Filtrarea semnalelor

Obiectivul general al acestui laborator este expunerea efectelor produse de aplicarea filtrelor asupra semnalelor audio și prezentarea modului de implementare al acestora în *React Native*.

**Short Track:**

Pentru a facilita desfășurarea laboratorului aveți disponibil proiectul la următoarea adresă: <https://github.com/Mossy23Chest/laborator_filtrare.git>. Pentru a clona proiectul local, utilizați în terminal comanda *git clone*, urmată de adresa furnizată mai sus. După clonarea proiectului intrați în noul director creat utilizând *cd laborator\_filtrare* în terminal. Este necesară instalarea tuturor dependențelor utilizate în cadrul aplicației. Acest lucru se realizează prin rularea în terminal a comenzii *npm install*. În cele din urmă, executați în terminal *npx expo run:android*  pentru a porni execuția aplicației. Respectarea acestor pași asigură configurarea corectă a mediului de dezvoltare pentru aplicația de filtrare a sunetului. De asemenea, în interiorul codului veți regăsii diverse comentarii prin intermediul cărora sunt explicate funcționalitățile anumitor instrucțiuni. În plus, pentru înțelegerea fluxului de execuție al aplicației, veți întâlnii numeroase afișări în consolă, specifice *JavaScript*, însă, în funcție de versiunea de *React Native* utilizată, acestea ar putea să genereze erori, deoarece sunt eliminate în *React Native 0.77*. Este important de menționat că unele aspecte, în special cele legate de interfața cu utilizatorul nu vor fi detaliate, deoarece acestea nu fac obiectul laboratorului. În continuare, va fi prezentată logica de implementare a funcționalităților aplicației, iar la finalul platformei de laborator veți regăsi cerințele ce trebuie rezolvate.

Programul a fost extins prin următoarea structură:

* *audioFilters.ts,* (regăsit în *utils/audioFilters.ts*) pentru aplicarea filtrului asupra eșantioanelor
* *my\_iir\_filter\_coeffs.json,* (regăsit în *utils/filters/my\_iir\_filter\_coeffs.json*) pentru introducerea coeficienților modelului matematic al filtrului

Noțiuni teoretice importante:

* filtrare, proces de manipulare a benzilor de frecvență care compun un semnal, prin evidențierea sau atenuarea acestora

Funcționalitățile adăugate aplicației sunt:

* aplicarea filtrelor asupra semnalelor originale

În finalul platformei de laborator veți regăsi atașate o serie imagine cu aplicația finală și modul în care ar trebui utilizată.

TODO:

Proiectați diverse tipuri de filtre (i.e. trece-sus, trece-sus, trece-banda) în funcție de rezultatul pe care doriți să îl obțineți și documentați rezultatele obținute. Ce constatați?

Comparați rezultatele obținute în urma aplicării unui filtru la alegere asupra unui semnal în *React Native,* cu efectul produs de aplicarea aceluiași filtru pentru același semnal în *Python*. Care sunt concluziile?

**Long Track:**

Pentru detalii suplimentare privind implementarea aplicației de laborator verificați comentariile din cod, deoarece acolo este explicată utilitatea fiecărei linii, oferind profunzime platformei actuale de laborator.

Ultima componentă implementată în cadrul acestui proiect este capacitatea de a filtra semnale. Datorită modului de structurare al proiectului, axat către înregistrarea în timp real a semnalelor, lucru care presupune că dimensiunea fișierului audio este variabilă, vom utiliza filtrele *IIR* (i.e. *Infinite Impulse Response*). Acestea au fost alese în principal datorită ordinului scăzut necesar proiectării acestui tip de filtre, cât și modului de aplicare recursiv al acestora asupra semnalelor, factori care au facilitat realizarea acestei componente în *React Native*. Proiectarea filtrului va fi efectuată în *Python*, așa cum a fost realizată și în prima parte a semestrului, urmând ca după obținerea filtrului, coeficienții acestuia să fie trecuți în *React Native*, în fișierul *my\_iir\_filter\_coeffs.json.* Pentru pasul final, aplicarea efectivă a filtrului asupra eșantioanelor semnalului, a fost creat fișierul *audioFilters.ts,* în care este implementată următoarea formulă recursivă[1]:

Unde:

* x[n] este semnalul de intrare
* y[n] este semnalul de ieșire
* b și a sunt coeficienții filtrului
* M și N sunt ordinele celor două polinoame

Pentru aplicarea filtrului a fost introdus un buton, având responsabilitatea de a furniza aplicației, fie eșantioanele originale, fie eșantioanele asupra cărora a fost aplicat filtrul. Astfel, prin introducerea unui *hook useEffect,* în momentul în care starea butonului este modificată (i.e. este apăsat) începe procesul de generare al spectrogramei, utilizând datele furnizate, fie originale, fie filtrate.

TODO:

Încercați să implementați funcționalitatea de proiectare directă a filtrelor în *React Native* și comparați rezultatele obținute cu filtrele proiectate în *Python*.

Bibliografie:

[1] J.G. Proakis, D.G. Manolakis, Digital Signal Processing: Principles, Algorithms, and Applications, Pearson Education, 2014

A screenshot of a computer screen

AI-generated content may be incorrect.

Figura a4.1 - Spectrograma semnalului înaintea aplicării filtrului trece-jos

A screenshot of a computer screen

AI-generated content may be incorrect.

Figura a4.2 - Spectrograma semnalului după aplicarea filtrului trece-jos

A screenshot of a computer screen

AI-generated content may be incorrect.

Figura a4.3 - Reprezentarea în timp și în frecvență a semnalului înaintea aplicării filtrului

A screenshot of a computer screen

AI-generated content may be incorrect.

Figura a4.4 - Reprezentarea în timp și în frecvență a semnalului după aplicarea filtrului