Auditiv översättning av morse

Casper Larsson, Viktor Larsson, MT3b, 10 oktober 2021

1. Inledning

Morse byggs upp på ett sätt att skicka information med hjälp av en telegraf, som kan skicka korta och långa signaler. Alfabetet är uppbyggt av kombinationer av de korta och långa signalerna som skickas sekventiellt. För att kunna översätta en morsekod auditivt krävs mycket övning för att göra det snabbt. Syftet med det här projektet är att låta en dator göra översättningen istället med endast en godtycklig ljudsignal som indata.

2. Metod och material

Till projektet användes MATLAB för dess inbyggda funktioner att hantera ljudsignaler och möjlighet att analysera den samt dess frekvensområden.

2.1 Signalbehandling

Innan ljudsignalen översattes krävdes omvandling för att skapa en ideell signal. Eftersom ljudet upptogs via mikrofonen tillkommer visst brus och andra störningar som i största mån filtrerades bort med hjälp av MATLABs inbyggda funktion *bandpass*, som endast sparar det frekvensområdet som morsekoden skickas i. Brus kunde fortfarande förekomma vilket togs bort genom att ljudet som var lägre än ett visst tröskelvärde ändrades till noll, det vill säga helt tyst. Därefter kunde signalen analyseras och en counter användes, som ökade värdet varje gång den var nollskild. Detta skapade en vektor, kallad *occurrence*, som innehöll längden av alla korta och långa signaler som sedan kunde jämföras.

2.2 Antaganden och begränsningar

För att kunna tillämpa översättningen till en godtycklig ljudsignal krävdes ett antal antaganden om ljudet. En generell morsesignal kan använda olika hastigheter för att skicka varje signal, och behöver inte nödvändigtvis vara samma varje exekvering. Det första antagandet som gjordes var att ta fram skillnaden mellan korta och långa ljudsignaler som skickades. Detta gjordes genom att ta det högsta värdet i *occurrence* och definiera det som en lång signal. En kort signal definierades sedan som en tredjedel av en lång. Mellanrummet mellan bokstäver bestämdes sedan som fyra stycken korta samt mellanrummet mellan ord sattes som åtta stycken korta signaler.

Ett annat antagande som krävdes för att utföra filtreringen av ljudet, var vilket frekvensområde som morsekoden skickades i. Det finns ingen fast definition på vilken frekvens signalen använder, utan beror från fall till fall. För att ta reda på frekvensen kan antingen ljudkällans parametrar undersökas. Det är också möjligt att utföra en frekvensanalys där det dominerande frekvensomfånget senare sparas. Det senare alternativet kräver att just morsesignalen är det dominerande innehållet i ljudet. Till detta projekt kunde frekvensen väljas fritt uppspelningsprogrammet|1|, och en senare frekvensanalys visade att denna parameter stämde.

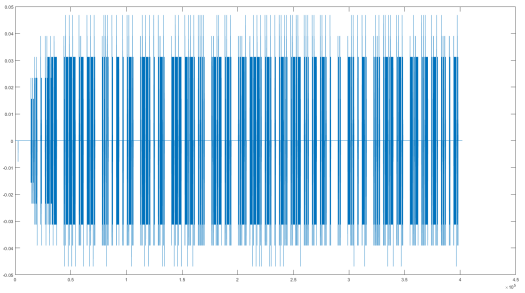
2.3 Översättningen

Vektorn *occurrence* användes sedan till översättningen, där varje värde jämfördes med definitionerna av lång och kort signal. En procentuell osäkerhet användes som ett intervall, då alla långa och korta förekomster

inte är exakt likadana. Programmet skapade därefter en textrad av punkter och bindestreck (. -) som representerade morsekoden. Denna textrad jämfördes bokstav för bokstav mot en databas |2| för att få det slutgiltiga resultatet.

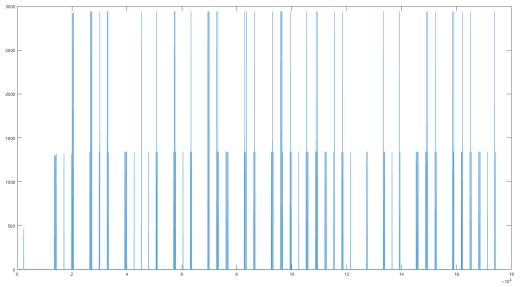
3. Resultat

Följande resultat behandlar meningen *"hej jag heter Per och är föreläsare i den här kursen"* som spelas upp på en enhet och tas upp via MATLAB på en annan. Den inspelade ljudsignalen visualiseras i Figur 1 där det tydliggörs en skillnad mellan långa och korta signaler, samt de olika mellanrummen mellan bokstäver och meningar.



Figur 1. Den inspelade signalen

I Figur 2 visas vektorn *occurrence* som innehåller toppar över de olika längderna i signalen. De olika långa signalerna sammanfaller och skiljer sig från varandra endast med en liten marginal, likadant för de korta signalerna. Vektorn stämmer tydligt överens med insignalen i Figur 1.



Figur 2. Vektorn *occurrence*

När programmet läst av *occurrence* skapades följande sträng som morsekod:

".... . .--- space .--- .- --. space .... . - . .-. space .--. . .-. space --- -.-. .... space .-.- .-. space ..-. ---. .-. . .-.. .-.- ... .- .-. . space .. space -.. . -. space .... .-.- .-. space -.- ..- .-. ... . -."

Mellanrummen definierar uppdelningen mellan bokstäver och *"space"* visar var ett nytt ord börjar. De övriga punkter och bindestreck är de avkodade bokstäverna. När denna sträng jämförts med databasen returnerades den sökta meningen *"hej jag heter Per och är föreläsare i den här kursen"*.

4. Diskussion

Syftet med projektet var att låta en dator översätta morse till text endast från en godtycklig ljudsignal som data. Detta lyckades dock med lite varierande resultat, vilket kommer diskuteras nedan.

Första problemet uppstår med att ta in ljud genom mikrofonen, då det alltid kommer finnas något brus i bakgrunden. Om detta brus skulle vara för högt samt inom samma frekvensområde som den inskickade signalen kan resultatet bli fel då programmet tror att bruset är en del av signalen. Dock är största delen av brus är väldigt låga frekvenser vilket leder till att det oftast inte påverkar resultatet.

Eftersom programmet definierar om det är en kort respektive lång signal genom att jämföra längden på den längsta signalen fungerar inte programmet om ljudet endast är uppbyggd av korta signaler. Då kommer resultatet bara bli bindestreck, eftersom programmet först bestämmer den längsta signalen i vektorn *occurrence* som en lång signal. Om då alla signaler är korta kommer dom räknas som långa. Programmet valdes att programmeras på detta sätt då en kort och lång signal inte har en bestämd längd, utan kan variera beroende på hur snabbt orden vill skrivas. Metoden valdes då det bara finns några få bokstäver som endast innehåller korta signaler, och det är sällan ett ord eller mening innehåller enbart sådana bokstäver.

Resultatet kunde variera mellan olika datorer då det är olika mikrofoner på datorerna. Med vissa mikrofoner fungerade programmet som tänkt medan andra mikrofoner tar upp ett för stort brus vid varje signal

Ett annat problem med programmet är att det inte nödvändigtvis skulle fungera att en människa matar in signalen med en telegraf. Detta eftersom de långa och korta signalerna skulle variera för mycket i längd. En signal skickas genom att trycka på en knapp. Eftersom en människa inte kan trycka med exakt samma längd varje gång kommer programmet möjligen kunna klassificera en kort signal som en lång signal och vice versa. Dock skulle osäkerheten i programmet hjälpa att minimera påverkan av den mänskliga faktorn.

Programmet har även problem om hastigheten på morsekoden blir för snabb då långa och korta signaler blir för lika. Detta uppstår då det finns en osäkerhet för långa och korta signaler, och när långa och korta signalerna blir för lika kommer programmet inte kunna veta vad som är en lång och vad som är en kort signal.

5. Slutsatser

Utifrån detta projekt visar det att det är fullt möjligt att översätta morsekod med hjälp av en dator om än med några komplikationer. Slutresultatet gör det möjligt att överföra ett antal olika meningar med korrekt översättning under ideala förhållanden. Vidarearbetet skulle vara att filtrera insignalen från ytterligare aspekter för att i flera fall skapa dessa ideala förhållanden.

Referenser

|1| Burak Özdemir. *Morse Code Translator - Morse Decoder*. https://morsedecoder.com/ (Hämtad 2021-09-28)
|2| Wikipedia. 2021. *Morse code*. https://en.wikipedia.org/wiki/Morse_code (Hämtad 2021-09-09).