

## Epic-Arpeggiator

Syftet med detta projekt var att undersöka och tillämpa grundläggande ljudfysikprinciper för att manipulera ljudvågor och generera olika ljudeffekter. För att uppnå detta utvecklades en webb-app som möjliggör för användare att skapa, modifiera och omvandla ljudvågor för att producera effekter som under- och övertoner, vibrato, tremolo samt ekon. En central funktion i appen är en arpeggiator, som bygger vidare på dessa begrepp genom att skapa melodiska ljud från en enskild ton. Applikationen erbjuder dessutom en visuell representation av hur dessa effekter påverkar ljudvågens form, vilket gör den till ett värdefullt verktyg både för inlärning och kreativt skapande inom ljuddesign och musikproduktion.

## Teori & implementation

För att skapa effekterna måste vi först tolka en ren sinusvågfunktion. En ren sinusvåg kan matematiskt uttryckas som  $A \sin(\phi(t))$ , där  $A$  representerar amplituden, och  $\phi(t)$  är fasfunktionen, som bestämmer frekvensen och formen på vågen. Frekvensen  $f_0$  (som motsvarar hur snabbt vågen oscillerar) representeras inom fasfunktionen, som vanligtvis definieras som  $\phi(t) = 2\pi f_0 t$ . Med hjälp av Web Audio API kan vi skapa och spela upp ljudvågor, med möjlighet att kontrollera bland annat amplitud, frekvens och vågtyp.

För att få ett mer intressant ljud används en kuvertgenerator. En kuvertgenerator kontrollerar hur ljudet påverkas över tid. Generatoren kontrolleras med hjälp av fyra parametrar: 'attack', som bestämmer hur lång tid det tar för tonen att gå från noll till sitt maxvärde, 'decay', vilket anger hur snabbt ljudet sjunker från maxvärdet till 'sustain', och 'release', som bestämmer hur lång tid ljudet tar att klinga ut efter att tonen avslutats." Genom att ändra på parametrarna kan flera olika ljud skapas, ett högt värde på 'attack' och 'release' kan leda till ett mer mystiskt ljud och låga värden på 'decay' och 'sustain' kan leda till ett 'pluck' ljud.

## Grundton, överton och underton

Grundtonen är den lägsta frekvensen i en ljudvåg och utgör basen för ljudets tonhöjd. Den bestämmer vilken ton vi uppfattar, och i vårt projekt representeras grundtonen av användarens inställda frekvensvärde. I vår applikation kan exempelvis grundton vara inställd på 440 Hz.

Övertoner, som ligger ovanför grundtonen i frekvensspektrumet, bidrar till att forma ljudets klangfärg. Övertonerna är harmoniska multipler av grundtonens frekvens och tillför ljudet en större komplexitet. Genom att addera övertoner skapas ett rikare och mer mångfacetterat ljud. I detta projekt tillhandahålls en funktion för att generera övertoner genom knappen "Add Harmonics", vilket gör det möjligt för användaren att skapa en harmonisk överlagring av frekvenser. För exempelvis en grundton på 440 Hz kan övertoner genereras vid frekvenser som 880 Hz och 1320 Hz, vilket leder till ett mer komplext och nyanserat ljud.

Undertoner utgör motsatsen till övertoner och återfinns under grundtonens frekvens. Även om undertoner är mindre vanliga inom traditionell musikteori, kan de användas för att skapa ett ljud med mörkare och djupare klang. I projektet finns en funktion för att addera undertoner via knappen "Add Undertones". Exempelvis kan en grundton på 440 Hz ha

TFYA65 projektrapport  
Karl Kostoulas (karko847), Deema Abo Gheda(deeso509),  
Alexis Vargas (mator905), Linus Jansson (linja782)

undertoner vid frekvenser som 220 Hz eller 110 Hz, vilket bidrar till en mer komplex ljudbild med både högre och lägre frekvenser.

Genom att kombinera grundtonen med över- och undertoner skapas en mer mångsidig och fyllig ljudkaraktär

### **Vibrato**

För att introducera vibratoeffekten behöver vi modifiera vågens fasfunktion för att skapa en lätt ondulering i tonhöjd. Detta görs genom att lägga till en sinusoidal komponent som fluktuerar runt den grundläggande frekvensen. Specifikt inför vi en frekvensavvikelse  $\{\Delta F\}$  som indikerar djupet av vibraton (Vibrato Gain i appen) och en vibratofrekvens  $f_R$  som dikterar hur snabbt dessa fluktuationer sker (Vibrato Frequency i appen). Således blir den nya fasfunktionen:

$$\phi(t) = 2\pi f_0 t + \Delta F \sin(2\pi f_R t)$$

Genom att sätta tillbaka denna modifierade fasfunktion i den ursprungliga oscillatorns ekvation kan vi producera ett ljud som uppvisar den karakteristiska vibratoeffekten.

Effektivt vibrato sker vanligtvis vid en frekvens på omkring 4 till 5 Hz; frekvenser som är betydligt högre än detta, särskilt när de kombineras med ett stort djup, kan leda till en effekt som avviker från traditionellt vibrato och kan låta mer som tremolo eller andra modulerings effekter. För vårt syfte är lärandet vi tillåter användaren att sätta vibratofrekvens och frekvensavvikelse värdena för att se/höra i realtid hur dessa påverkar ljudens egenskaper.

### **Tremolo**

Tremolo till skillnad från vibrato är lågfrekvent amplitudmodulering (AM) istället för frekvensmodulering. Frekvensen påverkas inte medan amplituden ändras med en satt frekvens. Tremolo kan beskrivas med  $e(t) = 1 - D + D \sin(2\pi f_R t)$  där 'D' bestämmer djupet på effekten, även kallad "depth". 'F' är frekvensen med vilket ljudet ska ändras med och '1-D' läggs till för att undvika att ljudet klipper. Effektens frekvens är vanligtvis mellan 3 Hz och 10 Hz.

### **Eko**

Eko skapas genom att repetera ljud med lägre gain en stund efter att original ljudet spelas. Funktionen för ljudet med eko kan skrivas som  $y(t) = x(t) + a y(t-D)$ , där 'y' är ljudutmätningen, 'x' är ljudinmätningen, 'a' beskriver hur starkt det studsande ljudet är och 'D' är hur lång tid det tar för att ljudet ska studsas tillbaka. När arpeggiatorn spelas sätts 'D' värdet till samma tid som är mellan arpeggiatorns notuppspelningar för att få en mer behaglig kombination mellan arpeggiatorn och ekot.

### **Arpeggiator**

Arpeggios är ett sätt att spela toner i en order ton för ton. Arpeggios kan hjälpa med ackordsförlopp, bygga en mer intressant rytm och att lägga till melodisk ornamentik. Arpeggios kan spelas för hand eller med hjälp av en arpeggiator. En arpeggiator är ett verktyg inom musik som spelar arpeggios automatiskt. Det här projektet implementerade en arpeggiator genom att lyssna på tangentbordet och spela respektive ton med konstant mellanrum. Arpeggiatorn har inställningar som 'play mode' vilket bestämmer vilken ordning som tonerna spelas upp. 'Up' går från lägsta ton till högsta och sedan hoppar tillbaka till

TFYA65 projektrapport  
Karl Kostoulas (karko847), Deema Abo Gheda(deeso509),  
Alexis Vargas (mator905), Linus Jansson (linja782)

lägsta igen när den nått slutet. 'Down' fungerar som 'Up' men baklänges och 'Up-Down' klättrar från lägsta till högsta ton och sedan klättrar tillbaka ner. 'Tempo' bestämmer hur lång tid det är mellan spelningarna av tonerna. 'Gate' bestämmer hur länge tonerna spelas. Ett gate-värde på ett spelar en ton som varar under hela fördröjningstiden mellan tonerna, medan ett värde på en halv spelar en ton under hälften av denna tid. 'Octave' bestämmer hur många gånger tonerna repeteras i nästa oktav. Det fungerar genom att spela tonen frekvens multiplicerat med två upphöjt med det nuvarande oktav värdet. De här enkla funktionerna tillåter användaren att skapa och modifiera intressanta ackord som uttrycker olika känslor.

### **Diskussion och Resultat :**

I vårt projekt, har vi utvecklat en webbapplikation för att skapa och manipulera ljudeffekter baserade på ljudfysik. De implementerade effekterna övertoner, undertoner, vibrato, tremolo och eko fungerade i stort sett som förväntat, men det finns några reflektioner kring förbättringspotential. Våra övertoner och undertoner skapade en rikare och mer mångfacetterad ljudbild, vilket gav användarna en djupare ljudupplevelse. Effekterna stämde bra överens med de teoretiska förväntningarna, även om vissa finjusteringar i amplitud kunde ha förbättrat ljudbalansen vid vissa inställningar.

Vibratoeffekten gav en naturlig variation i tonhöjd och fungerade väl med balanserade inställningar, vilket bidrog till en realistisk ljudvariation. Tremoloeffekten, som tillförde en regelbunden variation i ljudets amplitud, fungerade också bra, men vid högre inställningar blev effekten ibland för stark, vilket påminde mer om en störning än en musikalisk modulering. Ekoeffekten skapade den förväntade illusionen av rymd och djup, särskilt i större simuleringar, men vi identifierade ett förbättringsområde i att implementera en mer avancerad algoritm för att hantera eko vid snabbare uppspelningar för att undvika överlappande ljud.

Om vi skulle genomföra projektet igen, hade vi valt att ändra fokus på hur användare interagerar med de första effekterna. Istället för att användaren bara justerar parametrar som frekvens eller amplitud genom kontroller, skulle vi låta användare ändra själva vågekvationen direkt. Detta skulle ge användaren möjlighet att experimentera mer fritt och på ett mer interaktivt sätt, samtidigt som de lär sig hur varje del av ekvationen påverkar ljudet. Detta skulle förbättra förståelsen av de grundläggande principerna bakom ljudmanipulering.

Slutsatsen av vårt projekt visar hur viktig förståelsen av ljudfysik är för att skapa och manipulera ljud. Genom att använda matematiska modeller som sinusvågor och tillämpa effekter som övertoner, undertoner, vibrato och eko, har vi fått insikt i hur frekvens, amplitud och fas påverkar ljudet.