**Universitatea Tehnică din Cluj-Napoca**

**Facultatea de Electronică, Telecomunicații și Tehnologia Informației**

**ASRSV**

**Sintetizator virtual implementat cu Python**

Profesor coordonator: Student:

prof. dr. ing. Eugen Lupu Jimon Lucian-Daniel

2024

**Cuprins**

[Introducere 3](#_Toc156224106)

[Despre proiect 3](#_Toc156224107)

[Problema 3](#_Toc156224108)

[Soluția 3](#_Toc156224109)

[Tehnici de programare Python 3](#_Toc156224110)

[Fundamentare teoretică 4](#_Toc156224111)

[Sinteza digitală a sunetelor 4](#_Toc156224112)

[Generarea de forme de undă 4](#_Toc156224113)

[Notele Muzicale 4](#_Toc156224114)

[Funcționarea Sintetizatoarelor 4](#_Toc156224115)

[Imitarea Instrumentelor Acustice 5](#_Toc156224116)

[Tehnici Avansate de Sinteză 5](#_Toc156224117)

[Anvelope ADSR 5](#_Toc156224118)

[Aplicații Practice în Muzică și Design de Sunet 6](#_Toc156224119)

[Importanța Interfețelor Utilizator 6](#_Toc156224120)

[Viitorul Sintezei Sunetului 6](#_Toc156224121)

[Implementare 7](#_Toc156224122)

[Backend 7](#_Toc156224123)

[GUI 9](#_Toc156224124)

[Rezultate 11](#_Toc156224125)

[Concluzie și posibile îmbunătățiri 12](#_Toc156224126)

[Bibliografie 13](#_Toc156224127)

# Introducere

## Despre proiect

Prezentul proiect intitulat „Sintetizator virtual implementat cu Python” a fost structurat și dezvoltat ca parte a materiei „Analiza, sinteza și recunoașterea semnalului vocal”. Proiectul utilizează mai multe aspecte teoretice și de programare ce sunt utilizate în domeniul strâns legat al materiei, acela de analiză și recunoaștere vocală, însă diferă în motivul principal, care în cazul de față este generare de sunete muzicale, și nu vocale.

Scopul proiectului a fost dezvoltarea unui software care, bazat pe tehnici de procesare și generare de sunet, poate simula o octavă dintr-un sintetizator (orgă) digitală, sunetul căreia poate fi modificat de utilizator.

### Problema

Problema principală a obiectivului proiectului a fost găsirea unei soluții de a putea genera mai multe sunete în același timp, pentru a reda senzația de orgă reală, cât de mult se poate. Acest aspect a fost tratat prin strategia de multi-threading, astfel procesorul poate genera notele muzicale în mod paralel. O altă

### Soluția

După cum sugerează titlul lucrării, limbajul de programare folosit a fost Python, care, printre alte beneficii extraordinare, vine cu o utilizare extinsă a diferitelor biblioteci și funcții predefinite sau open-source, în special în domeniul științei datelor. Astfel, Python a fost ales ca instrument principal al proiectului, și module specifice analizelor și sintezelor de date ca numpy.

Proiectul combină în mod unic sinteza digitală a sunetului, anvelopa ADSR, efectele de întârziere și elementele interactive ale interfeței grafice pentru o experiență de utilizare îmbucurătoare..

În cele din urmă, deoarece scripturile au fost destinate să fie utilizate de utilizatori umani, GUI-ul pentru proiect a fost realizat cu Tkinter, care este biblioteca standard GUI pentru Python.

## Tehnici de programare Python

Python este unul dintre cele mai populare limbaje de programare, fiind un limbaj rapid, ușor de înțeles și accesibil, cu mii de biblioteci disponibile, făcându-l cel mai folosit limbaj de programare din câțiva ani (la ora actuală Python este încă cel mai folosit limbaj de programare și a câștigat premiul „Cel mai bun limbaj de programare 2021” [1] ).

Una dintre principalele utilizări ale Python este în știința datelor, iar acest lucru poate fi dovedit de multitudinea de biblioteci de învățare automată, simplitatea utilizării lor în scripturi și ușurința sintaxei limbajului, făcându-l perfect pentru scopuri precum acest proiect. .

# Fundamentare teoretică

## Sinteza digitală a sunetelor

Sinteza digitală a sunetelor este un proces complex și diversificat care implică crearea de sunete prin metode electronice, folosind algoritmi de calculator. Această tehnică, care a evoluat considerabil de la primele sale zile, atunci când sintetizatoarele analogice erau standardul, a cunoscut o evoluție remarcabilă în era digitală. Sintetizatoarele moderne pot fi fie hardware, fie software, aducând o nouă putere și flexibilitate în tehnologia digitală, care reflectă progresele semnificative în tehnologia audio și informatică [2].Ele permit producătorilor de muzică și artiștilor să creeze sunete care altfel ar fi dificil sau imposibil de generat prin metode acustice tradiționale.

## Generarea de forme de undă

În centrul sintezei sunetului stă generarea formelor de undă. O formă de undă este o reprezentare grafică a variațiilor de presiune sonoră sau a semnalului electric care produce sunetul. Sintetizatoarele digitale încep procesul de sinteză cu un oscilator care generează o frecvență sonoră sub forma unei forme de undă. Există mai multe tipuri de forme de undă utilizate în sinteza digitală, inclusiv undele sinusoidale, pătrate și triunghiulare.

* Unda Sinusoidală: Este cea mai simplă formă de undă și produce un ton pur și clar. Datorită naturii sale netede, este adesea utilizată ca bază pentru alte forme de undă mai complexe.
* Unda Pătrată: Are un caracter distinct, producând un ton mai aspru și mai bogat în armonice decât unda sinusoidală. Este frecvent folosită în muzica electronică pentru a crea sunete de bas sau lead-uri care se remarcă în mix.
* Unda Triunghiulară: Ocupă un loc intermediar între unda sinusoidală și cea pătrată, având un caracter mai moale decât unda pătrată, dar mai bogat în armonice decât unda sinusoidală.

## Notele Muzicale

Notele muzicale sunt esențiale în muzică, fiecare notă reprezentând o frecvență sonoră specifică. De exemplu, nota "La" standard (A) are o frecvență de 440 Hz. În sintetizatoare, fiecare notă este generată prin ajustarea frecvenței oscilatorului pentru a produce frecvența corespunzătoare notei. Acest lucru permite sintetizatoarelor să imite sunetele instrumentelor acustice sau să creeze sunete complet noi și unice [3].

## Funcționarea Sintetizatoarelor

Sintetizatoarele muzicale funcționează prin generarea și manipularea electronică a undelor sonore, oferind o platformă versatilă pentru crearea unei game largi de sunete. Iată o prezentare generală a modului în care funcționează sintetizatoarele:

Inima unui sintetizator este oscilatorul. Acesta produce unde sonore brute, nefiltrate, cum ar fi undele sonore în formă de ferăstrău, pătrate, sinusoidale, triunghiulare sau zgomot. În sintetizatoarele digitale, oscilatoarele pot fi tabele de unde sau chiar mostre înregistrate. Tonalitatea acestor unde este adesea modificată de tensiuni de control, care sunt generate de o tastatură sau de un secvențiator.

După oscilator, sunetul călătorește către o secțiune de filtrare (sau VCF), unde sunt modelate calitățile sale armonice. Există diferite tipuri de filtre, inclusiv filtre trece-jos, trece-înalt, trece-bandă și filtre crestătoare. Fiecare tip permite în mod selectiv trecerea anumitor frecvențe și blochează altele, modelând astfel timbrul sunetului.

Sunetul ajunge apoi la un amplificator (VCA), care îi determină volumul. O anvelopă atașată la amplificator controlează modificarea volumului în timp folosind patru etape: Attack, Decay, Sustain și Release (ADSR). Aceste etape definesc caracterul sunetului pe măsură ce evoluează din momentul în care este cântată o notă până când aceasta se estompează.

## Imitarea Instrumentelor Acustice

Unul dintre cele mai fascinante aspecte ale sintetizatoarelor este capacitatea lor de a imita sunetele instrumentelor acustice tradiționale, cum ar fi pianul, vioara sau flautul. Acest lucru se realizează prin modelarea atentă a caracteristicilor specifice ale sunetelor acestor instrumente, cum ar fi atacul, decăderea, menținerea și eliberarea sunetului, precum și prin simularea rezonanței și altor proprietăți acustice.

## Tehnici Avansate de Sinteză

Pe lângă formele de undă de bază și imitarea instrumentelor acustice, sintetizatoarele digitale oferă și alte tehnici avansate de sinteză. Printre acestea se numără:

* Sinteza Aditivă: Această tehnică modelează un spectru sonor ca un set de linii discrete corespunzătoare sinusoidelor. Prin combinarea diferitelor frecvențe și amplitudini ale acestor componente, se poate crea o gamă largă de sunete
* Sinteza substractivă: Acesta este un tip comun de sinteză a sunetului în sintetizatoarele analogice, în care părțile parțiale ale unui semnal audio bogat în armonici sunt atenuate de un filtru pentru a modifica timbrul sunetului. Se pornește de la o formă de undă complexă, care este apoi modelată prin filtre și anvelope.
* Sinteza FM (Modulare în Frecvență): Utilizează două oscilatoare, unul pentru generarea unei frecvențe originale și celălalt pentru modularea primului prin introducerea armonicelor suplimentare. Această tehnică a fost popularizată de sintetizatorul Yamaha DX7 și este cunoscută pentru sunetele sale metalice și vibrante etc.

## Anvelope ADSR

Un aspect esențial al sintezei sunetului este utilizarea anvelopelor ADSR (Attack, Decay, Sustain, Release) pentru modelarea dinamicii sunetelor. Anvelopa ADSR definește modul în care intensitatea unui sunet evoluează de-a lungul timpului, de la momentul inițierii (atac) până la sfârșitul său (eliberare). Faza de „Atac” determină rapiditatea cu care sunetul atinge punctul său maxim după ce o notă este apăsată, urmată de „Decay” care descrie ritmul de scădere a sunetului la nivelul „Sustain”, care este intensitatea constantă a sunetului până la eliberarea notei. În cele din urmă, faza de „Release” descrie cum sunetul se estompează după ce nota nu mai este apăsată. Aceste faze se pot observa în următoarea figură, preluata de la [4].  
Acest mecanism permite artiștilor să controleze expresivitatea și forma sunetului, de la sunete abrupte și percutive, la tonuri lungi și evolutive, fiind un instrument esențial în designul sonor și sinteza muzicală (Valimaki & Takala, 1996).

## Aplicații Practice în Muzică și Design de Sunet

Sinteza digitală a sunetelor a devenit o componentă esențială în producția muzicală contemporană, designul de sunet pentru filme și jocuri video, precum și în crearea de efecte sonore speciale. Compozitorii și designerii de sunet se bazează pe flexibilitatea și diversitatea sunetelor sintetizate pentru a crea compoziții unice și medii imersive.

## Importanța Interfețelor Utilizator

Interfețele utilizator ale sintetizatoarelor, fie că sunt hardware sau software, joacă un rol crucial în modul în care artiștii interacționează cu tehnologia. Interfețele bine proiectate permit un acces rapid și intuitiv la parametrii sintetizatorului, facilitând experimentarea și crearea sunetelor. Aceasta include utilizarea de controllere MIDI, interfețe tactile și alte dispozitive care ajută la crearea unei conexiuni mai directe și expresive între artist și instrument.

## Viitorul Sintezei Sunetului

Pe măsură ce tehnologia avansează, noi posibilități apar în domeniul sintezei sunetului. Dezvoltările recente includ sinteza bazată pe inteligență artificială, care poate genera sunete și muzică în moduri noi și neașteptate, și tehnologiile emergente vor continua să transforme modul în care interacționăm și experimentăm muzica și sunetul [5].

# Implementare

Implementarea proiectului a fost realizată în două fișiere script „.py”.

## Backend

Partea principală a programului este prezent în fișierul „synth.py” care este un fel de backend pentru proiect. În acest fișier sunt definite funcțiile de procesare de sunet, sinteză, generare și redare.

Funcția pentru generarea formelor de unde, generate\_waveform() este capabilă să creeze diferite tipuri de forme de undă, cum ar fi sinusoidală, pătrată, sinusoidală-pătrată și forme de undă bazate pe mostre sonore (pian, flaut, trompetă).

În cazul în care utilizatorul alege forme de unde generate, programul generează acestea cu ajutorul modului numpy, care pe baza capabilităților sale avansate de calcul numeric, permite crearea rapidă și eficientă a formelor de undă. De exemplu, în cadrul funcției generate\_waveform(), pentru a genera o undă sinusoidală, codul calculează un set de eșantioane pe parcursul duratei specificate prin formula: np.sin(2 \* np.pi \* freq \* t), unde freq reprezintă frecvența undei și t este un array de valori de timp, calculat pentru a se potrivi cu rata de eșantionare și durata. Aceasta produce o undă sinusoidală pură, reprezentativă pentru tonul respectiv. Pentru unda dreptunghiulară, codul folosește np.sign(np.sin(2 \* np.pi \* freq \* t)), care generează valori de +1 și -1, reprezentând formele abrupte ale unei unde dreptunghiulare. Semnalul numit „sine-square” este combinația celor două menționate anterior.

Pe de altă parte, dacă se alege utilizarea mostrelor de sunet de la instrumente muzicale reale, acestea sunt preluate din directoare specificate, cu locația lângă codul sursă, unde fiecare notă muzicală este stocată în câte un fișier de tip „.wav”, denumindu-se după frecvența sunetului. Astfel sunt câte 12 fișiere audio pentru fiecare instrument, anume:

* 261.63.wav (C4)
* 277.18.wav (C#4)
* 293.66.wav (D4)
* 311.13.wav (D#4)
* 329.63.wav (E4)
* 349.23.wav (F4)
* 369.99.wav (F#4)
* 392.00.wav (G4)
* 415.30.wav (G#4)
* 440.00.wav (A4)
* 466.16.wav (A#4)
* 493.88.wav (B4)

Cu funcția load\_instrument\_samples se deschide fișierul căutat pe bază de instrument și frecvență cu ajutorul modului wave, care apoi este delimitat în cadre, din care mai departe se extrag eșantioanele cu numpy, care se stochează într-un array separat pentru fiecare notă. În generate\_waveform, în cazul unei opțiuni de instrument real, se caută notele pentru instrumentul cutare, și fiecare formă de undă specifică frecvenței notei muzical va avea propria formă de undă generată din sunete înregistrate.

O altă funcționalitate implementată este cea de anvelopă ADSR (attack-decay-sustain-release), o tehnică de manipulare a sunetului foarte des folosită. Aceste patru efecte redau dinamica sunetului generat de către sintetizatorul virtual, și respectă parametri aleși de către utilizator. Dacă nu este ales niciun parametru, anvelopa nu este utilizată. Fiecare din cele patru efecte dinamice se poate trata ca un array de valori, definite cu ajutorul modului numpy. Attack este definit ca un vector de valori între 0 și 1, care are numărul de elemente egal cu valoarea de „attack” aleasă de către utilizator și rata de eșantionare care este setată la 44.1 kHz (np.linspace(0,1,attack\_length)). Decay este un vector de valori între 1 și valoare de sustain, care are numărul de elemente egal cu valoarea de „decay” aleasă de către utilizator și rata de eșantionare (np.linspace(1, sustain\_level, decay\_length)). Sustain este un vector de lungime sustain\_length (care este lungimea în timp a perioadei în care intensitatea sunetului se menține stabilă și este calculată ca fiind egal cu len(waveform) - attack\_length - decay\_length - release\_length), care are doar elemente de valoare sustain\_level. Release este reprezentat ca un vector de lungime release\_length, cu valori între sustain\_level și 0. La finalul funcției, forma de undă este înmulțită cu această anvelopă, astfel eșantionale vor avea valori diferite bazate pe parametri de timp și de valoare pentru fiecare etapă dinamică.

Pentru generarea de efect de ecou, se folosesc trei parametri luați de la GUI: feedback, delay\_time și mix. Primul este intensitatea ecoului, al doilea este timpul de delay acordat între sunetul principal și ecoul său, iar ultimul parametru controlează balanța între sunetul original și ecou. Initial, prin funcția de apply\_delay se definesc eșantionele de delay pe baza de delay\_time și rata de eșantionare, după care se crează un vector (care simulează o formă de undă) de lungime formă de undă de input + eșantionele de delay, prima parte a vectorului va fi locul eșantionelor din sunetul de bază iar restul valorilor vor fi 0 inițial, după care cu un ciclu for se modifică eșantionele astfel încât fiecărui eșantion cu index i+delay\_samples se adaugă valoarea sa înmulțită cu parametrul de feedback: delayed\_waveform[i + delay\_samples] += delayed\_waveform[i] \* feedback. După aceasta se generează outputul fiind o combinație între forma de undă originală și cea modificată cu ecou: output = delayed\_waveform[:len(waveform)] \* mix + waveform \* (1 - mix).

Implementarea modulației de frecvență (FM) este implementată în generate\_waveform și este bazată pe cele două argumente preluate de la interfața grafică, frecvența de modulare și indexul de modulare. Atunci când opțiunea de „fm” este aleasă, un sinusoid este generat cu numpy, fiind forma de undă modulatoare, iar outputul va fi un alt sinus în a cărei faze va fi integrat și forma de undă modulatoare, înmulțit cu indexul de modulare, cu linia de cod: waveform = np.sin(2 \* np.pi \* freq \* t + modulation\_index \* modulator), unde freq este frecvența notei muzicale actuale.

O altă funcționalitate importantă este paralelitatea, ceea ce ajută programul în a simula un pian real, pe care un muzician poate produce multiple sunete deodată (pentru acorduri de ex.). Aceasta este realizată în două faze. În prima fază, la pornirea programului, se pregenerează notele muzicale pentru tipurile de sunet simple (sinus, dreptunghiular și combinație), cu parametri de ADSR aleși impliciți și fără delay și feedback, pentru a optimiza programul inițial. După, la sfârșitul funcției de redare de sunet, se apelează funcția din backend de redare ca o funcție lamda, fiind un thread separat definit cu ajutotul modului the Thread threading.Thread(target=lambda: play\_waveform(waveform)).start(). Astfel fiecare sunet produs va fi controlat separat de către CPU, asigurând posibiliatea de a produce acorduri și multiple sunete.

## GUI

În cele ce urmează, va fi prezentată GUI-ul aplicației, care este dezvoltat în cadrul „SynthApp.py”.

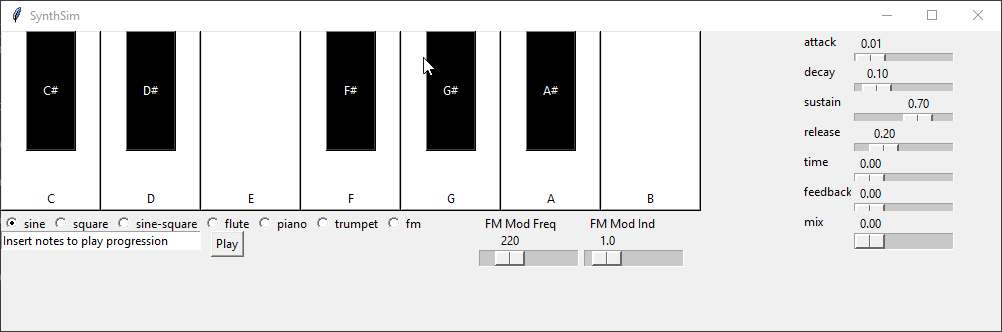


Figure 4: GUI of the application

După cum arată Figura XX, interfața grafică proiectată are trei funcționalități principale:

1. Implementarea unei claviaturi virtuale, cu o singură octavă. Aceasta poate fi folosită atât apăsând cu cursorul pe clape, cât și folosind tastatura, notele fiind mapate în plaja tastelor A-J
2. Setarea sunetului dorit, care poate fi modificat privind sunetul (forme de unde ‚ca sinus, dreptunghi sau utilizarea mostrelor de sunet de la instrumente reale), effecte ADSR sau feedback și modulație de frecvență cu setarea parametrilor
3. Scrierea unei secvențe de note muzicale care după, apăsând pe butonul Play, se poate reda cu ajutotul sintatizoarului, care va prelua și setările de sunet

Întreaga interfață grafică a fost creată cu ajutorul modului grafic implicit al Python, Tkinter.

Claviatura virtuală este implementată cu ajutorul obiectelor de tip buton din modulul menționat, care sunt setate să fie sau albe (pentru note întregi) sau negre (pentru semi note), pentru a semăna cu un pian real cât de bine posibil. La apăsarea unui buton sau a unei taste mapate de pe tastatură, funcția de play\_sound implementată în backendul proiectului va prelua setările de sunet pentru note și va reda sunetul generat sau căutat din mostre, cu ajutorul modului simpleaudio.

Setările de efecte și timbre sunt realizate prin butoane de tip radio sau slider-uri pentru a alege valori, care au magnitudine dependentă de felul efectului (de ex. effectul „attack” poate avea valoare între 0 și 1, pe când frecvența de modulație poate fi între 0 și 1000).

A screenshot of a computer

Description automatically generatedRedarea unui șir de note muzicale se realizează cu un textbox, în care utilizatorul trebuie să introducă note muzicale reale, cu majuscule. Acestea vor fi redate utilizând setările de sunet alese, cu o pauză de 0.5s între ele pentru a generaliza funcționalitatea. În cazul în care utilizatorul introduce note muzicale incorecte, un mesaj de eroare apare pe ecran, așa cum prezintă figura următoare.

# Rezultate

# Concluzie și posibile îmbunătățiri

# Bibliografie

|  |  |
| --- | --- |
| [1] | Tiobe, "TIOBE Index for January 2024," 2024. [Online]. Available: https://www.tiobe.com/tiobe-index/. |
| [2] | C. R. A. R. S. D. D. E. K. S. M. N. Jesse Engel, "Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders," in *Internation Conference of Machine Learning*, 2017. |
| [3] | M. N. M. Samridha Kumar, "Synthesizing musical notes of Harmonium using Spectral Domain Modeling," *International Journal of Scientific Engineering and Applied Science (IJSEAS),* vol. 1, 2015. |
| [4] | Making Music, "Quick Guide to Envelopes," [Online]. Available: https://making-music.com/quick-guides/envelopes/. [Accessed 15 01 2024]. |
| [5] | W. X. X. T. Q. L. Y. G. Zhen Ye, "NAS-FM: Neural Architecture Search for Tunable and Interpretable Sound Synthesis based on Frequency Modulation," *International Joint Conference on Artificial Intelligence,* 2023. |
| [6] | site. |
| [7] | M. R. Shailesh Ramamurthy, "Filter Design for Synthesis of Musical Notes: A Multidimensional featurebased approach," in *IEEE ICSIPA 2013*, Melaka, Malaysia, 2013. |
| [8] | S. K. S. D. P. S. Avani Dharne, "OPUS-An Android Based Speech to Musical Notes Converter," *INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT),* vol. 4, no. 4, 2015. |
| [9] | V. H. Valerii Kozlin, "Musical Projects in Guitar Pro 7.5 Sequencer," 12 2023. [Online]. Available: https://www.researchgate.net/publication/376744758\_Musical\_Projects\_in\_Guitar\_Pro\_75\_Sequencer. [Accessed 15 01 2024]. |
| [10] | G. Essl, "Topology in Sound Synthesis and Digital Signal Processing -- DAFx2022 Lecture Notes," in *DAFx2022*, Vienna, Austria, 2022. |
| [11] | V. m. i. —. n. s. u. p. models, "VESA VÄLIMÄKI, TAPIO TAKALA," *Organised Sound,* vol. 1, no. 2, pp. 75-86, 1996. |

# Anexa