Aplikacija za sintezo ter obdelavo zvoka
Vmesno poročilo

I. Cilji

Cilj seminarske naloge je izdelati preprosti grafični vmesnik, ki bo uporabnikom omogočal sintezo in nadaljnjo obdelavo zvočnih signalov. Za primer nabora funkcionalnosti bomo vzeli knjižnico JSyn [1], ki temelji na jeziku Java.

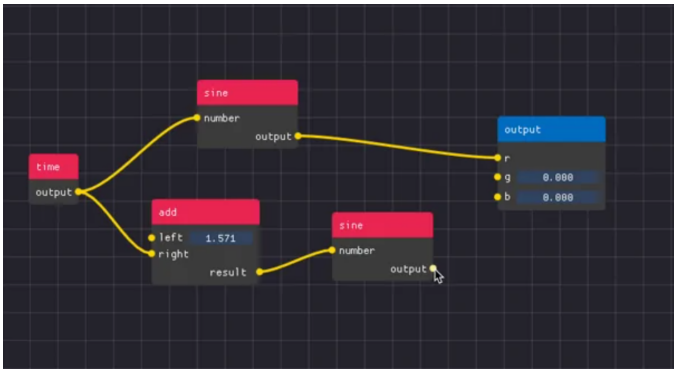

Fig. 1. Primer grafičnega vmesnika

Grafični vmesnik bo predstavljal ploščo na katero bodo uporabniki lahko postavljali in skupaj povezovali različne funkcijske bloke. Ti bloki bodo prejeli vhod, ga preračunali ter podali nov izhod.

II. Primerjava pristopa

Za implementacijo rešitve smo se odločili za jezik Python. Predvsem zaradi hitrejšega razvojnega cikla in tudi raznolike ponudbe uporabnih knjižnic. Za izrisovanje vmesnika bomo uporabili knjižnico DearPyGui [2]. Ta velja za nekoliko manj znano ogrodje za izdelavo grafičnih vmesnikov.Izbrali smo jo predvsem zato, ker podpira izrisovanje blokov, kar nam bo precej olajšalo delo.

Za različne kalkulacije bomo uporabili paket NumPy. Ta nam bo omogočal hitrejše računanje vseh funkcijskih blokov in generatorjev. Izhodni blok bo na podlagi podane dolžine in frekvence vzorčenja zvoka ustvaril polje diskretnih časovnih oznak ter ga nato podal povezanim blokom za nadaljnjo obdelavo.

Okviren pristop bo tesno sledil delovanju orodja JSyn. Ta ima definirana posamezna vozlišča, ki jih lahko uporabniki programsko povezujejo skupaj ter na koncu tudi izračunajo ter predvajajo rezultat. Vsak blok oziroma vozlišče bo imelo funkcijo *calculate*, ki prejme polje diskretnih časovnih oznak. Funkcija nato pridobi vse vhode, izračuna ter vrne polje vrednosti, ki ustrezajo časovnim oznakam.

Za predvajanje in shranjevanje rezultatov bomo uporabili knjižnico *sounddevice*. Ta nam omogoča, da polje vrednosti predvajamo skupaj z podano frekvenco. Prav tako nam omogoča shranjevanje in nalaganje zvočnih datotek.

III. Trenutno delo

Delo smo pričeli z grafičnim vmesnikom za upravljanje z bloki. Čeprav je izrisovanje večinoma že implementirano moramo vseeno dodati lastno implementacijo za sledenje stanju vseh blokov ter povezav med njimi. Definirali smo abstraktni razred iz katerega izhajajo vsi funkcijski bloki. Podobno implementacijo smo naredili tudi za vse atribute, ki pripadajo posameznim blokom. Na ta način si olajšamo izdelovanje novih blokov z različnimi operacijami.

Atributi predstavljajo vhode, izhode in statične nastavitve posameznih blokov. Če vzamemo za primer generator sinusnega signala ima dva atributa: frekvenco ter amplitudo. Atributi lahko vrednost prejmejo kot izhod drugega bloka, ali pa nastavljeno statično s strani uporabnika.

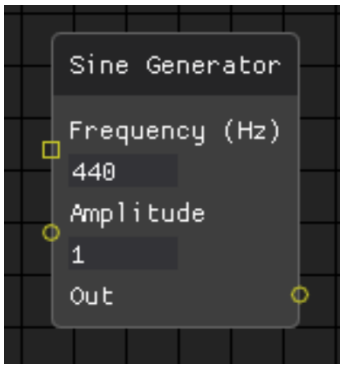

Fig. 2. Blok za generiranje sinusnega signala

Atributi so tudi tipizirani. Trenutno sta implementirana celoštevilska in racionalna številska tipa. Pri ustvarjanju povezav preverjamo tudi pravilno ujemanje tipov.

Vse implementirane bloke lahko prosto dodajamo, urejamo ter tudi odstranjujemo. Edina izjema je izhodni blok, ki mora biti vedno prisoten na platnu. Trenutna implementacija ima

na voljo blok za generiranje sinusnega signala ter seštevalnik. Izhodni blok ima zaenkrat implementirano le predvajanje izhodnega signala skozi zvočnike na napravi. Generiranje in predvajanje se trenutno sproži, ko uporabnik pritisne na tipkovnico.

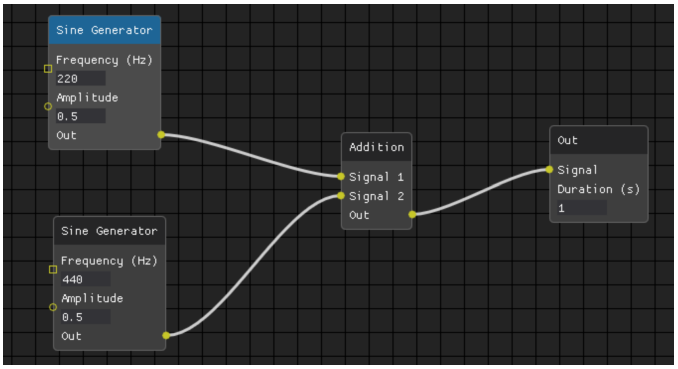

Fig. 3. Primer uporabe vseh implementiranih blokov

Trenutno implementacijo smo tudi testirali. Posneli smo sistemski zvok, ki ga tvori naša aplikacija ter uvozili v Audacity. Tam smo s pomočjo spektralne analize potrdili ustreznost tvorjenih frekvenc.

IV. Nadaljni koraki

V preostanku dela se bomo ukvarjali z mnogimi izboljšavami uporabniške izkušnje ter predvsem dodajanjem novih blokov. Manjka nam veliko blokov za osnovne matematične operacije, kot na primer: odštevanje, množenje, deljenje. Za tem bomo implementirali bloke za filtriranje signalov. Implementirali bomo tudi blok ki bo omogočal nalaganje signala iz zvočne datoteke.

Če bo čas dopuščal, bomo dodali preprosto vizualizacijo trenutnega izhodnega signala. To nam bo pomagalo pri potrjevanju delovanja naše aplikacije brez nepotrebne uporabe orodja Audacity.

References

[1] *JSyn*: Java sound synthesis library https://www.softsynth.com/jsyn/
[2] *PyIMGUI*: Python GUI library https://github.com/pyimgui/pyimgui
[3] Cook, Perry R. *Real sound synthesis for interactive applications.* CRC Press, 2002.