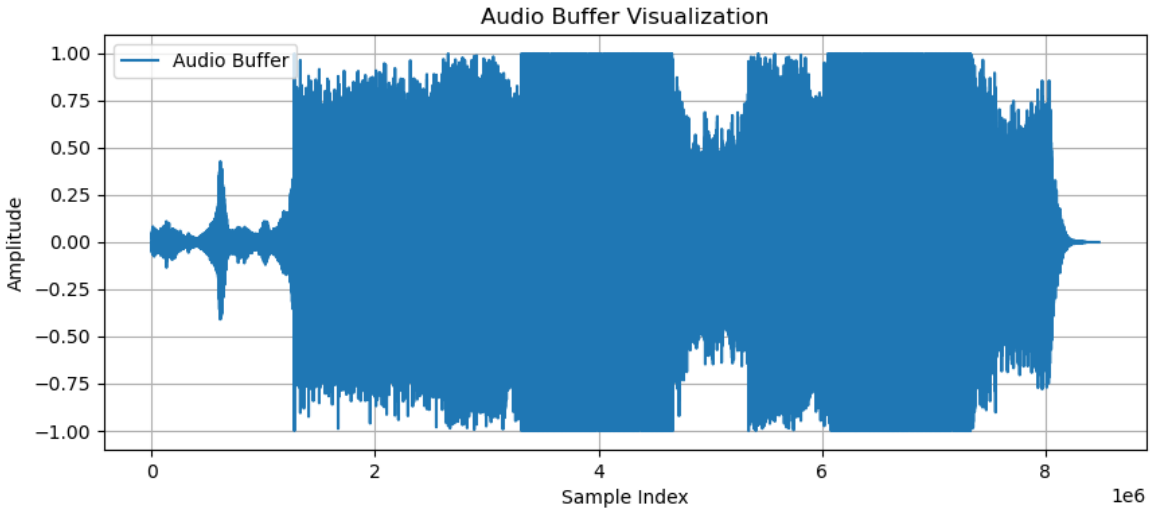
Propozycja II projektu zaliczeniowego z PiMN:

Obróbka dźwięku (format WAV),

czyli elementy transformaty Fouriera

Opracowanie *Jędrzej Wardyn*

1 lipca 2024



Rysunek 1: Wizualizacja kawałka "Crab Rave"

Wszelkie pytania proszę kierować na mail: `j.wardyn@uw.edu.pl`

1 Program do obróbki plików dźwiękowych za pomocą transformaty Fouriera

Jednym z przykładów narzędzi do obróbki plików dźwiękowych i video jest ffmpeg. Oparty jest o niego między innymi program do nagrywania OBS. Jeśli mówimy o graficznych programach tylko do audio mamy na przykład Tenacity(klon Audacity, który jest wolny/libre i otwartoźródłowy (FLOSS)).

W tym zadaniu celem będzie odtworzyć część funkcji z takiego programu mając do dyspozycji pliki w formacie .wav [dla uproszczenia], gdzie za pomocą cross-platformowej biblioteki mamy bezpośrednio dostęp do zapisu fali dźwiękowej.

Dzięki temu będziemy mogli kontrolować dokładnie co dzieje się z dźwiękiem, napisać nowe, unikalne funkcje, których w tych programach nie ma albo nie są tak wygodnie opakowane.

Będzie wam dostępna biblioteka "libsndfile"do otwierania plików .wav oraz podstawowy plik początkowy .cpp. W pliku początkowym znajduje się odczytywanie pliku i zapisywanie pliku do formatu .wav. Transformatę Fouriera można robić np przez bibliotekę FFTW. Resztę musicie napisać sami.

Poniżej znajdują się wskazówki co chciałbym zobaczyć w takim programie, niemniej cenię waszą kreatywność w ujęciu tego tematu. Jeżeli ktoś nie wykona czegoś nawet wymaganego, ale zrobi za to coś innego fenomenalnie to otrzyma punkty.

Audio do testów można ściągać z stronki z Royalty Free muzyką.

Stronka do konwersji .mp3 do .wav

Nie jest koniecznym korzystanie z C++ ani praca na plikach .wav wyłącznie, ale macie już tu dostępny kod w C++ poniżej.

PLIK NA POCZĄTEK STARTOWY: DO UŻYTKU

1.1 Co widziałbym, że ma się znaleźć:

1. Program może być odpalany z konsoli z argumentami funkcji main jaki plik ma przyjąć i co ma z nim robić. (Trochę jak ffmpeg). [OPCJONALNE]: Jak ktoś chce się pogimnastykować, to może spróbować zrobić coś na kształt menu do wyboru funkcji, coś bardziej graficznego, ale w konsoli to proszę bardzo.

2. Podstawa: program ma ALBO wypisywać audio input i output do plików które może wyrysować ALBO bezpośrednio wyrysowywać rysunki. Istotna jest możliwość inspekcji tego, jak nasz program obrobi audio albo jego konkretnej sekcji, warto zrobić jakąś opcję zoomowania.

3. Jako funkcje do włączania: wycinacz ciszy w audio, gdzie możemy wybrać sobie wartość progową jak i długość czasu od którego ma się zaczynać usuwanie. Np nie ma usuwać krótszego dźwięku niż 1 sekunda.

4. Usuwanie DC: jeśli dźwięk jest podniesiony całościowo w jedną stronę, czyli na przykład bardziej wystaje z góry niż z dołu (niesymetrycznie) to powinien zostać obniżony, żeby był z powrotem symetryczny.

5. "Główny punkt programu": Equalizer: Wykorzystując szybką transformatę Fouriera (a również STFT) oraz jej odwrotność mamy na celu napisanie czegoś, co może nam usuwać wybrane przedziały częstotliwości z całego nagrania lub fragmentu nagrania.
   Celem **nie jest tutaj** napisanie od zera algorytmu FFT, a umiejętność jego zastosowania.

do wyboru 1. Low-pass/high pass filter: rozszerzenie powyższej idei: napisz zestaw filtrów, które możesz zastosować na częstotliwościach w celu wywołania odpowiedniego efektu.

do wyboru 2. Przyspieszacz: [1] Wykorzystajcie transformatę Fouriera do utworzenia przyspieszacza/spowalniacza audio, ale takiego, który nie podnosi ogólnego tonu nagrania [jak 2x na youtube]. Podpowiedź: jeśli rozbijamy dźwięk na punkty to z twierdzenia o próbkowaniu wystarczą nam 2 punkty do odtworzenia ciągłej fali sinusoidalnej.

do wyboru 3. Pitch Shift: Napisz coś co zmienia całościowy ton danego nagrania.

do wyboru 4. Kompresor: Czasami audio które posiadamy nie jest znormalizowane, lub nie podlega kompresji (zmniejszeniu dynamiki czyli różnic między najcichszymi dźwiękami i najgłośniejszymi). Jest to problematyczne, gdy nie jest to zabieg artystyczny a błąd w nagraniu. Napisz coś co zidentyfikuje najgłośniejsze fragmenty dźwięku i ściszy je [sam ustal jakie parametry zastosujesz, ale możesz wzorować się np na Audacity], oraz opcjonalnie podgłośni te cichsze. Zachęcam do rozważenia opcji rozbudowania tego narzędzia o różne parametry.

1.2 Dla uproszczenia:

1. Nic nie musi dziać się w czasie rzeczywistym, program ma przyjąć, popracować sobie i oddać gotowy plik audio po przeróbkach i wyrysować cały plik lub jego wybraną część.

2. Nie musimy tworzyć żadnego konwertera do .mp3 lub .opus, ani innych formatów, biblioteka jedynie ma zastosowanie do .wav i innych bardzo prostych w strukturze plików. Ale nie zabraniam.

3. Wystarczy obrabiać audio monofoniczne.

[1] Osobista historia: Będąc na drugim roku mój sąsiad w domie studenta 2 puszczał mi przez ścianę 14 razy z rzędu wersję nightcore kawałka z filmu o piratach. Od tamtego czasu inaczej patrzę na przyspieszanie muzyki.

2 Wymagania i obwieszczenia:

Poniżej umieszczone są warunki, które wpływają na ostateczną punktację projektu. Zasadniczo każdy zaczyna z maksymalną ilością punktów, które odejmuję za każdą skuchę, chyba że robi absolutne minimum i widać, że nie włożył w to żadnego postarania. Wtedy zaczynam dodawać punkty od dołu.

Kod projektu:

1. Powinien posiadać czytelną i zrozumiałą instrukcję (w osobnym pliku) kompilacji i użytkowania programu.
2. **Powinien kompilować/uruchamiać się (brak tego warunku daje 0 punktów).**
3. Powinien uruchamiać się bez błędów w stylu `segmentation fault`
4. Nie powinien skutkować jakimś szkodliwym działaniem, na przykład nadprodukcją zapisywanych plików zapełniających całą dostępną pamięć.
5. Przy wprowadzaniu lub użyciu danych, otwieraniu plików i innych czynnościach zależnych od pewnych czynników program powinien sprawdzać czy operacja została wykonana poprawnie, jeśli nie to powinien zostać pokazany odpowiedni do sytuacji komunikat błędu.
6. Kod programu powinien realizować wytyczne z opisu projektu.
7. (tam gdzie jest to możliwe) Warto jakby nie było zbytecznej redundancji
8. Warto by program nie był powolny lub nad wyraz zasobożerny polecam flagę (-O3 chyba, że macie wyraźne powody by jej nie użyć).
9. W kodzie dobrze jakby nie było części bezużytecznych (dodajcie `-Wall` do flag kompilacji), śmieci pozostałych po tworzeniu rozwiązania, proponuję wyczyścić kod a potem dodać komentarze.
10. W miarę możliwości kod ma być przejrzysty, opatrzony komentarzami w celu szybszego zrozumienia sprawdzającego, (co nie znaczy, że nie można używać jakichś trików)

Punkty 1-6 są najważniejsze, ale spełnienie ich nie gwarantuje maksymalnej ilości punktów.

Wszelkie pytania proszę kierować na mail: `j.wardyn@uw.edu.pl`