figur 2.2: Visuell beskrivning av en Wave fils uppbyggnad enligt RIFF specifikationen

2.1.4 Akustiskt fingeravtryck

Ett akustiskt fingeravtryck kan ses som en kort sammanfattning av en ljudfil. Att skapa ett fingeravtryck är komplext, för rapportens kontext presenteras därför bara

det grundläggande konceptet. För att få ett fingeravtryck skapas ett spektrogram av ljudfilen. Ett spektrogram är en 3 dimensionell graf av ett ljud som innehåller tid, frekvens och amplitud på respektive axel. Algoritmen delar därefter upp spektrogrammet i flera segment för att sedan välja ut punkter i grafen som motsvarar toner med högre energi jämfört med dess närliggande punkter. [6][7] Punkterna från varje segment bildar ljudets fingeravtryck.

2.2 Signalanalys

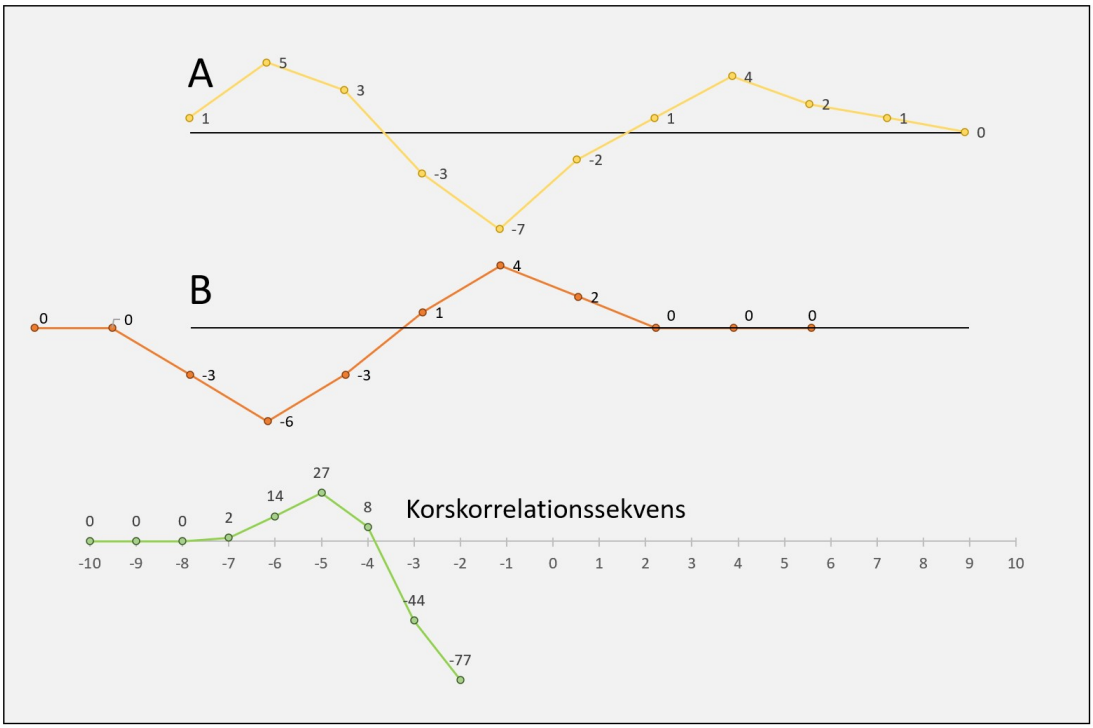
Det finns många olika sätt att analysera en ljudsignal, nedan presenteras de analysmetoder som systemet använder sig av.

2.2.1 Korrelation

Korrelation är ett ord som används för att beskriva ett förhållande mellan två eller flera objekt. En godtycklig signal är en typ av objekt som har en mätbar relation med en annan godtycklig signal. Två signaler har en stark relation om amplituden hos vardera signal reagerar på samma sätt vid en förändring i tid.

Korskorrelation är en typ av mätningsoperation inom signalanalys vars resultat beskriver likheten mellan två signaler vid olika tidsförskjutningar gentemot varandra. Med operationen bygger man upp en sekvens som innehåller alla möjliga positioner de två signaler kan överlappas vid. För varje position i sekvensen går det att läsa av hur stark korrelationen är. Sekvensen går att visualisera som en graf där x ledet är antal överlappningssteg och y värdet är de summerade produkterna av de överlappade värdena. Värdet y indikerar alltså hur stor likhet signalerna har vid överlappsposition x. [8]

Fig. 2.3 illustrerar en ögonblicksvy av sekvensens uppbyggnad där signal B befinner sig vid överlappsposition -2 gentemot signal A. Överlappningen sker från vänster till höger och signalerna är helt överlappade vid position 0.

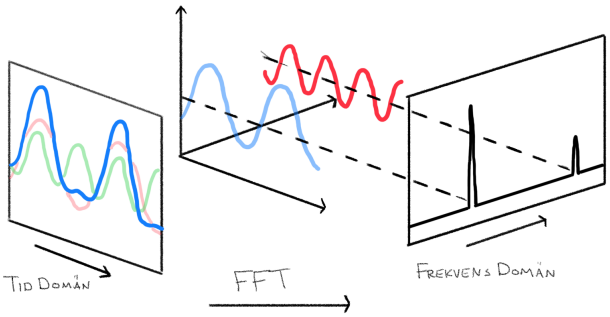


Figur 2.3: En ögonblicksbild tagen mitt i en korrelationssekvens uppbyggnad. Signal B är i detta ögonblick överlappad 8 steg från första överlappspositionen.

Om man har fastställt att det finns en korrelation mellan två signaler kan det även vara intressant att veta hur stark denna korrelation faktiskt är. Detta kan mätas med något som kallas ”Korrelationskoefficient”. Korrelationskoefficienten r går att räkna ut på ett antal olika sätt men resultatet är alltid en siffra mellan -1 och 1 där $r = -1$ innebär att signalerna har en direkt inverterad korrelation det vill säga, medan den ena signalen ökar i värde så minskar den andra. Korrelationskoefficient $r = 1$ betyder att de är identiska.[9]

2.2.2 Frekvensspektrum

Från en ljudsignal kan man räkna ut signalens frekvensspektrum som i sin tur kan visa vilka frekvenser signalen innehåller. Fast Fourier transform (FFT) är en metod som konverterar en singal till enskilda spektralkomponenter och kan på så sätt visa frekvensinformation från signalen. [10] FTT är en optimerad algoritm som baseras på Discrete Fourier Transformation (DFT). Signalen samplas över en tidsperiod och delas sedan upp i frekvenskomponenter. Komponenterna består av sinusformade signaler med distinkta frekvenser med en fast fas och amplitud. Förloppet visas i Fig. 2.4 där signalen, under en viss tidsperiod, består av 2 distinkta frekvensdomäner.



Figur 2.4: Vy av en ljudsignal i tid och frekvensdomänen

Enligt Nyquistfrekvensen [11] så måste samplingsfrekvensen för en signal vara dubbel så hög jämfört med den högsta frekvensen i den behandlade signalen. Med en samplingsfrekvens på 44100 Hz kan man således maximalt upptäcka ett ljud med frekvensen 22050 Hz. [11] Matematiskt kan man räkna ut samplingsfrekvens enligt Nyquist formel:

$$f_{Nyq} = 2 \cdot f_{naturlig}$$

2.2.3 Volym

Det finns flera olika sätt man kan bedöma hur högt ett ljud låter. Ett sätt är att beräkna ljudvågens Root Mean Square (RMS) värde. Som namnet antyder så representerar RMS värdet kvadratroten av genomsnittsvärdet av ljudvågens amplituder. Värdet konverteras sedan till den logaritmiska skalan för att få ett värde i decibel. För att räkna ut ljudvågens RMS så används följande formel där n = antal samplingar, x = amplitudvärdet.

$$RMS = \sqrt{\frac{1}{n} \cdot (x_1^2 + x_2^2 + ... + x_n^2)}$$

Konvertering till dB:

$$RMS_{dB} = 20 \cdot \log_{10}(RMS)$$

3
Metod och genomförande

I följande avsnitt presenteras projektets olika arbetsfaser, vilka verktyg som använts samt hur projektet planerats och genomförts.

3.1 Verktyg för utveckling

När projektets syfte och mål blev definierat påbörjades en efterforskning för att hitta ett bibliotek som kunde hantera ljuddata från en Wave-fil. En viktig aspekt var att biblioteket skulle kunna presentera datan på ett sätt som gjorde det möjligt att analysera och jämföra datan mot en annan Wave-fil. Systemet är utvecklat i Java med Visual Studio Code IDE[12]. Valet av Java gav oss möjligheten att utveckla enligt ett objektorienterat arbetssätt.

3.1.1 Musicg

Musicg är ett bibliotek för ljudanalys utvecklat i Java. Musicg kan användas för att extrahera ljuddata från en wave-fil [13]. Biblioteket tillhandahåller ett Application Programming Interface (API) som gör det möjligt att bland annat rendera en vågform eller spektrogram, editera en wave-fil, läsa frekvens-tid data och läsa amplitud-tid data.

3.1.2 MATLAB

För att underlätta vid signalanalysen användes en integration av programmet MATLAB[14] som tillhandahåller att API som gör det möjligt att skriva MATLAB - kod direkt i Java. Med hjälp av integrationen kunde avancerade signalberäkningar utföras och visualiseras i form av grafer. Graferna kunde sedan utvärderas för att säkerhetsställa att analysmetoderna gav önskat resultat. Projektets system har utvecklats i Java, inte bara på grund av valet av Musicg, utan också för att det ansågs underlätta vid skapandet av en objektorienterad struktur som var lätt att följa och skala upp.

3.1.3 Utrustning

För projektets ljudinspelningar användes till största del redan befintlig utrustning i form av mikrofoner och ljudkort. Senare i projektet användes dock en speciell mätmikrofon, Superlux ECM999[15], som är anpassad för ljudanalys. Till denna

användes ett portabelt ljudkort, TC-Helicon Go Solo[16], som anslöts via USB till datorn som systemet kördes på.

3.2 Planering

Inledningsvis skapades ett flödesschema av hur systemet skulle fungera, utifrån schemat skapades en grov modell av systemets funktionalitet. Processchemat och modellen baserades på diskussioner med Diadrom kring vad som var önskvärt och realistiskt för systemet inom projektets tidsram. Ett preliminärt tidsschema skapades för att lättare kunna strukturera arbetet från start till slut. För att visualisera planeringen och tidsschemat användes Microsoft Planner[17] för att skapa en digital tavla med modellens olika delar. Varje del kunde sedan delas upp i mindre och mer konkreta moment som kunde bockas av allt eftersom arbetet fortskred.

3.2.1 Modellen

Programmeringsmodellen strukturerades med hjälp av ett Unified Modeling Language (UML) diagram. På så sätt kunde utvecklingens olika delar visualiseras. Systemets funktioner delades upp i klasser med tydliga ansvarsområden. Systemet UML finns i Bilaga.A.2.

3.3 Utveckling

En blandning av teoretisk efterforskning och praktisk experimentation användes för att bygga och förbättra systemet. Dessa två delar utfördes simultant genom hela processen eftersom de beror av varandra. Insamlad information från efterforskning applicerades och integrerades med systemet och utvärderades därpå för en bedömning av dess användbarhet. Löpande avstämningsmöten med Diadrom genomfördes för att diskutera idéer och för att få återkoppling på arbetet.

3.4 Testning

För att testa att klassernas metoder fungerade som förväntat har framförallt JUnit använts. Man vet i förtid vad metoden skall returnera, Man kontrollerar sedan om metodens retur är enligt specifikation. För en detaljerad vy av testerna se Bilaga. A.1.

För att testa och utvärdera systemets signalanalyser spelades olika ljud in med mikrofon. Ljuden som testades hade olika karaktär samt inspelningsförhållanden i form av avstånd och vinkel från ljudkällan. Dessa användes sedan som en fast utgångspunkt för att simulera både inkommande ljud till systemet och referensljud

vilket gav en indikation på, till exempel, hur väl en analysmetod i systemet presterade gentemot en annan. Fig. 3.1 visar hur mikrofonen placerades vid ljudkällan.

Figur 3.1: Mikrofonens placering vid ljudkällan

För att ytterligare testa analysmetoderna så modifierades inspelade ljud genom att editera in avvikelser i form av frekvensskillnader, längdskillnader och volymskillnader genemot referensljudet. Genom att ladda in det editerade ljudet i systemet kunde funktionerna testas för att säkerhetsställa att alla avvikelser hittades.

4

Systembeskrivning

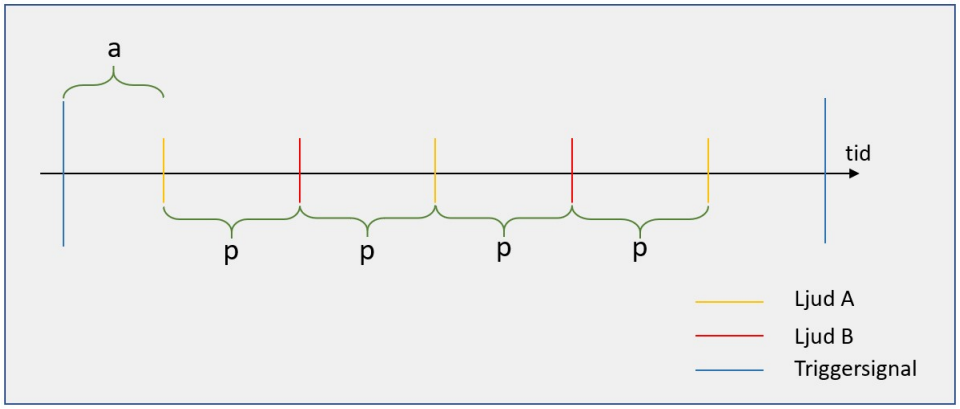
Det här kapitlet ämnar att förklara hur systemet är uppbyggt och hur en användare kan interagera med det. Första sektionen kommer förklara systemets flöde från start till slut av ett användningsfall vilket kommer introducera begrepp som förklaras närmare senare i kapitlet.

4.1 Systemflöde

Systemets flöde illustreras lättast genom ett konkret användningsfall som presenteras i följande stycken.

4.1.1 Exposition

Anta ett användningsfall där man har ett fysiskt objekt med flera integrerade system i samspel. Ett av dessa system är dedikerat till att spela upp ljud vars funktion är att upplysa användaren om statusändringar hos andra system och är det system som i det här fallet skall testas. En sådan ljuduppspelning kan aktivt triggas av användaren genom en fysisk signal eller från ett av de många integrerade systemen. I detta användarfall väljer användaren att trigga en signal som enligt en systemspecifikation bland annat skall resultera i att ljuduppspelningssystemet spelar upp två olika ljud alternerande med en fast periodicitet tills dess att användaren fysiskt triggar en signal som avaktiverar ljuduppspelningen. En del av specifikationen specificerar hur ljuden skall förhålla sig i tid gentemot varandra och aktiveringsignalen.



Figur 4.1: En specifikation för ett test av två alternerande ljud aktiverat och avaktiverat med triggersignal från system

Denna del är illustrerad i Fig. 4.1 där a är tiden från dess att användaren aktiverade systemet till dess att första ljudet förekommer och p är tidsperiodiciteten med vilken ljuden alternerar. Utöver detta specificeras även hur långt varje ljud skall vara samt hur det skall låta enligt en ideal kopia på ljudvågorna som systemet spelar upp.

4.1.2 Konfiguration

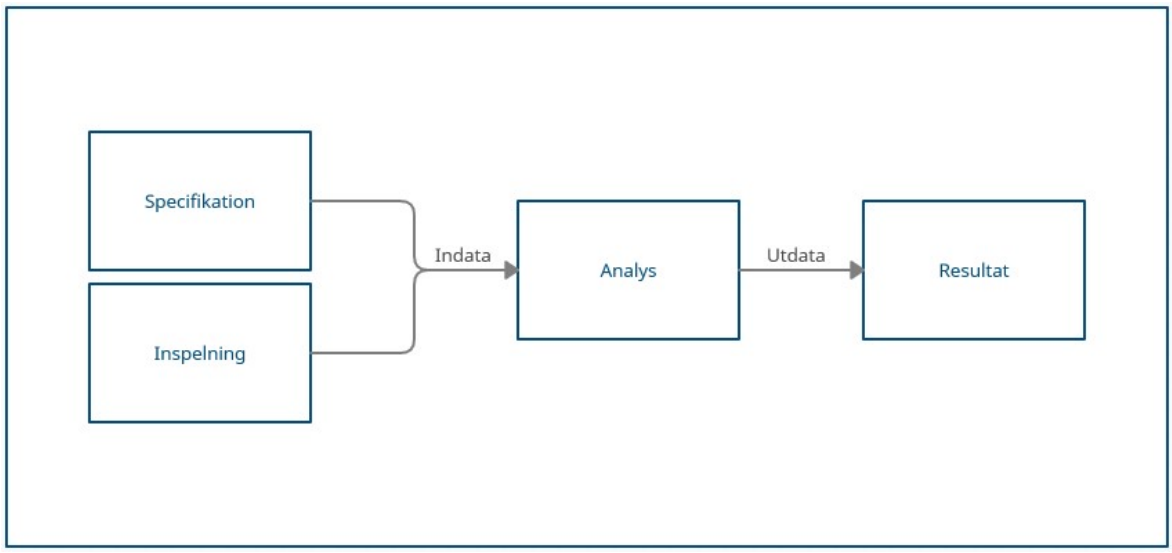
Baserat på specifikationen konfigurerar användaren de tidskonstanter som är önskade och eventuella toleransnivåer om avvikelser skall accepteras av testsystemet. Användaren konfigurerar även hur lång tid testsystemet skall spela in ljud från sin mikrofon innan den börjar analysera inspelningens innehåll. Till konfigurationen hör även att användaren sätter upp mikrofonen och kopplar samman den med datorn som systemet exekveras på. Med systemet konfigurerat och mikrofonen uppsatt startar användaren testet och aktiverar triggersignalen på testobjektet.

4.1.3 Testresultat

Testsystemet läser in den inspelade filen, delar upp den i mindre instanser av identifierade ljud och returnerar analysresultat för var och en. Användaren väljer om testresultatet skall loggas och med det avslutas testomgången.

4.2 Systemmodeller

För att förklara systemet lite djupare introduceras nu ett antal modeller. Den mest övergripande modellen som illustreras på Fig. 4.2 delar upp systemet i fyra delar som nu kommer att brytas ned och förklaras ytterligare.



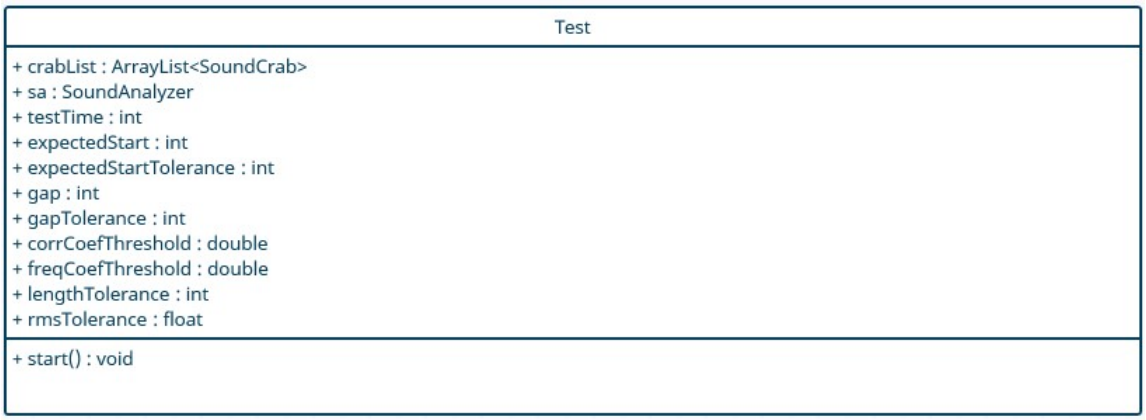
Figur 4.2: Den mest övergripande modellen av systemet.

4.2.1 Specifikation

För att systemet skall kunna lägga värdering i analysens utfall krävs en specifikation som beskriver vad som förväntas av inspelningen som är den första delen av indata som systemet får till sig.

Testobjekt

En testomgång representeras konceptuellt av informationen enligt Figur.4.3. Objektet skapas när användaren anger testets specifikation.



Figur 4.3: Modell av ett objekt av typen ”Test”. Detta är ett objekt användaren skapar när den vill initiera en ny testomgång.

Användaren har möjlighet att tillhandahålla ett flertal specifikationer för testet:

- **testTime:** Anger hur lång tid testets inspelning ska pågå innan analys sker.
- **expectedStart:** Specificerar hur lång tid det ska dröja innan första ljudet i inspelningen börjar.
- **expectedStartTolerance:** Specificerar en avvikelsetolerans på starttiden.
- **gap:** I fallet då inspelningen är en serie av periodiska ljud specificerar detta fält denna periodicitet.
- **gapTolerance:** Specificerar en avvikelsetolerans på periodiciteten.
- **corrCoefThreshold:** Specificerar ett tröskelvärde för korrelationskoefficienten framtagen mellan varje ljudklipp och dess referensljud.
- **freqCoefThreshold:** Specificerar ett tröskelvärde för korrelationskoefficienten framtagen mellan varje ljudklipps frekvensinnehåll och dess referensljuds frekvensinnehåll.
- **lengthTolerance:** Specificerar en avvikelsetolerans på ljudklippets längd.
- **rmsTolerance:** Specificerar en avvikelsetolerans på ljudklippets volym.

4.2.2 Inspelning

Inspelningen är det första som sker när användaren kallar på testobjektets startmetod och är den andra delen av indatan som används vid analys. Systemet spelar

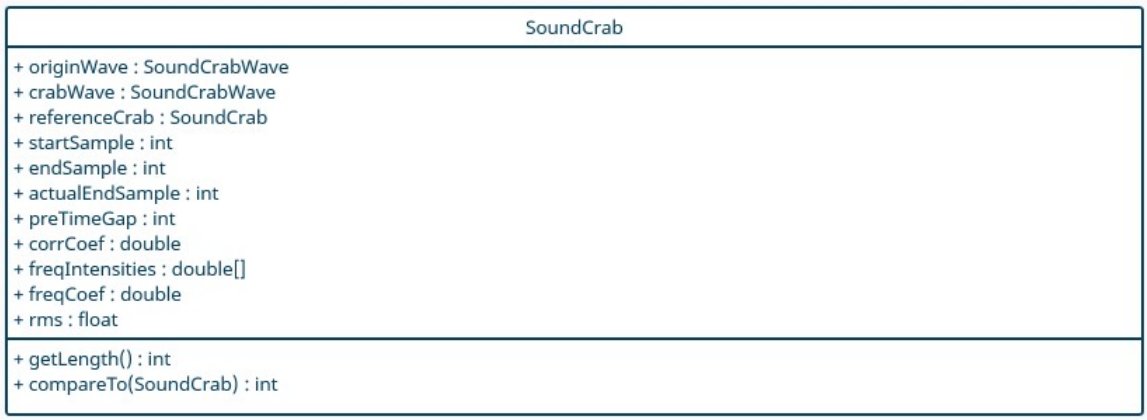
in ett ljud vars längd bestäms av användaren med testTime-fältet i specifikationen. Sedan tas det akustiska fingeravtrycket på inspelningen och jämförs med en databas av alla referensers fingeravtryck. På så sätt kan systemet snabbt upptäcka vilken referens som skall väljas för vidare analys av inspelningen. Användaren har också möjlighet att istället manuellt välja vilken referens som skall användas.

4.2.3 Analys

I analysdelen av systemet behandlas den inspelade signalen och information om helheten såväl som delar av den extraheras och lagras för jämförelse med specifikation i den senare resultatdelen.

SoundCrabs

Specifikationen kan innehålla krav dels på enskilda instanser av ljudklipp i den längre inspelningen men även krav på hur dessa små instanser av ljud skall förhålla sig gentemot varandra i tid. Det är således naturligt att systemet delar upp den stora signalen till flera objekt innehållande dessa mindre ljudklipp samt dess tidsegenskaper gentemot helheten. Systemet definierar ett sådant objekt som ett objekt av typen ”SoundCrab”. Varje SoundCrab kan behandlas individuellt av systemet och därmed erhålla egna analysdata, se Fig. 4.4 för en modell av ett sådant objekt.



Figur 4.4: Modell av ett ”SoundCrab” objekt. Systemet skapar ett sådant objekt per identifierat ljud och lagrar där information dels om dess position i sitt ursprungsljud men också analytisk information såsom frekvensinnehåll och volym mm.

En ”SoundCrab” är rent konceptuellt ett objekt innehållande en ljudvåg samt information från analys av denna ljudvåg. Om objektet skapas i samband med att dess ljudvåg identifieras i en längre inspelning ser systemet till att även lagra en referens till ursprungsljudvågen och en referens till det identifierade ljudets ideal som en ny SoundCrab. När analys på den inneboende ljudvågen sker lagras resultatet i SoundCrab-objektet. En ”SoundCrab” innehåller följande fält:

- **originWave:** en referens till den ursprungliga inspelningen som ett wave-objekt.
- **crabWave:** en referens till den identifierade delen av ursprungsinspelningen som ett wave-objekt.
- **referenceCrab:** en referens till ett SoundCrab-objekt av den ideala instansen av ljudet som identifierats.
- **startSample:** samplevärdet vid vilket ljudet startar i sin ursprungsfil.
- **endSample:** samplevärdet vid vilket ljudet antas sluta i sin ursprungsfil.
- **actualEndSample:** samplevärdet vid vilket ljudet slutar enligt analys.
- **preTimeGap:** skillnaden i samples mellan detta ljudets start och föregående ljudets start alternativt starten av inspelningen i fallet då soundcrab-objektet är det första identifierade ljudet i en inspelning.
- **corrCoef:** korrelationskoefficienten från analys mellan detta ljudet och dess ideala referens.
- **freqIntensities:** ljudets intensitet som en funktion av frekvensvärde
- **freqCoef:** korrelationscoeficient vid analys mellan ljudets frekvensinnehåll och den ideala referensens frekvensinnehåll
- **rms:** RMS-nivå på ljudet.

Vad systemet tolkar som en "intressant" del av helheten avgörs av vilket typ av ljud användaren av systemet väljer att testa. Systemet har möjlighet att läsa in ett eller flera referensljud för användning vid test. Detta sker antingen genom att användaren i specifikationen anger vilket eller vilka referensljud som skall användas eller att systemet själv försöker identifiera det utifrån inspelningens akustiska fingeravtryck. Referensljuden är ideala instanser av ljuden som systemet förväntar sig för ett specifikt test och används alltså i identifikationsprocessen och analysprocessen.

Identifikationsprocessen

En förutsättning för att systemet skall kunna jämföra inspelade ljud mot referensljud är att det på ett robust sätt kan identifiera och plocka ut de individuella ljuden ur en längre inspelning som eventuellt kan innehålla flera instanser av olika ljud. För att göra detta använder systemet sig av en korskorrelationssekvens som byggs upp av en korskorrelationsoperation med den inspelade signalen och referenssignalen som operander. I den resulterande sekvensen kommer en likhet kunna identifieras som ett lokalt maxima och dess position i x-led indikerar vilken sample i den längre inspelningen som det identifierade ljudet börjar. Som ett första steg antar systemet att det identifierade ljudet är lika långt som referensljudet och får därmed även en preliminär slutposition för ljudet. Med denna information kan systemet skapa ett SoundCrab objekt genom att först klona ursprungssignalen och sedan trimma bort allt innan den identifierade starten och allt efter det antagna slutet. Denna process sker för alla lokala maxima i korrelationssekvensen.

Det faktiska slutet av ett utklippt ljud undersöks av systemet på så sätt att det utgår ifrån det antagna slutet i inspelningen och kollar att genomsnittsvärdet på

ett antal samples framåt är under en viss gräns. Allt ovanför gränsen anses som en förlängning av det identifierade ljudet och allt under gränsen tolkas som en del av det statiska bakgrundsljud som ej behöver tas hänsyn till. Det faktiska slutet sätts således som den punkten då genomsnittsvärdet är under gränsen.

Volym

För att bedöma högljudheten av det inspelade ljudet räknas ljudsignalens RMS värde ut för varje identifierad SoundCrab enligt koden nedan. Referensens RMS värde räknas ut på samma sätt vilket gör att värdena kan jämföras för att bedöma om högljudheten är inom tolerans.

```
public void calculateRms(List<SoundCrab> scArr) {
    for(SoundCrab s : scArr){
        int n = s.crabWave.getLength();
        short[] arr = s.crabWave.getSampleAmplitudes();
        long square = 0;
        float root;

        for (short value : arr) {
            square += Math.pow(value, 2);
        }

        float mean = (square / (float) (n));
        root = (float) Math.sqrt(mean);
        s.rms = (float) (20 * Math.log10(root));
    }
}
```

Korrelationskoefficient & frekvensanalys

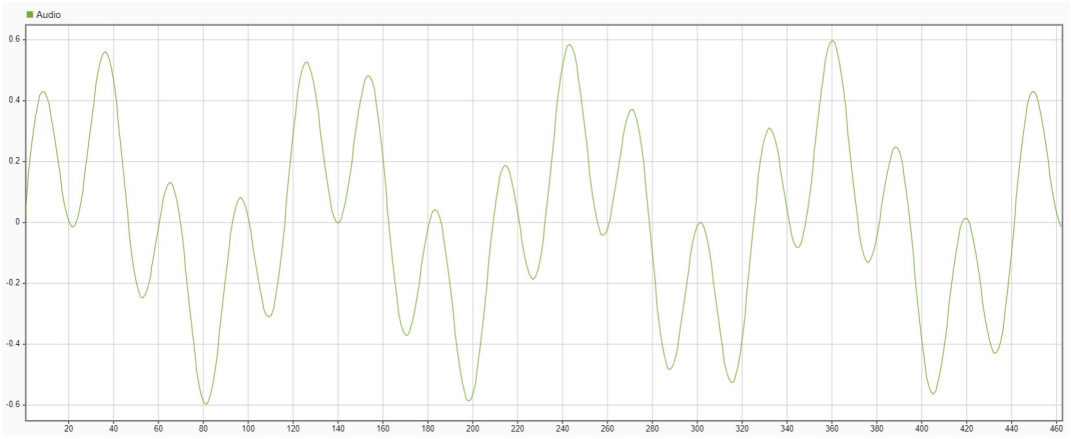
Systemet avgör likhet mellan identifierat ljud och dess referens med hjälp av korrelationskoefficienten. Dels mellan de obehandlade signalerna men även mellan respektive signalers frekvensspektrum.

För att få ut ett frekvensspektrum för varje SoundCrab körs en sekvens av MATLAB funktioner (visas nedan) för att räkna ut signalens FFT. Resultatet av uträkningarna blir två listor, en som innehåller ljudets intensiteter i olika frekvenskomponenter och en andra som innehåller alla frekvenskomponenter, som beror på ljudets samplingsfrekvens.

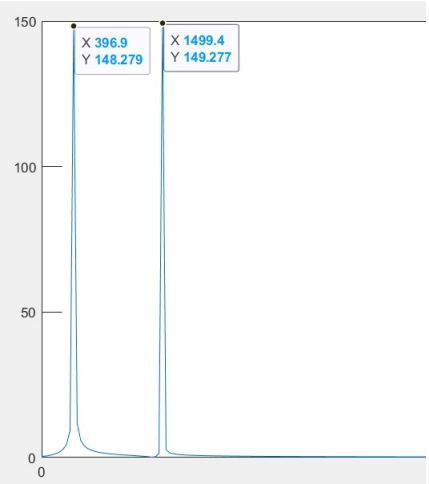
```
data = crabWave.normalizeOnPeak();
N = length(data);
Fs = samplingRate;
dataFFT = fft(data);
dataFFTabs = abs(dataFFT);
f = Fs*(0:N/2)/N;
```

För att hitta vilken frekvensenkomponent som har högst intensitet i ljudfilen kollar man index för det högsta värdet i intensitetslistan för att sedan kolla vilken frekvens det motsvarar genom att kolla på samma index i listan av frekvenskomponenter. I Fig. 4.5 visas vågformen hos en genererad ton som innehåller två distinkta frekvenser,

400 Hz och 1500 Hz. Fig. 4.6 visar resultatet av FFT uträkningen. Från analysen kan man tydligt se att ljudfilen består av just 400 Hz och 1500 Hz.



Figur 4.5: Genererad ton med frekvenser 400 Hz och 1500 Hz



Figur 4.6: Frekvenskomponenter från en den genererade tonen med frekvens 400 Hz och 1500 Hz

Genom denna analys kan vi verifiera att det inspelade ljudet innehåller samma frekvensspektrum som sin referens. Genom att räkna ut korrelationskoefficienten mellan inspelningens och referensens FFT grafer bedöms det om frekvensanalysen är godkänd eller ej. Den uträknade korrelationskoefficienten sparas i en variabel hos varje identifierad SoundCrab.

4.2.4 Testresultat

Systemet sparar värden från signalanalyserna i respektive SoundCrab. Värdena jämförs sedan mot referensljudets värden samt annan data i specifikationen och presenterar resultatet i form av en text. Texten specificerar vilka tester som godkändes eller ej samt presenterar alla värden som analyserna räknat ut. I Fig. 4.7 visar hur

ett resultat av en analys presenteras för användaren. Det uppmätta värdet samt tolerans enligt specifikation visas inom parenteserna.

```
Crab 1
Title: RippedTillslag
Correlation: false     (0.6422562656700007 | 0.8)
Rms: true             (56.8072 | 53.76104 +- 100.0)
Freqcorrelation: false (0.6340161346164519 | 0.9)
Timing: false          (21055 | 17500 +- 300)
Length: false          (1289 | 590 +- 200)

Crab 2
Title: RippedFrånslag
Correlation: true     (0.9999704849220594 | 0.8)
Rms: true             (51.17839 | 51.178276 +- 100.0)
Freqcorrelation: true (0.9999892856070232 | 0.9)
Timing: true          (17642 | 17500 +- 300)
Length: true          (590 | 590 +- 200)

Crab 3
Title: RippedTillslag
Correlation: true     (0.9999833066003148 | 0.8)
Rms: true             (53.76104 | 53.76104 +- 100.0)
Freqcorrelation: true (0.9999936894462438 | 0.9)
Timing: true          (17642 | 17500 +- 300)
Length: true          (590 | 590 +- 200)

Crab 4
Title: RippedFrånslag
Correlation: true     (0.9950490491700847 | 0.8)
Rms: true             (51.102634 | 51.178276 +- 100.0)
Freqcorrelation: true (0.9934451366508803 | 0.9)
Timing: false          (18736 | 17500 +- 300)
Length: true          (590 | 590 +- 200)
```

Figur 4.7: Text som specificerar testets resultat

4.3 Systemstruktur

Systemets struktur är designat på ett sätt så att varje klass ska ha en tydligt ansvarsområde. Strukturen visas bäst genom ett UML klassdiagram. Se Bilaga. A.2 för systemets UML klassdiagram.

5
Resultat

För att kunna avgöra hurvida de satta målen har uppfyllts presenteras i detta kapitel resultat av utförda experiment och utvärderingar.

5.1 Ljudidentifikation

En av de absolut viktigaste delarna hos systemet är dess förmåga att identifiera tidsegenskaper hos sekvenser av ljud i en större ljudbild. Eftersom resten av systemet bygger på detta är det viktigt att det görs på ett sätt som är robust och pålitligt. Denna robusthet går att testa genom att låta systemet försöka identifiera startpunkten på ett givet ljud i en inspelning och jämföra det med den faktiska startpunkten. För varje testomgång ökar man volymen på någon typ av bakgrundsstörning.

Brusvolym (RMS)	Förväntad startsample	Identifierad startsample
$-80dB$	1000	1000
$-70dB$	1000	1000
$-60dB$	1000	1000
$-50dB$	1000	1000
$-40dB$	1000	1000
$-30dB$	1000	1000
$-22dB$	1000	1000

Tabell 5.1: Resultattabell för ett identifikationstest där ljudet som ska identifieras har en fast volym mellan omgångarna och startar på samplingsposition 1000 i inspelningen. Störningen är av typen vitt brus och volymen är uppmätt med RMS

Tab. 5.1 visar resultatet av ett sådant experiment där en signal skapades på ett sådant sätt att man i förväg visste vilken samplingsposition ljudet man vill identifiera startar på. Utifrån detta test kan man se att starten på ljudet identifierades korrekt även vid testomgången med den starkaste störningen som gick att generera. Störningen är vid det här laget såpass stark att det mänskliga örat inte längre kan urskilja ljudet som systemet försöker identifiera från störningen.

I det första experiment var volymen på den intressanta delen av signalen fast vid -35.5dB RMS medan störningens volym ökades inför varje omgång. För att vidare testa processens robusthet utförs ett liknande experiment med skillnaden att man har en fast störning på -22dB men att ljudet som skall identifieras sänks i volym

inför varje testomgång så att den relativa skillnaden i volym mellan det intressanta ljudet och störningen alltså blir ännu högre.

Relativ volymskillnad (RMS)	Förväntad startsample	Identifierad startsample
13.5dB	1000	1000
14.5dB	1000	999
15.5dB	1000	999
16.5dB	1000	1035
17.5dB	1000	1035
18.5dB	1000	74 och 1109
23.5dB	1000	74 och 1109
28.5dB	1000	74
33.5dB	1000	1109

Tabell 5.2: Resultattabell för ett test av identifikationsmetoden där den relativa volymskillnaden mellan störningen och ljudet som identifieras trappas upp ytterligare.

Resultatet som är presenterat i Tab. 5.2 visar på att systemet börjar få problem med att hitta korrekt startsample först när den relativa volymskillnaden närmar sig 14,5dB vilket är en indikator på en utmärkt robusthet då det innebär en störningsnivå som är långt över den som systemet rimligtvis kommer verka i.

En sådan robust metod uppfyller dels direkt det uppsatta delmålet att kunna identifiera startpunkter för ett specifikt ljud men även indirekt målet om att kunna mäta tiden mellan olika ljud i en inspelning då detta värde går att tolka som skillnaden mellan ljudens startpositioner.

5.2 Likhetsanalys

För att avgöra hur likt ett ljud är sin referens undersöker systemet dels korrelationskoefficienten mellan ljudet som testas och dess ideala referensljud men även korrelationskoefficienten mellan vardera ljuds frekvensinnehåll.
Hur och var systemets mikrofon är placerad har en stor påverkan på systemets resultat. För att förstå hur mikrofonens placering påverkar resultatet utfördes tester där samma ljud spelades in med olika lägen från ljudkällan. Det visade sig att systemet alltid kunde identifiera och hitta ljuden oavsett avstånd. Däremot blev systemets förmåga att bedöma korrektheten hos det inspelade ljudet sämre när avståndet till ljudkällan ökade. Mikrofonens vinkel från ljudkällan visade sig spela mindre roll då mätningarna med en vinkel på 0 respektive 90 grader visade liknande resultat på majoriteten av mätningarna. Däremot påverkar vinkeln resultatet vid större avstånd från ljudkällan. Mätningarna gjordes med en fast volym på ljudkällan och en fast förstärkning för mikrofonen. Tab. 5.3 visar mätningarna. Vid ca 30cm avstånd ser man att korrelationskoefficienten minskar till en nivå när man inte längre kan avgöra om ljudet är tillräckligt likt referensen eller inte. Förklaring till varför mikrofonens placering är så viktig diskuteras i stycke 6.1.

Distans:	Vinkel:	Korrelationskoefficient:	Frekvenskoefficient:
$2cm$	0°	0,91	0,97
$10cm$	0°	0,85	0,96
$30cm$	0°	0,81	0,94
$50cm$	0°	0,75	0,87
$2cm$	90°	0,84	0,96
$10cm$	90°	0,90	0,96
$30cm$	90°	0,81	0,94
$50cm$	90°	0,64	0,74

Tabell 5.3: Resultattabell som visar hur testernas värden ändras av mikrofonens placering från ljudkällan. Distans och vinkel är relativ mot ljudkällan.

5.2.1 Missljud

Ett experiment för att utröna hur väl dessa korrelationskoefficienter indikerar närvaro av missljud genomfördes genom att man lät systemet testa ett ljud som är identiskt till sin referens i alla avseenden förutom att en ren sinuston manuellt lades till i ljudbilden innan testet, en mänsklig testperson fick även lyssna på varje ljud för att avgöra om den kunde identifiera missljudet. Experimentet utfördes ett flertal gånger med olika frekvenser och volym på sinustonen. Resultaten presenteras i Tab. 5.4 och Tab. 5.5 där den första mäter korrelationen mellan signalerna och den andra mäter korrelationen mellan frekvensinnehåll. Markeringen '(H)' indikerar volymnivån vid vilken den mänskliga testpersonen först upptäckte missljudet i ljudbilden.

Volym	50 Hz	100 Hz	1000 Hz	3000 Hz	10000 Hz	15000 Hz	18000 Hz
$-60dB$	0.99	0.99	0.99	0.99	0.99(H)	0.99	0.99
$-50dB$	0.99	0.99	0.99	0.98(H)	0.98	0.98(H)	0.99
$-40dB$	0.92(H)	0.91(H)	0.90(H)	0.90	0.90	0.90	0.90
$-30dB$	0.59	0.57	0.59	0.56	0.57	0.56	0.56

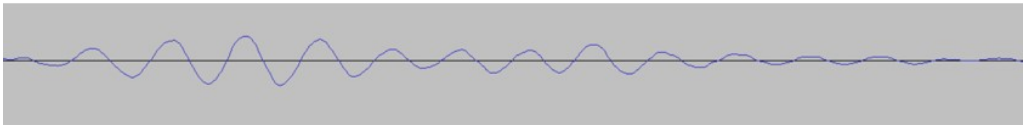
Tabell 5.4: Resultattabell från experiment där missljud vid olika frekvens och volym manuellt lagts till innan test. Likheten är här uppmätt med korrelationskoefficient mellan ljudet som testas och dess referens. Volymen är uppmätt med RMS beräkning.

Volym	50 Hz	100 Hz	1000 Hz	3000 Hz	10000 Hz	15000 Hz	18000 Hz
$-60dB$	0.99	0.99	0.99	0.99	0.99(H)	0.99	0.99
$-50dB$	0.98	0.98	0.99	0.97(H)	0.98	0.98(H)	0.99
$-40dB$	0.89(H)	0.90(H)	0.91(H)	0.87	0.89	0.90	0.90
$-30dB$	0.51	0.55	0.78	0.51	0.54	0.51	0.54

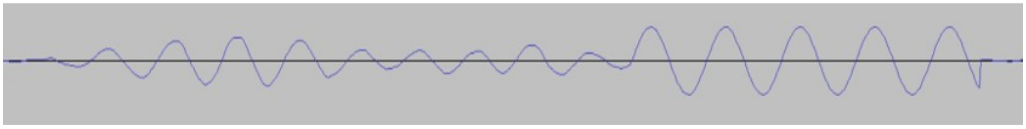
Tabell 5.5: Resultattabell från experiment där missljud vid olika frekvens och volym manuellt lagts till innan test. Likheten är här uppmätt med korrelationskoefficient mellan frekvensinnehållet för ljudet som testas och dess referens. Volymen är uppmätt med RMS beräkning.

5.2.2 Frekvensanalys

Genom att lägga till toner med annat frekvensinnehåll kunde frekvensanalysen verifieras. I Fig. 5.1 visas en omodifierad inspelning av ett blinkerstillslag, vid jämförelse med dess referens blir korrelationskoefficienten av frekvensspektrum 0,97 vilket betyder att deras frekvensspektrum är väldigt lika. I Fig. 5.2 visas samma blinkerstillslag fast med en editerad del som innehåller ett missljud. Detta leder till att korrelationskoefficienten av frekvensspektrum med referens istället visar 0,75 vilket betyder att det finns viss avvikelse i frekvensspektrum.



Figur 5.1: Inspelad ljudsignal av ett blinkerstillslag, frekvenskoefficient = 0,97



Figur 5.2: Editerad inspelning av blinkerstillslag, frekvenskoefficient = 0,75

5.2.3 Ljudets uppmätta volym

Inspelningens RMS värde är en direkt funktion av vågformens amplituder. För att testa att RMS värdet fungerade i verkligheten genomfördes tester där mikrofonen gain (förstärkning) successivt ökade mellan inspelningar. Teoretiskt ska en högre förstärkning betyda att volymen blir högre.

Resultatet av mätningarna presenteras i Tab. 5.6. I tabellen kan man se att RMS värdet ökar för varje inspelning och att teorin om att RMS värdet ökar vid högre förstärkning stämmer. Resultatet stämmer även överens med den upplevda volymen när testet genomfördes. Vid uppspelning av ljuden upplevdes inspelningen med förstärkning 8 ha högre volym jämfört med inspelningen med lägre förstärkning.

Förstärkning (1 - 10)	Uppmätt RMS värde
1	26,95
2	29,04
4	37,46
6	54,22
8	61,57

Tabell 5.6: Resultattabell för test att RMS värdet ökar med en högre förstärkning

För att ytterliga testa analysen utfördes tester där ett bakgrundsljud på 20 000 Hz, som det mänskliga örat inte kunde höra, lades till. Resultaten visas i Tab. 5.7. Man

kan se att det uppmätta volymen med bakgrundsljud är mycket högre samtidigt som den mänskliga hörseln inte upplever någon skillnad i volym. Se stycke 6.1 för en förklaring till detta.

Förstärkning (1 - 10):	RMS utan bakgrundsljud:	RMS med bakgrundsljud:
1	26,95	52,94
2	29,04	54,27
4	37,46	63,12
6	54,22	78,86
8	61,57	87,01

Tabell 5.7: Resultattabell för test att med högfrekvent bakgrundsljud

6
Diskussion

Detta kapitel är dedikerat till diskussion kring rapportens huvuddel.

6.1 Frågeställningar

Då syftet med rapporten är att avgöra till vilken grad man kan eliminera den mänskliga faktorn med ett datorsystem vid testning av ljud så är det intressant att diskutera hur väl resultatet står sig jämfört med en mänsklig testare.

- Hur väl står sig det framtagna systemet mot människans förmåga att avgöra timing?

  Delen av systemet som identifierar startpositioner på instanser av ljud är enligt de presenterade resultaten väldigt robust och pålitlig. En intressant del i det resultatet är att systemet korrekt identifierar starten på ett ljud som har en såpass stark bakgrundsstörning att en människa inte kan urskilja att ljudet ens är närvarande. Dessutom har en människas förmåga att identifiera avvikelser i timing inte riktigt lika hög precision som systemet har gällande detta.

- Hur väl står sig det framtagna systemet mot människans förmåga att avgöra volymskillader?

  Precis som med timing har systemet förmågan att på ett väldigt objektivt sätt avgöra volymen hos en ljudfil. Däremot tar den inte hänsyn till den subjektiva delen av människans upplevelse. Eftersom människor upplever olika frekvenser med samma intensitet som olika starka i volym hade en mer utförlig filtrering behövts för att kunna få en mer korrekt siffra enligt människans subjektiva upplevelse. Ett exempel på detta visar sig i Tab. 5.7 där man kan se att RMS värdet påverkas av frekvenser som människan inte uppfattar. Detta faktum förminskar ju dock inte volym uppmätningens användbarhet i systemet som en objektiv mätdata.

- Hur väl står sig det framtagna systemet mot människans förmåga att upptäcka missljud?

  Den här frågeställningen är väldigt intressant då det i vanliga fall är upp till

personen som testar att avgöra vad som kvalificeras som ett missljud och inte. Systemet har möjlighet att jämföra ett inspelat ljud med en perfekt instans av det ljudet och kan avgöra hur lika de är rent objektivt med hjälp av korrelationskoeficienten mellan själva ljudsignalerna men även frekvensinnehållet. Detta är en bra indikation på hur lika ljuden är men enbart denna metod räcker inte för att eliminera en människa ur testprocessen eftersom det i vissa fall endast krävs väldigt små justeringar av signalen för att en mänsklig testare ska uppfatta ett missljud. Dessa justeringar är såpass små att korrelationskoeficienten fortfarande visar en hög likhet mellan signalerna. Detta visar sig i experimentet som beskrivs i sektion 5.2.1.

Vid vissa frekvenser kunde testpersonen höra störningen medan korrelationskoefficienten fortfarande visar på en väldigt stark korrelation. Så för att kunna ersätta människans hörsel här krävs ett system som är känsligare för dessa små förändringar. Experimentet visar också att det motsatta kan ske, vid - 30dB på 18 000Hz uppfattade inte den mänskliga testaren något missljud medan systemet indikerade en låg korrelation.

Ett annat förväntat problem som uppenbarar sig med den här metoden är att mikrofonen och dess placering i testmiljön är en stor felkälla. I Tab. 5.3 kan vi se hur korrelationskoefficienterna ändrar sig beroende på mikrofonens position i förhållande till ljudkällan. Detta är pågrund av att en större del ljudvågorna från ljudkällan har påverkats av omgivningen innan de når mikrofonen vilket skapar oförutsägbara variationer hos signalen som mikrofonen matar ut till systemet. Även högtalaren som producerar ljudvågen kommer att stå för en del av skillnaden mellan referenssignalen och inspelningen.

6.2 Etiska aspekter

När det handlar om att spela in ljud kommer naturligt frågan om etik. Det handlar framförallt om aspekter hur de inspelade ljuden sparas och används. Det är viktigt att användare av systemet är medvetna om hur inspelningen fungerar och vad den används till. Indikation när mikrofonen spelar in skulle kunna öka medvetenheten av att en inspelning pågår. Systemet sparar endast små delar av den totala inspelningen som matchar en referens. Detta gör att det är svårt att få en kontext av något som spelas in av misstag eftersom de bitarna inte kommer sparas eller matchas mot en referens.

En annan etisk aspekt är att systemet avser att ersätta den mänskliga hörseln vid ljudtester. Ur ett företags synvinkel skulle det kunna vara en besparing på lång sikt, men det innebär samtidigt att människan som idag utför testerna manuellt på sikt förlorar arbetstimmar eller i värsta fall blir helt överflödig. Däremot flyttas ansvaret för godkänna ljud från människan till systemet vilket skulle kunna minska en persons stressnivåer då den inte längre behöver avgöra om något är rätt eller fel.

Systemet skulle kunna bidra till ett mer varierande arbete där man inte behöver utsättas för att ständigt utsättas för olika ljud.

Aspekterna som lyfts upp är spekulativa. Det finns inte belägg eller bevis för fallen ovan. I slutändan är det upp till gruppen eller företaget som väljer att implementera systemet att beakta de etiska aspekterna.

6.3 Miljöaspekter

Miljö är något som är högst aktuellt och påverkar alla företag och system. Generellt skulle vårt system ha en negativ miljöpåverkan i form av ökad energiförbrukning för hårdvaran som systemet körs på. Vi bedömer dock att systemet skulle ha en minimal miljöpåverkan ur företagets eller användarens totala perspektiv.

7

Slutsats och framtida rekommendationer

Det här kapitlet presenterar en slutsats och några förslag på områden där fortsatt arbete skulle kunna ta vid.

Rapportens syfte var att utforska möjligheter att med ett datorsystem ersätta den mänskliga hörseln vid testning av återkopplingsljud. Resultatet påvisar att ett sådant system faktiskt kan prestera bättre än en människa när det kommer till rent objektiva aspekter av ljud så som timing och volym. Däremot är det svårare för systemet att ersätta människans förmåga att identifiera små missljud i en ljudvåg då detta kräver en tydligare definition av vad som skall tolkas som missljud och ej. En del av problematiken i det ligger i att människans hörsel är olika känslig i olika frekvensområden vilket leder till att förändringar i en signal som ett datorsystem anser vara signifikant kan gå obemärkt av människan och vice versa.

7.1 Förslag för vidarearbete

- Ett intressant alternativ för att ta reda på "vad är ett missljud?" skulle vara att genomföra en omfattande studie där flera personer får lyssna på olika ljud och ange när de upplever en skillnad mot referensljudet.

- När man tydligt har definierat vad som skall tolkas som missljud behöver man utveckla en metod kalibrerad efter denna definition som kan identifiera detta.

- Man kunde med fördel använda flera referenserinspelningar med olika inspelningsförhållanden till ljuden. På så sätt kunde inverkan av mikrofonens placering blir mindre.

Litteratur

[1] D. G. S. Richard E Berg, *Physics of Sound.* Benjamin Cummings, 2004, s. 8.

[2] J. G. M. Thewissen, *Sensory Evolution on the Threshold: Adaptations in Secondarily Aquatic Vertebrates.* University of California Press Books, 2008.

[3] H. Nicklasson, *Jakten på det perfekta PA-ljudet, Handbok i ljudteknik.* HN Ljuddesign, 2006.

[4] U. Yavanoglu, B. Ozcakmak och O. Milletsever, "A New Intelligent Steganalysis Method for Waveform Audio Files," i *2012 11th International Conference on Machine Learning and Applications*, vol. 2, 2012, s. 233–239. DOI: 10.1109/ICMLA.2012.150. URL: https://ieeexplore.ieee.org/document/6406756 (hämtad 2021-04-12).

[5] "WAVE PCM soundfile format," URL: http://soundfile.sapp.org/doc/WaveFormat/ (hämtad 2021-04-12).

[6] A. Wang m. fl., "An industrial strength audio search algorithm.," i *Ismir*, Citeseer, vol. 2003, 2003, s. 7–13. URL: https://www.ee.columbia.edu/~dpwe/papers/Wang03-shazam.pdf (hämtad 2021-04-12).

[7] M. O. Derbasov, V. E. Gai och P. A. Shagalova, "A Hierarchical Method of Forming Fingerprints of a Sound Signal," i *2018 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon)*, 2018, s. 1–4. DOI: 10.1109/FarEastCon.2018.8602783. URL: https://ieeexplore.ieee.org/document/8602783 (hämtad 2021-04-15).

[8] D. W. Boyd, *Systems Analysis and Modelingn.* 2001. DOI: 10.1016/B978-0-12-121851-5.X5000-X. URL: https://www.sciencedirect.com/topics/earth-and-planetary-sciences/cross-correlation (hämtad 2021-04-21).

[9] "Correlation Coefficients," URL: https://www.andrews.edu/~calkins/math/edrm611/edrm05.htm (hämtad 2021-04-30).

[10] X. Ming och D. Kang, "Corrections for frequency, amplitude and phase in a fast Fourier transform of a harmonic signal," *Mechanical Systems and Signal Processing*, årg. 10, nr 2, s. 211–221, 1996. URL: https://www.sciencedirect.com/science/article/abs/pii/S0888327096900151 (hämtad 2021-04-12).

[11] J. Hrisko. (2018). "Audio Processing in Python Part I: Sampling, Nyquist, and the Fast Fourier Transform," URL: https://makersportal.com/blog/2018/9/13/audio-processing-in-python-part-i-sampling-and-the-fast-fourier-transform (hämtad 2021-04-27).

[12] "Visual Studio Code official website," URL: https://code.visualstudio.com/ (hämtad 2021-06-01).

[13] "Musicg project," URL: https://code.google.com/archive/p/musicg/ (hämtad 2021-04-12).

[14] "MATLAB official website," URL: https://se.mathworks.com/products/matlab.html (hämtad 2021-05-12).

[15] "Thomann web store, Superlux ECM999," URL: https://www.thomann.de/se/superlux_ecm999.htm.

[16] "Thomann web store, TC-Helicon Go Solo. Accessed May. 3, 2021," URL: https://www.thomann.de/se/tc_helicon_go_solo.htm (hämtad 2021-05-03).

[17] "Microsoft Planner official website," URL: https://www.microsoft.com/en-us/microsoft-365/business/task-management-software (hämtad 2021-04-30).

A

Bilagor

A.1 JUnit klasstester

```
public class SoundAnalyzerTest
{
    SoundAnalyzer sa;
    Audio a;
    ArrayList<SoundCrab> arr;
    ArrayList<SoundCrabWave> refArr;
    @Before
    public void init(){
        sa = new SoundAnalyzer();
        a = Audio.getInstance();
        refArr = new ArrayList<>();
        refArr.add(a.rippedTill);
        refArr.add(a.rippedFran);
        arr = a.rippedBlinkersComplete.edit.findCrabsXCorr(refArr);
    }

    @Test
    public void compareFingerprintTest(){
        float f = sa.compareFingerprint(a.rippedBlinkersComplete.
   clone(), a.rippedBlinkersComplete.clone());
        assertEquals(100, f, 0.0);
    }

    @Test
    public void correlationCoefficientCheckTest(){
        sa.correlationCoefficientCheck(arr);
        assertEquals(0.99, arr.get(1).corrCoef, 0.01);
    }

    @Test
    public void calculateRmsTest(){
        sa.calculateRms(arr);
        assertEquals(53.6, arr.get(2).rms, 0.1);
    }

    @Test
    public void findReferenceTest(){
        assertEquals("Blinkers", sa.findReference(a.
   rippedBlinkersComplete));
    }

```

```
40      @Test
41      public void frequencyCoefficientCheckTest(){
42          ArrayList<SoundCrab> list = new ArrayList<>();
43          for (SoundCrab s : arr) {
44              list.add(s.referenceCrab);
45          }
46          sa.getSpectrum(list);
47          sa.getSpectrum(arr);
48          sa.frequencyCoefficientCheck(arr);
49          assertEquals(0.9995044518661262, arr.get(2).freqCoef, 0.0);
50      }
51
52      @Test
53      public void timingCheckTest(){
54          sa.timingCheck(arr);
55          assertEquals(35280, arr.get(29).preTimeGap);
56      }
57  }
```

```
1   public class SoundCrabWaveTest {
2       @Test
3       public void getPeakIndexTest(){
4           int peakIndex = 0;
5           byte[] data = {0, 0, 1, 1, 3, 3, 2, 2, 0, 0};
6           SoundCrabWave wave = new SoundCrabWave(new WaveHeader(),
        data);
7           peakIndex = wave.getPeakIndex();
8           assertEquals(2, peakIndex);
9       }
10
11      @Test
12      public void getNegativeIndexTest() {
13          byte[] data = {0, 0, 1, 1, -3, -3, 2, 2, 0, 0};
14          SoundCrabWave wave = new SoundCrabWave(new WaveHeader(),
        data);
15          assertEquals(2, wave.getNegativePeakIndex());
16      }
17  }
```

```
1   public class SoundEditorTest {
2       SoundAnalyzer sa;
3       Audio a;
4       ArrayList<SoundCrab> arr;
5       ArrayList<SoundCrabWave> refArr;
6
7       @Before
8       public void init(){
9           sa = new SoundAnalyzer();
10          a = Audio.getInstance();
11          refArr = new ArrayList<>();
12          refArr.add(a.rippedTill);
13          refArr.add(a.rippedFran);
14      }
15
16      @Test
17      public void findCrabsXCorrTest(){
```

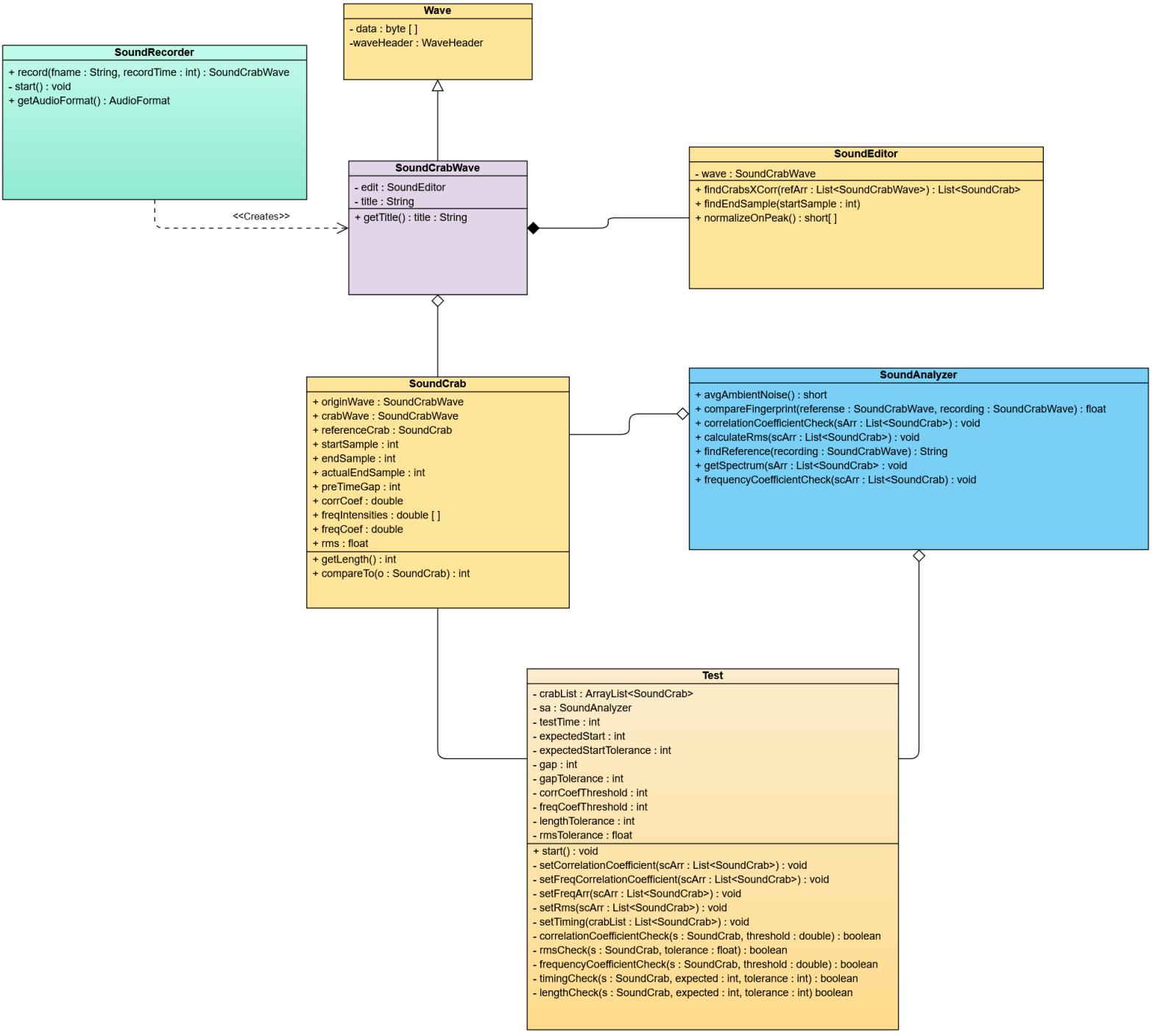
```
        arr = a.rippedBlinkersComplete.edit.findCrabsXCorr(refArr);
        assertEquals(32, arr.size());
    }

    @Test
    public void findEndSampleTest(){
        int end = a.rippedTill.edit.findEndSample(0);
        assertEquals(412, end);
    }

    @Test
    public void normalizeOnPeakTest(){
        short [] norm = a.rippedTill.edit.normalizeOnPeak();
        assertEquals(Short.MAX_VALUE, norm[a.rippedTill.
    getPeakIndex()]);
    }
}
```

```
public class SoundRecorderTest {
    @Test
    public void recordTest(){
        //Record 2 sec
        Wave wav = SoundRecorder.getInstance().record("TEST2",
    2000);
        //File exists
        assertTrue(wav!=null);
        //File samplelength is bigger than the set samplerate (
    since recording time is > 1 sec);
        assertTrue(wav.getSampleAmplitudes().length > SoundRecorder
    .getInstance().getAudioFormat().getSampleRate());
    }
}
```

A.2 UML - Klassdiagram



INSTITUTIONEN FÖR DATA- OCH INFORMATIONSTEKNIK
CHALMERS TEKNISKA HÖGSKOLA
GÖTEBORGS UNIVERSITET
Göteborg, Sverige 2021
www.chalmers.se

CHALMERS