VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2018 Peter Pilát

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

VYTVOŘENÍ VAMP PLUGINU PRO APLIKACI SONIC VISUALISER
VAMP PLUGIN FOR SONIC VISUALISER

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Peter Pilát**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Tomáš Kiska**
SUPERVISOR

BRNO 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Audio inženýrství**
Ústav telekomunikací

Student: Peter Pilát ***ID:*** 189032
Ročník: 3 ***Akademický rok:*** 2017/18

NÁZEV TÉMATU:

Vytvoření Vamp pluginu pro aplikaci Sonic Visualiser

POKYNY PRO VYPRACOVÁNÍ:

Naprogramujte zcela nový Vamp plugin pro aplikace Sonic Visualiser a Sonic Annotator. Ten bude obsahovat řadu parametrizačních funkcí popisující hudební signál zejména z hlediska barvy zvuku (spektrogram, chromagram), ale také tempa (BPM) či dynamiky (RMS). V neposlední řadě bude možno za pomoci Sonic Annotator možné tyto funkce využit i v prostředí Matlab.

DOPORUČENÁ LITERATURA:

[1] CANNAM, C.; LANDONE, C.; SANDLER, M. Sonic Visualiser: An Open Source Application for Viewing, Analysing, and Annotating Music Audio Files, in Proceedings of the ACM Multimedia 2010 International Conference, 2010: s. 1-2.

[2] MÜLLER, M. Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications [online]. Springer International Publishing Switzerland, 2015, 483 s. ISBN 978-3-319-21945-5.

Termín zadání: 5.2.2018 ***Termín odevzdání:*** 29.5.2018

Vedoucí práce: Ing. Tomáš Kiska
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

Abstrakt

V mojej bakalárskej práci sa venujem získavaniu informácií z hudby, spôsobom, akými je možné ich získavať, aspektom hudobných informácii a tiež samotnej aplikácii daných metód. Následne analyzujem obsahovo orientované metódy na spravovanie hudby a tiež do nich zahrnutú parametrizáciu hudobných nahrávok a zvukového signálu celkovo. Po oboznámení s konkrétnymi parametrizačnými nástrojmi do ktorých budem zrealizovaný Vamp plugin implementovať a ktorými sú Sonic Visualiser a Sonic Anotator, charakterizujem Vamp Plugin a podrobne vysvetľujem prvky jeho zloženia. Po vysvetlení ako dané pluginy a v nich prebiehajúce výpočty fungujú, uvádzam výsledok riešenia danej problematiky, ktorým sú 4 funkčné Vamp Pluginy s funkciou zobrazenia Spektrogramu a Chromagramu s rôznymi nastaviteľnými parametrami, funkciou výpočtu RMS daného signálu s možnosťou segmentácie a tiež funkciou zobrazenia tempa danej zvukovej stopy alebo prípadných zmien v tempe skladby. V neposlednej rade uvádzam možné využitie týchto pluginov v budúcnosti a to v rôznych odvetviach.

Kľúčové slová

Získavanie hudobných informácií, Aspekty hudobnej informácie, Aplikovanie MIR, Použité metódy, Parametrizácia, Nástroje pre parametrizáciu, Sonic Visualiser, Sonic Annotator, Vamp Plugin, Efektívna hodnota, RMS, Miera, Spektrogram, Chromgram, DFT, constant Q, Určovanie tempa

Abstract

In my Bachelor Thesis, I devote myself to obtaining information from music, the way it can be obtained, the aspect of musical information, and the use of the methods themselves. Then I analyze content-oriented music management methods and also include parameterisation of music recordings and audio signal overall. After familiarizing with the specific parameterization tools to implement the Vamp plugin, which are Sonic Visualiser and Sonic Anotator, I characterize the Vamp Plugin and explain in detail its composition. As explained in the manuals and the calculations in progress, the RMS calculation function of the given signal with the possibility of segmentation functions as well as the function of displaying the sound rate or possible changes in the track temperature. Last but not least, we mention the possible use of these supplements in the future and in different sectors.

Keywords

Music Information Retrieval, Facets of Music Information, MIR aplication, Methods used, Parametrization, Tools for Parametrisation, Sonic Visualiser, Sonic Anotator, Vamp Plugin, Root mean Square, RMS, constant Q, Spectrogram, Chromagram, DFT, Tempo Estimation

Bibliografická citace:

PILÁT, P. *Vytvoření Vamp pluginu pro aplikaci Sonic Visualiser*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2018. 66 s. Vedoucí bakalářské práce Ing. Tomáš Kiska

Prehlásenie

„Prehlasujem, že svoju záverečnou prácou na téma *Vytvoření Vamp Pluginu pro aplikaci Sonic Visualiser* som vypracoval samostatne pod vedením vedúceho semestrálnej práce a s použitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce. Ako autor uvedenej záverečnej práce ďalej prehlasujem, že v súvislosti s vytvorením tejto záverečnej práce som neporušil autorské práva tretích osôb, hlavne som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a som si plne vedomý následkov porušení ustanovení § 11 a nasledujúcich autorského zákona č. 121/2000 Sb., vrátane možných trestne právnych dôsledkov vyplývajúcich z ustanovení časti druhej, hlavy VI. diel 4 Trestného zákonníku č. 40/2009 Sb.

V Brne dňa **31. Mája 2018** ……

podpis autora

Poďakovanie

Ďakujem vedúcemu bakalárskej práce Ing. Tomášovi Kiskovy za účinnú metodickú, pedagogickú a odbornú pomoc a ďalšie cenné rady pri spracovaní mojej bakalárskej práce.

V Brne dňa **31. Mája 2018** ……

podpis autora

OBSAH

ZOZNAM OBRÁZKOV

ZOZNA TABULIEK

ÚVOD

Svet zvukových informácií je pre veľa ľudí veľmi vzdialený, hoci do styku s ním prichádzajú každý deň. Či už počúvate rádio, prezeráte nové skladby na youtube alebo sa snažíte nájsť pieseň, ktorá vám už dlhšie hrá v hlave ale nemôžete si spomenúť ako sa volá. Ako je možné, že všetky skladby na internete alebo v rádiu majú tak kvalitný zvuk? Prečo nikde nie je počuť žiadny šum alebo praskanie? Ako len funguje tá magická aplikácia Shazam na mojom telefóne, ktorá behom pár sekúnd nájde pieseň, ktorú ja nemôžem spoznať?

Celý proces úpravy a spracovania hudobných skladieb alebo filmového zvuku sa odvíja od zvukových informácií. To čo sa nám na prvý pohľad môže zdať ako odstránenie nežiaduceho šumu v nahrávke, je v skutočnosti použitie určitého pluginu, za ktorým sa ukrývajú hodiny programovania a ladenia. Pri spracovaní zvukových informácií napríklad spomenutým pluginom dochádza k rozloženiu zvukového signálu na fragmenty alebo inak povedané vzorky, ktoré putujú na parametrizáciu aby sme na ne neskôr mohli aplikovať rôzne funkcie, ktoré nám v konečnom výsledku napríklad pomôžu rozoznať výšku tónu v skladbe, prítomnosť ľudskej reči alebo za nás prepíšu skladbu do notových osnov.

V prvej kapitole mojej práce sa venujem Získavaniu hudobných informácií, spôsobom akými tieto informácie môžeme získavať ale tiež rôznym aspektom hudby, ktorých znalosť je k pochopeniu získavania hudobných informácií nevyhnutná. Taktiež uvádzam konkrétne spôsoby aplikovania „Získavania hudobných informácií “ v praxi. V druhej kapitole podrobne rozoberám problém parametrizácie hudobných nahrávok, spôsoby akými je možné parametrizáciu vykonávať.

V tretej kapitole uvádzam konkrétne príklady parametrizačných programov spolu s náväznosťou na neskôr spomenutý Vamp Plugin. V nasledujúcej kapitole sa dostávame k jadru spracovania hudobných informácií a teda k samotnému Vamp Pluginu, jeho stavbe a funkcii. Po oboznámení sa s Vamp Pluginom uvádzam konkrétne výpočty, rovnice a grafy potrebné k zostaveniu našich pluginov. Na záver práce uvádzam využitie týchto pluginov v spolupráci s programom Sonic Annotator ako aj význam a využitie mojich pluginov v praxi.

1 ZÍSKAVANIE HUDOBNÝCH INFORMÁCIÍ (MIR)

Predstavte si svet, v ktorom prídete k počítaču a spievate si kúsok piesne, ktorý ste počuli pri raňajkách v rádiu a nemôžete sa ho zbaviť. Počítač akceptuje váš nie práve najlepší spev, opraví ho a okamžite vám navrhne, že "*Camptown Races*" je príčinou vášho problému. Vy potvrdíte návrh počítača vypočutím jedného z mnohých MP3 súborov, ktoré našiel. Spokojný, láskavo odmietate ponuku na prevzatie všetkých ostatných existujúcich verzií skladby vrátane nedávno vydaného talianskeho rapového remixu a orchestrálnej verzie s doprevádzaním na gajdy. Existuje takýto systém dnes? Nie. Bude existovať v budúcnosti? Áno. Bude sa dať takýto systém ľahko vyrábať? Rozhodne nie. Treba prekonať ešte mnohé ťažkosti pred tým než sa Systémy na získavanie hudobných informácií (MIR) stanú realitou. Závratne komplexná interakcia hudobných rozsahových, časových, harmonických, farebných, editoriálnych, textových a bibliografických aspektov, demonštruje len jeden z komplikovaných problémov MIR. Voľba reprezentácie hudby - či už ide o symboly, zvuky alebo oboje- záleží na dalších zlúčeninách, kedže každá voľba určuje šírku pásma, výpočet, vyhľadávanie, požiadavky a možnosti rozhrania. Z prekrývania multikultúrnych, multizážitkových a multidisciplinárnych aspektov hudby je zrejmé, že výzvy, ktorým čelí výskum MIR a vývoj majú ďaleko od triviálnych.[1]

Obnovenie hudobných informácií (MIR) je teda interdisciplinárna veda o získavaní informácií z hudby. Je to malá, ale rastúca oblasť výskumu s mnohými aplikáciami v reálnom svete. Tí, ktorí sa podieľajú na MIR, môžu mať vedomosti z hudobnej oblasti, psychologickej oblasti, oblasti akademickej hudobnej štúdie, spracovania signálov, strojového učenia alebo ich kombinácie. [2]

Obnovenie hudobných informácií ako proces v sebe zahŕňa množstvo ďalších bodov a metód, ktorými sú:

- Výpočtové metódy pre klasifikáciu, zoskupovanie a modelovanie
- Formálne metódy a databázy
- Softvér na vyhľadávanie hudobných informácií
- Interakcia človeka s počítačom a rozhraniami
- Hudobné vnímanie, poznanie, ovplyvňovanie a emócie
- Hudobné analýzy a reprezentácie znalostí
- Hudobné archívy, knižnice a digitálne zbierky
- Duševné vlastníctvo a práva

- Sociológia a ekonomika hudby

Pri MIR je potrebné si uvedomiť, že hudba sa podstatne líši od textu a preto sa treba zaoberať jej vlastnosťami medzi ktoré patria:

- **Rozsah**
  - súvisí s vnímaním základnej frekvencie zvuku
  - pohybuje od nízkych alebo hlbokých po vysoké alebo akútne zvuky
- **Intenzita**
  - ktorá súvisí s amplitúdou a tým pádom aj s energiou vibrácií
  -slovné označenie pre rozsah intenzity je od mäkkých až po hlasné zvuky
  - intenzita je tiež definovaná ako hlasitosť tónu.
- **Farba**
  -je definovaná ako vlastnosť zvuku, ktorá umožňuje poslucháčom vnímať ako odlišne znejú dva zvuky v rovnakej výške a s rovnakou intenzitou

Ďalšie dôležité pojmy v rámci získavanie informácií z hudby sú:

- **Orchestrácia**
  -odvíja sa od skladateľov a účinkujúcich a ich voľby, ktoré nástroje budú zahrnuté v piesni
- **Akustika**
  -môže byť považovaná za príspevok akustiky miestnosti, hluku pozadia, post-spracovania zvuku, filtrovania a ekvalizácie
- **Rytmus**
  -týka sa periodického opakovania s možnými malými variantmi
- **Melódia**
  -je tvorená sekvenciou tónov s podobnou farbou, ktoré majú rozoznateľnú výšku v rámci malého frekvenčného rozsahu
- **Harmónia**
  -je skupina simultánnych zvukov s rozpoznateľným rozstupom pozdĺž časovej osy
- **Štruktúra**
  - je horizontálny rozmer, ktorého časová mierka je odlišná od predchádzajúcich, je spojená s makro-úrovňovými funkciami, ako sú opakovania, prekladanie tém a refrénov, prítomnosť prestávok, zmeny časových údajov atď. [3]

Tab.1.1: Rozdelenie vlastností hudby z rôznych hľadísk

Časová os	Dimenzia	Obsah
Krátka doba	Farba Orchestrácia Akustika	Kvalita produkovaného zvuku Zdroje zvukov Kvalita nahraného zvuku
Stredná doba	Rytmus Melódia Harmónia	Vzory zvukovej konfigurácie Sekvencie nôt Sekvencie akordov
Dlhá doba	Štruktúra	Organizácia hudobného diela

1.1 Aspekty hudobnej informácie:

1.1.1 „Multiaspektová výzva“

Hudobné informácie pozostávajú zo siedmych aspektov, z ktorých každý hrá rôzne úlohy pri definovaní domény MIR. Tieto aspekty sú rozsah, časové, harmonické, editoriálne, textové, bibliografické aspekty a aspekt farby. V dôsledku komplikovanosti vlastného zobrazovania hudobných informácií, to čo nasleduje nie je analýza aspektov v užšom zmysle, pretože aspekty sa vzájomne nevylučujú. Napríklad termín *adagio*, ktorý môžeme nájsť v notovom zápise môže byť umiestnený v kontexte v rámci časových aj editorálnych aspektov. Harmonický aspekt podobne vyplýva predovšetkým z interakcie rozsahových a časových aspektov. Problémy, ktoré vyvstávajú z komplexná interakcia rôznych aspektov hudobných informácií môžu byť označené ako *"multiaspektová výzva"*.

1.1.2 Aspekt Výšky

Výška tónu je „ je základná charakteristika tónu úmerná jeho kmitočtu (počtu kmitov za sekundu)(Randel, 1986, str. 638). Najznámejšie je grafické znázornenie (napr. 𝅘𝅥𝅯, 𝅗𝅥, 𝅝, 𝄻 , atď.) v ktorom je výška reprezentovaná vertikálnou pozíciou nôt na notovej osnove. Názvy nôt (napríklad A, B, C #), rozsahy stupníc (napr. I, II, III ... VII), solfège (napr. Do, ré, mi ... ti) a čísla tried výšky (napr.0, 1, 2, 3 ... 11) sú tiež jednou z mnohých metód reprezentácie výšky. Rozdiel medzi dvoma výškami

tónov sa nazýva interval. Intervaly môžu byť reprezentované rozdielnym znakom medzi dvoma výškami nameranými v polohách (napr. -8, -7 ... -1, 0, +1 ... +7, +8 atď.) alebo ich tónovou kvalitou , ktorá je určená umiestnením dvoch polôh v rámci syntaxe *„Západnej teoretickej tradície“.* Napríklad interval medzi tónmi A a C # je nazývaný *Veľká Tercia*, zatiaľ čo foneticky ekvivalentná vzdialenosť medzi tónmi A a Des je *zmenšená Kvarta*. Melódia môže byť považovaná za súbory buď stôp alebo intervalov postupne usporiadané v čase. Pojem kľúč je tu zahrnutý ako „podaspekt“ výšky. Súbor tónov EDCEDC (napr. "Tri slepé myši") v husľovom kľúči je považovaný za hudobne ekvivalentný tónom BAGBAG v basovom kľúči. To znamená, že ich melodické rysy (t.j. vzorec intervalov) sú vnímané poslucháčmi ako rovnocenné napriek skutočnosti, že absolútne výšky ich tónov sú rozdielne.

1.1.3 Časový aspekt

Informácie týkajúce sa trvania hudobných diel spadajú pod časový aspekt. Patria sem indikátory tempa, merač, dĺžka trvania, harmonické trvanie a akcenty. Týchto päť prvkov dohromady tvorí rytmickú zložku hudobného diela. Pomlčky v ich rôznych formách môžu byť považované za ukazovatele trvania hudobných diel, ktoré neobsahujú žiadne informácie o výške tónov. Časové informácie znamenajú významné reprezentatívne a prístupové problémy. Môžu byť absolútne (napr. metronóm indikujúci MM=80),všeobecné (napr. *adagio*, *presto*, *fermata*), alebo relatívne (napr. *rýchlejšie, pomalšie*). Občas sa vyskytujú časové deformácie (napr. Rubato, accelerando, rallentando). Pretože rytmické aspekty diela určuje komplexná interakcia medzi tempom, metrom, výškou a harmonickým trvaním a prízvukom (či už označeným alebo implicitným), je možné reprezentovať daný rytmický model mnohými spôsobmi, z ktorých všetky prinášajú foneticky identické výsledky. Niektoré postupy pri hudobnom predstavení, pri ktorých sa očakáva, že sa hráči budú odchyľovať od prísnych rytmických hodnôt zaznamenaných v notovej osnove (napr. v barokovom, jazzovom) spôsobujú ďalšie komplikovanosti podobné tým ktoré sú spôsobené časovými skresleniami uvedenými vyššie. Dá sa teda povedať, že reprezentácia časových informácií na účely vyhľadávania je skutočne náročná.

1.1.4 Harmonický aspekt

Keď zaznejú dve alebo viac výšok naraz, hovorí sa, že došlo k simultánnosti alebo k harmónii. Tomuto sa tiež hovorí polyfónia. Neprítomnosť polyfónie sa nazýva monofónia (tzn. Iba jedna výška znejúca v čase). Výšky, ktoré sa vertikálne zaraďujú do štandardného západného notopisu, vytvárajú harmóniu. Vzájomné pôsobenie

výšky a časových aspektov na vytvorenie polyfónie je ústrednou črtou západnej hudby. V priebehu storočí hudobní teoretici kodifikovali najčastejšie simultány do niekoľkých komplexných reprezentatívnych systémov, založených na ich základných intervaloch alebo výškach a vnímanej funkcii týchto intervalov alebo výšok v rámci kontextu diel, v ktorých sa objavujú. Teoretici tiež kodifikovali spoločné postupné vzory súbežnosti nájdené v západnej tónovej hudbe. Aj keď je mimo rozsah tejto kapitoly podrobne preskúmavať komplexnú oblasť západnej harmonickej teórie a praxe, je dôležité poznamenať, že jednotlivá harmonická udalosť môže byť označená kombináciou výšok tónov alebo intervalov, ktoré obsahuje, a pozíciou stupnice jej "koreňovej" alebo základnej výšky. Akord, ktorý zaznie, keď brnkneme do gitary, je príkladom harmonickej udalosti. Sekvencie akordov alebo harmonické udalosti môžu byť reprezentované akordovými menami. Veľmi častá harmonická sekvencia alebo postup v ladení veľké C, [C + F + G + C +] je tu reprezentovaný názvom základnej noty každého akordu. "+" označuje, že každý akord obsahuje intervaly Veľká tercia a Čistá kvinta, merané od základnej noty. Ďalšou metódou reprezentácie tejto harmonickej progresie, ktorá ju zovšeobecňuje na všetky hlavné kľúče/ľadenia, je naznačiť stupeň stupnice koreňa akordu pomocou rímskej číselnej notácie: I-IV-V-I.

Jednoduchý prístup ku kodifikovaným aspektom harmonickej informácie diela môže byť problematický, pretože jeho harmonické vlastnosti, hoci prítomné v notovom zápise, nie sú zvyčajne jednoznačne označené jedným zo spôsobov opísaných vyššie. Výnimkou je zaradenie akordových názvov alebo akordových symbolov do najpopulárnejších hudobných hier a harmonická skratka nazývaná basso continuo alebo figurované basy, ktorá sa bežne vyskytuje v barokovej hudbe. Táto záležitosť je ešte komplikovaná faktom, že ľudská myseľ môže vnímať a dôsledne pomenovať jednu z kodifikovaných simultánností napriek prítomnosti mimoriadnych výšok, ktoré sa nazývajú tóny bez akordov. Dokonca aj pri absencii alebo oneskorení jedného alebo viacerých tónov akordov, väčšina členov západných spoločností môže stále dôsledne klasifikovať akord.

1.1.5 Aspekt farby

Aspekt farby zahŕňa všetky aspekty tónovej farby. Zvukové rozlíšenie medzi notou hranou na flaute a na klarinete je spôsobené rozdielmi vo farbe. Z tohto hľadiska spadá do úvahy informácie o orchestrácii, teda označenie špecifických nástrojov na vykonávanie všetkých alebo časti diela. V praxi sú informácie o orchestrácii, hoci sú súčasťou aspektu farby, niekedy považované za súčasť bibliografického aspektu.

Jednoduché vymenovanie nástrojov použitých v kompozícii je obvykle zahrnuté ako súčasť štandardného bibliografického záznamu. Zistilo sa, že tieto informácie pomáhajú pri opise a tým pri identifikácii hudobných diel. Široká škála metód výkonu ovplyvňuje aj farbu hudby (napr. Pizzicatti, mutings, pedalings, bowings). Tu sa hranica medzi informáciami o farbe a editačnými informáciami stane rozmazanou, keďže tieto výkonové metódy môžu byť umiestnené aj v editačnej podobe. Akt určovania metódy výkonu, ktorý ovplyvňuje farbu je editoriálny. Zvukový efekt výkonu zvolenej metódy je časový.

Časové informácie sú najlepšie prenášané v zvukovom alebo signálnom zobrazení diela. Sprístupnením časových informácií prostredníctvom časového dopytu (napr. Prehrávanie tlmených trubiek a požiadavka na zhody) vyžaduje pokročilé možnosti spracovania signálu. Jednoduchšia, no menej precízna metóda by bola prístup k časovým informáciám prostredníctvom nejakého druhu interpretácie redakčných značiek. Toto možné riešenie by samozrejme podliehalo rovnakým ťažkostiam spojeným so zastupovaním redakčných informácií, o ktorých sa ďalej hovorí.

1.1.6 Vydavateľský aspekt

Pokyny pre predstavenie tvoria väčšinu vydavateľského aspektu. Patria sem prstové prvky, ornamentácia, dynamické pokyny (napr. ppp, p, … f, fff), ligatúry, artikulácie, staccati, šmyky a tak ďalej. Rozmary vydavateľského aspektu predstavujú množstvo ťažkostí. Jedna obtiažnosť spojená s vydavateľskými informáciami, je to, že môžu byť buď ikonické (napr. -, 3,!) alebo textové (napr. Crescendo, diminuendo) alebo oboje. Ďalej vydavateľské informácie môžu zahŕňať aj samotné časti hudby. Písanie harmónií z *basso continuo*, tiež známe ako "Realizácia vykreslenia basu", je redakčný akt. *Cadenzi* a ďalšie sóla, ktoré pôvodne zamýšľali mnohí skladatelia improvizovať, sú často realizované vydavateľom. Nedostatok vydavateľských informácií je ďalší problém, ktorý treba zvážiť. Rovnako ako *basso continuo*, kde sú harmónie implikované, takmer všetci skladatelia pred Beethovenom – a veľa z nich aj po ňom – jednoducho predpokladali, že výkonní umelci sú kompetentní robiť prácu správnym spôsobom bez pomoci vydavateľských informácií. V mnohých prípadoch vydavateľské rozdiely medzi vydaniami tej istej práce robia výber "*konečnej*" verzie diela pre začlenenie do systému MIR veľmi problematické.

1.1.7 Textový aspekt

Texty piesní, árií, chóra, hymny, symfónie a tak ďalej, sú zahrnuté do textového aspektu. Libretti, text opery, je tam tiež zahrnutý. Je dôležité poznamenať, že textový aspekt hudobných informácií je viac nezávislí od melódií a usporiadaní, ktoré sú s ním spojené, ako by si jeden myslel. Daný textový fragment niekedy nie je dostatočne informatívny na to, aby identifikoval a získal a požadovanú melódiu a naopak (Temperley, 1993). Voľné výmeny textu a hudby sú silnou tradíciou v západnej hudbe. Dobrým príkladom tohto javu je melódiu *"Boh zachráňme kráľovnú*". Táto melódia je známa občanom Britského spoločenstva národov ako hymna ich kráľovnej. Táto jednoduchá melódia je tiež známa Američanom ako ich republikánska pieseň, *"Amerika",* alebo *“My Country ’tis of Thee”.* Mnoho piesní prešlo aj prekladom do rôznych iných jazykov. Jednoducho povedané, treba si uvedomiť, že daná melódia môže mať viacero textov a daný text by mohol mať viacero hudobných prevedení. Je tiež dôležité pamätať si existenciu obrovského množstva hudby bez akéhokoľvek textu.

1.1.8 Bibliografický aspekt

Informácie týkajúce sa názvu diela, skladateľa, aranžéra, editora, autora textov, vydavateľa, vydania, čísla katalógu, dátumu vydania, diskografie, umelca (-ov), atď., sú všetky časťami bibliografického aspektu. Toto je jediný aspekt hudobných informácií, ktorý nie je odvodený od obsahu kompozície. Je to skôr informácia o hudobnom diele. Jedná sa o hudobné metadáta. Všetky ťažkosti spojené s tradičným bibliografickým popisom a prístupom nás stretávajú aj tu. *Howard a Schlichte (1988)* načrtávajú tieto problémy spolu s niektorými z navrhovaných riešení. *Temperley (1993)* je ďalšia dôležitá práca pri riešení tejto problematickej témy. [4]

1.2 Aplikovanie MIR

1.2.1 Odporúčacie systémy

Niektoré odporúčacie systémy pre hudbu už existujú, ale prekvapujúco málo je založených na technikách MIR, namiesto toho sa využíva podobnosť medzi používateľmi alebo náročná kompilácia údajov. *Pandora* napríklad používa odborníkov na označenie hudby osobitnými vlastnosťami, ako napríklad "*speváčka*" alebo "*silná basová linka*". Mnoho ďalších systémov vyhľadáva používateľov, ktorých história počúvania je podobná a ponúka im doteraz

nepočutú hudbu z príslušných zbierok iných užívateľov. Techniky MIR začínajú byť pre podobnosť v hudbe súčasťou takýchto systémov.

1.2.2 Rozlíšenie skladieb a rozpoznávanie nástrojov

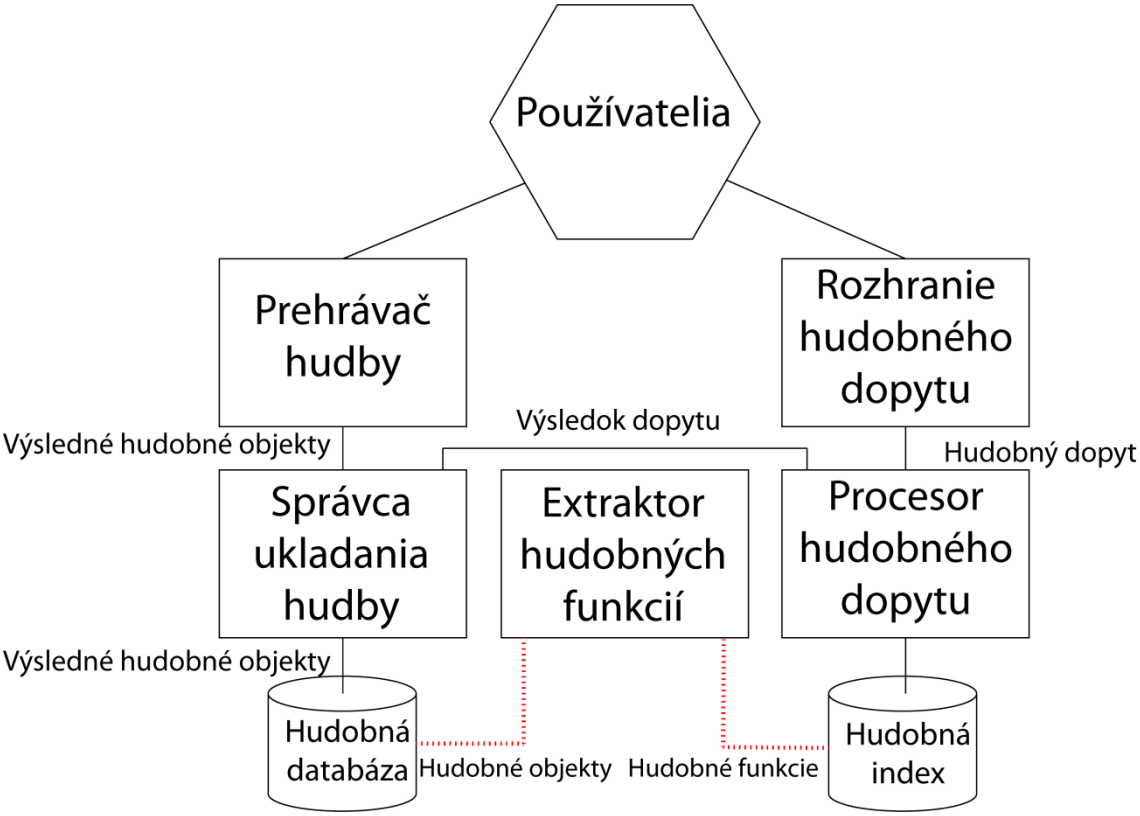
Oddelenie skladieb je o vyňatí pôvodných skladieb , ktoré môžu mať viac ako jeden nástroj hrajúci v skladbe. Rozpoznanie nástroja je o identifikácii použitých nástrojov a / alebo o rozdelení hudby na jednu stopu na nástroj. Boli vyvinuté rôzne programy, ktoré môžu rozdeliť hudbu do zložkových častí bez prístupu k hlavnej kópii. Týmto spôsobom môžu byť vytvorené napríklad karaoke skladby z bežných hudobných skladieb, hoci proces ešte nie je dokonalý kvôli vokálom, ktoré zaberajú časť rovnakého frekvenčného priestoru ako ostatné nástroje.

1.2.3 Automatické prepisovanie hudby

Automatické prepisovanie hudby je proces prevodu zvukových nahrávok do symbolickej notácie, ako je notový zápis alebo súbor MIDI . [5] Tento proces zahŕňa niekoľko podskupín, medzi ktoré patrí detekcia viacerých rozstupov, detekcia nástupu , odhad trvania, identifikácia nástroja a extrakcia rytmických informácií. Tento problém stáva náročnejším s väčším počtom nástrojov a vyššou úrovňou polyfónie .

1.2.4 Automatická kategorizácia

Kategorizácia hudobného žánru je bežnou úlohou pre MIR a je zvyčajnou úlohou pre každoročnú výmenu informácií o obnovení hudobných informácií (MIREX).[6] Techniky strojového učenia, ako napríklad „*Podporné vektorové stroje*“, majú tendenciu dobre fungovať napriek trochu subjektívnej povahe klasifikácie. Ďalšie možné klasifikácie zahŕňajú identifikáciu umelca, miesto pôvodu alebo náladu diela. Tam, kde sa predpokladá, že výstup bude číslom, a nie triedou, je potrebná regresná analýza .[7]



Obr.1.1: Stavba MIR

2 OBSAHOVO ORIENTOVANÉ METÓDY PRE SPRÁVU DIGITÁLNEJ HUDBY

Osobné archívy digitálnej hudby sa v posledných rokoch stávajú čoraz častejšími, s popularitou MPEG Layer 3 (MP3) [8] [9] a iných proprietárnych formátov. S týmto trendom, ktorý má veci urýchliť, budú potrebné nástroje, ktoré budú navigovať a pomáhať spravovať tieto archívy. Informácie o hudobnom žánri sa môžu ukázať ako užitočné, keď pomáhajú používateľom sústrediť sa na hudbu podľa svojich preferencií. Zatiaľ čo mnohé kompresné formáty obsahujú primerané žánrové anotácie, sú často nevhodné, nepresné, nadmerné alebo jednoducho vynechané. Tieto textové anotácie môžu byť doplnené alebo nahradené vlastnými starostlivo vybranými kategóriami používateľa, ktoré zaplnia presne požadovanú množinu hudby. Možnosť ďalšieho využitia je vyhľadávanie hudby založené na obsahu. Schopnosť nahrávať hudbu akusticky podobnú skladbe alebo krátkemu extraktu je neoceniteľným nástrojom na nájdenie nových skladieb, ktoré zodpovedajú vkusu používateľa vo veľkých archívoch (prípadne v celej sieti).

Alternatívne techniky, ako je napríklad analýza nákupných modelov, vyžadujú značné údaje a sú uplatniteľné len po určitom čase po uvoľnení. Rovnaká základná technológia má iné aplikácie. Napríklad ukladanie skladieb podľa obľúbenosti užívateľovi umožňuje dostať sa k novým skladbám ,o ktorých systém predpokladá, že sa budú užívateľovi páčiť. Umožňuje to filtrovanie potenciálnych skladieb do archívu, alebo napríklad digitálneho rozhlasového vysielania prispôsobeného hudobnému vkusu. Na spracovanie hudby sú v tejto práci skúmané dve základné techniky. Po prvé, Gausova modelácia zmesí (GMM), ktorá modeluje distribúciu akustiky a osvedčila sa pre klasifikáciu audia. Za druhú metódu je považovaná metóda Tree-Based Vektorovej kvantizácie (TreeQ) [10], ktorá uplatňuje diskriminačný prístup. Každá z týchto techník vyžaduje parametrizáciu zvukových vzorkov do funkčných vektorov. Východiskové výsledky používajú Mel-Frequency Kepstrálne koeficienty, ktoré sú populárne v komunite reči. Zavádza sa alternatívna metóda, ktorá zabraňuje úplnej dekompresii zvuku a parametrizácii výslednej krivky. V tejto schéme sú MPXEP, kepstrálne parametre rýchlo odvodené z čiastočne dekomprimovaného zvuku vrstvy MPEG. Táto práca popisuje dané techniky a parametre podrobnejšie. Následne je popísané ich použitie pri klasifikačných a vyhľadávacích pokusoch a sú uvedené výsledky.

2.1 Techniky

2.1.1 Gaussova modelácia zmesy

Gaussova modelácia zmesy sa používa na rôzne úlohy klasifikácie zvuku, ako je napríklad rozpoznanie reproduktorov [11]. V mojej práci sa táto technika používa na modelovanie skladby alebo hudobného žánru ako funkcia hustoty pravdepodobnosti (PDF) pomocou kombinácie hmotnostných čísel Gaussových komponentov PDF (zmesí). Pretože vektory funkcií v tejto práci majú primerane nekorelované komponenty, môžu sa použiť výpočtovo výhodné dvojité kovariantné matice. Predpokladá sa, že parametre GMM maximalizujú pravdepodobnosť sekvencie tréningového pozorovania. Aj keď nie je známy spôsob, ako to riešiť v uzavretej forme, pravdepodobnosť sa môže lokálne maximalizovať pomocou metódy Raurn-Welch (očakávanie-maximalizácia). Tento iteračný proces pokračuje dovtedy, kým sa hodnoty parametrov nezhromaždia s optimálnym riešením. GMM sa používa na určenie pravdepodobnosti, že vektor testovacieho objektu patrí k tomuto modelu. Iteráciou všetkých pozorovaní sa môže vytvoriť notový zápis pre celú skladbu. Namiesto jednoduchého súčtu tohto zápisu sa tu však používa forma normalizácie rámca na obmedzenie príspevku každého jednotlivého rámca.

2.1.2 Tree-Based vektorová kvantizácia (TreeQ)

Druhá technika radšej rozlišuje vektorový kvantifikátor, než by sa snažila priamo modelovať akustickú funkciu [12]. Výsledky boli hlásené pomocou tejto techniky na identifikáciu reproduktorov ako aj na vyhľadávanie krátkych záberov zvuku a hudby. Cvičné dáta sú parametrizované najprv do vektorov funkcií. Každý tréningový príklad je spojený s triedou, ako je umelec alebo žáner. Vytvorí sa strom kvantifikácie, ktorý automaticky rozdeľuje priestor funkcie do oblastí, ktoré majú maximálne rozdielne triedy populácií. Po vytvorení sa môže strom použiť na vytvorenie šablóny histogramu pre skladbu alebo žáner. Všetky vektory funkcie spadajú cez strom do listových buniek. Relatívne množstvo vzoriek v každej bunke vytvára histogramovú šablónu. Tieto šablóny sú kompaktným vyjadrením akustiky piesne a vzdialenosť medzi ktoroukoľvek z nich poskytuje odhad akustickej podobnosti. Napriek tomu, že je k dispozícii mnoho techník na porovnanie dvoch vektorov, v tejto práci sa používa meranie vzdialenosti kosínusu [13].

2.2 Schémy parametrizácie

Prvá etapa spracovania hudby vyžaduje, aby komprimovaná digitálna hudba vo formáte MP3 bola parametrizovaná na vhodné vektory. Tieto vektory by si mali uchovávať charakteristické informácie pri odstraňovaní nepotrebných akustických detailov. Mel-Frekvenčné kepstrálne koeficienty (MFCC) [14] [15] boli testované jako prvé. Tento proces vyžaduje, aby bol zvuk úplne dekomprimovaný. Výsledná digitálna vlnová krivka sa v priebehu parametrizácie vráti do spektrálnej domény, aj keď je iná. V druhej schéme MPXEP sa táto čiastočná redundancia eliminuje pri použití parametrov odvodených priamo z dát MP3 subpásma.

2.2.1 Mel-Frekvenčné kesptrálne koeficienty(MFFC)

MFCC sú populárne v komunite spracovania reči a poskytujú dobrú diskriminačnú výkonnosť s primeranou odolnosťou proti hluku. Aj napriek tomu, že MFCC sú určené pre reč, nie pre hudbu, sú dobrým východiskovým bodom. V našej schéme sa okná zvuku 25 milisekúnd zvažujú každých 10 milisekúnd. Diskrétna Fourierova transformácia premieňa každé okno na spektrum. Spektrálne koeficienty sa nahromadia do zásobníkov na mel-mierke - nelinearita zdôrazňujúca vnímanie dôležitých oblastí s nízkou a strednou frekvenciou. Logaritmické hodnoty zásobníka sú potom transformované pomocou diskrétnej kosínovej transformácie na dvanásť rozumných nekorelovaných kepstrálnych koeficientov. Typicky je

pripojený logaritmický energetický pojem. Trinásť zložkový vektorový prvok je produkovaný 100 krát za sekundu.

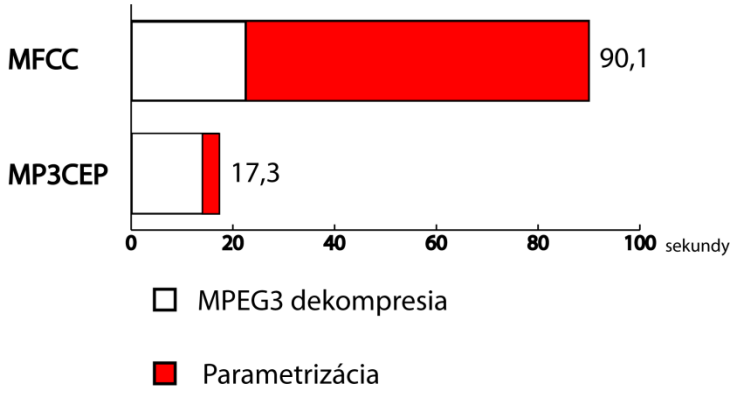
2.2.2 MP3CEP

Tento proces začína ako bežná MPB dekompresia vrátane parsovania bitových tokov a dekompresie frekvenčnej vzorky. Keď sú dostupné údaje o čiastkových pásmach, použije sa toto ako zdroj parametrizácie skôr než by sa mali syntetizovať skutočné vzorky syntetickým filtrom. Každý rámec MP3 (zodpovedajúci 1152 vzorkám PCM) pozostáva z dvoch granúl. Pri štandardnej piesni s rozlíšením 44,1 kHz sa granule vyskytujú približne každých 13 ms v rovnakom poradí ako frekvencia snímok MFCC 10 ms. Na kopírovanie veľkosti okna MFCC (25 ms) sa vytvárajú vektory s využitím subpásmových dát v dvoch susedných granulách (13 * 2 = 26 ms). Dve granuly obsahujú 36 vzoriek subpásiem, pričom každá vzorka subpásma je vektorom s 32 amplitúdami frekvenčného pásma (rovnako rozmiestnenými). Vektor veľkosti subpásma sa vytvorí súčtom veľkostí týchto šesťdesiat šiestich vektorov vzoriek subpásma. Výsledný 32-komponentný vektor sa redukuje na 20 zložiek pomocou deformácie podobnej mel-deformácii. Komponenty s nižšou frekvenciou sú nezmenené, zatiaľ čo tie vyššie sú kombinované, v čoraz väčších počtoch. Po zapojení logaritmu každej zložkovej hodnoty sa produkuje dvanásť kepstrálnych koeficientov pomocou diskrétnej kosínusovej transformácie.

Napokon, sumarizácia veľkosti vo všetkých subpásmach odhaduje energetický pojem, ktorého logaritmus je pripojený. Trinásť zložková funkcia vektoru sa vypočíta 76,6 krát za sekundu.

2.2.3 Parametrizačné časy

Schéma MFCC je určená na dobrý výkon, zatiaľ čo schéma MP3CEP je určená predovšetkým na rýchlosť. Na obrázku 2.1 nižšie, je zobrazený čas potrebný na parametrizáciu typickej skladby vo formáte MP3 pre obe schémy. Skutočné hodnoty sú menej dôležité ako pomer medzi nimi, pretože pre obe stratégie sa môžu použiť rôzne optimalizácie. Možno vidieť, že parametrizácia MP3CEP je približne šesťkrát rýchlejšia. Je však potenciálne menej presná a je špecifická pre formát. To, či tieto nevýhody prevažujú nad rýchlosťou, je záležitosť špecifická pre aplikáciu.



Obr.2.1: Porovnanie času spracovania (v sekundách), nadekompresiu a parametrizáciu vo vektoroch funkcií MFCC a MP3CEP pre typickú skladbu vo formáte MP3 (3 minúty 45 sekúnd).

2.3 Porovnanie parametrizácií

Druhý stĺpec tabuľky 2.1 je výsledkom replikácie experimentov s vektormi funkcií MP3CEP, a nie s MFCC. Trend GMM okrajovo prekonávajúci systém TreeQ je znova zrejmé. Zaujímavým výsledkom je, že presnosť klasifikácie pomocou parametrov MP3CEP pre najlepší systém GMM32 je 90,9% len 1,2% za údajom MFCC. To je prekvapujúco dobrý výsledok, najmä ak sa používajú viac ako 20% menej vektorov. Pre systém TreeQ poklesne výkonnosť z MFCC na MP3CEP na 5%. Degradácia výkonu však môže predstavovať malú cenu, ktorú treba zaplatiť za účelom umožnenia spracovania hudby pri interaktívnych rýchlostiach. Hlavnou nevýhodou MP3CEP je jeho špecifickosť vo formáte MP3. Napriek tomu, že pre alternatívne schémy kompresie zvuku sú možné podobné techniky, napríklad formát MPEG2-AAC, kompatibilita je nepravdepodobná a nasadenie v archíve zmiešaného formátu je nevhodné.[16]

Systém	Vektory funkcie	
	MFCC	MP3CEP
GMM 8	90,3	86,9
GMM 32	92,0	90,9
TreeQ 500	89,7	85,1

Tab.2.1: Percento skladieb presne klasifikovaných pomocou zmiešavacích systémov GMM 8 a 32 a najlepšieho systému TreeQ pomocou vektorových funkcií MFCC a MP3CEP.

3 NÁSTROJE PRE PARAMETRIZÁCIU

Pri parametrizovaní nahrávok môžeme používať rôzne hudobné softvéry, ktoré slúžia na nahrávanie a úpravu hudby alebo zvukov. V mojom prípade si bližšie predstavíme dva softvéry, do ktorých som vkladal mnou naprogramovaný Vamp plugin, ktorými sú Sonic Visualiser a Sonic Anotator.

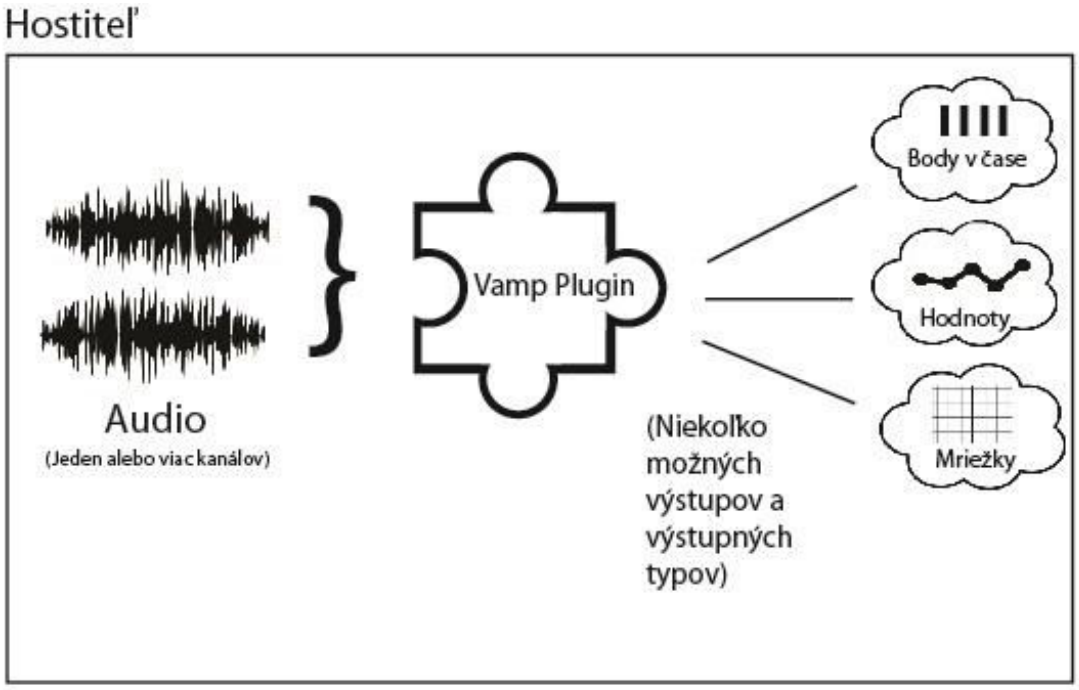
3.1 Sonic Visualiser

Základnou požiadavkou na analýzu nahrávok je systém, ktorý vám umožňí pozorne a pružne počúvať danú nahrávku a na tento účel slúži Sonic Visualiser. Softvér bol vyvinutý Chrisom Cannamom z Centra pre digitálnu hudbu v Queen Mary v Londýnskej univerzite (vrátane zopár vstupov od spoločnosti CHARM) je tento bezplatný program veľmi prispôsobivým prehrávacím a vizualizačným prostredím, ktoré obsahuje také funkcie ako prehrávanie s premenlivou rýchlosťou, slučky a schopnosť spoznámkovávať nahrávku, napríklad kvôli identifikácii konkrétnych referenčných bodov. Funkcia poznámok môžeme tiež využiť na priame prepisovanie BPM(úderov za minútu) skladby, čím vytvoríme údaje o tempe, ktoré sa dajú zobraziť na obrazovke alebo vyexportovať do tabuľkového programu. Obzvlášť atraktívnou funkciou je schopnosť synchronizovať počet nahrávok, aby ste mohli skočiť z jedného bodu v nahrávke do zodpovedajúceho bodu v inej nahrávke. Existuje tu celý rad vstavaných vizualizácií, ako sú spektrogramy, ktoré sú opäť vysoko prispôsobiteľné, ale základnou silou Sonic Visualiser je jeho schopnosť podporovať pluginy tretích strán. Tie ponúkajú neustále rozširujúce sa spektrum analytických zariadení od automatizovaného zisťovania nástupu až po odhad tónovej výšky a zachytenie údajov o intenzite. [17]

4 VAMP PLUGIN

Vamp Plugin je kus kompilovaného programového kódu, ktorý vykonáva analýzu digitálneho audio signálu, vráti výsledky v inej forme ako audio signál. Tieto výsledky sa často nazývajú zvukové funkcie. Vamp pluginy sú distribuované v súboroch zdieľanej knižnice s príponou .dll, .so alebo .dylib v závislosti od použitej platformy s jedným alebo viacerými pluginmy v každej knižnici. Plugin sa zvyčajne identifikuje pomocou počítača, názvu zdieľanej knižnice a krátkeho textového identifikátora, ktorý pomenuje doplnok v knižnici. Plugin nemôže byť použitý samostatne, ale iba s vyhovujúcou "hostiteľskou" aplikáciou, ktorá načíta plugin zo

svojej zdieľanej knižnice a zavolá funkcie v rámci kódu pluginu, aby ho nakonfiguroval, naplnil údajmi a spustil ho.



Obr.4.1: Stavba Vamp Plugin-u

Do pluginu nie je privedený celý zvukový vstup naraz, ale namiesto toho sú doňho privádzané krátke série blokov: v tomto ohľade pripomína pluginy na spracovanie zvuku v reálnom čase, ale vo všetkých ostatných ohľadoch je vykonávaný "off-line", čo znamená, že sa neočakáva, že bude fungovať v reálnom čase a nie je povinný vracať vlastnosti signálu tak ako sa objavujú. Binárne rozhranie pre Vamp plugin je definované v jazyku C a plugin SDK (sada vývoja softvéru) v jazyku C++, ale odporúča sa, aby programátori pluginov používali rozhrania C ++.

4.1 Zloženie Vamp Plugin-u

Vamp plugin potrebuje sprístupniť určité množstvo informácií hostiteľskému programu (napr. Sonic Visualiser).
Každý plugin, bez ohľadu na to aký jednoduchý, musí poskytnúť hostiteľovi nasledujúce:

1. Identifikátor, názov a popis samotného pluginu.
2. Meno výrobcu pluginu, stav autorských práv a číslo verzie produktu.
3. Vstupnú doménu ktorej plugin poskytne svoj zvuk.
4. Predvolená veľkosť kroku a veľkosť bloku pluginu pre audio vstup.

5. Minimálny a maximálny počet vstupných kanálov, s ktorým je plugin schopný pracovať.
6. Zoznam deskriptorov výstupov, ktoré obsahujú informácie o štruktúre výsledkov, ktoré môže plugin vytvoriť.
7. Implementáciu štandardných funkcií, ktoré nadstavujú, obnovujú a spúšťajú doplnok.

Niektoré pluginy majú parametre, ktoré je možné nastaviť tak, aby upravili spôsob spracovania. Tieto pluginy bude navyše musieť poskytnúť hostiteľovi:

8. Zoznam deskriptorov parametrov, ktoré obsahujú informácie o editovateľných parametroch pluginu. Hostiteľ môže použiť tieto deskriptory napríklad na to, aby používateľovi ukázal okno s ovládacími prvkami pluginu.
9. Implementáciu štandardných funkcií, ktoré získavajú a nastavujú hodnoty parametrov.

Zásuvný modul môže mať aj súbor vopred definovaných konfigurácií, ktoré môžu byť používateľovi ponúkané podľa mena. Tie sú známe ako programy a plugin, ktorý ich podporuje, bude tiež musieť poskytnúť hostiteľovi:

10. Zoznam názvov programov.
11. Implementáciu štandardných funkcií, ktoré načítajú aktuálny názov programu a vyberú nový program. Základná trieda jazyku C ++ poskytuje virtuálne metódy, ktoré ich potlačujú.

4.2 Základné triedy

Vamp SDK obsahuje jednu triedu, z ktorej by mali byť odvodené triedy implementácie pluginov. Táto trieda, Plugin, odhaľuje čisto virtuálne metódy pre väčšinu prístupových a vykonávacích funkcií, ktoré majú potrebujú triedy pluginov implementovať. Tie, ktoré nie sú priamo definované v doplnku Plugin, sú sami zdedené z ďalšej triedy nazývanej PluginBase, ktorá obsahuje virtuálne metódy pre veci, ktoré nie sú špecifické pre výstupné štruktúry používané vo Vamp - názov pluginu, výrobca, parametre, názvy programov atď. Rovnako ako všetko v súbore SDK,aj tieto triedy sa nachádzajú v priestore názvov Vamp(Vamp namespace). Triedy Plugin a PluginBase tiež obsahujú množstvo dátových tried, ktoré sa

používajú pri návrate balíkov informácií o funkciách (Feature, FeatureList, FeatureSet), výstupov (OutputDescriptor), parametrov (ParameterDescriptor) atď.

4.2.1 Identifikátory, názvy a deskriptory

Vamp využíva kombináciu reťazcov "identifier", "name" a "description" na opis niekoľkých druhov objektov. Samozrejme, samotný plugin musí implementovať metódy getIdentifier, getName a getDescription (zdedené z čisto virtuálnych metód v PluginBase), ktoré vrátia textové informácie o plugine. Podobné údaje sú zahrnuté ako členy verejných údajov v triedach ParameterDescriptor a OutputDescriptor.

Vo všetkých týchto prípadoch sú účely troch reťazcov tieto:

- **Identifikátor (Identifier)** - Mal by obsahovať krátky reťazec, ktorý môže hostiteľ použiť na označenie objektu v rámci bezprostredného okolia. To znamená, že identifikátor pluginu musí byť jedinečný v rámci knižnice pluginu. Identifikátor deskriptora výstupu musí byť jedinečný medzi deskriptormy výstupu pluginu. Podobne to platí pre deskriptory parametrov. Identifikátory sú veľmi obmedzené čo sa týka znakov a môžu obsahovať iba: abecedné znaky ASCII, veľké a malé písmená, číslice 0 až 9, pomlčku (znamienko mínus, "-") a znaky podčiarknutia ("_")
- **Názov (name)** – Ide o text, ktorý ukáže hostiteľ používateľovi ako normálny štítok objektu
- **Opis (Description) –** tento reťazec je voliteľný a môže obsahovať ďalší text na opis účelu objektu spôsobom, ktorý rozširuje informácie uvedené v „názve“. Hostitelia, ktorí zobrazujú popis používateľovi, to zvyčajne urobia okrem „názvu“, takže by nemali duplikovať informácie, ktoré už sú v reťazci „Názov“.

4.2.2 Vstupy (Inputs)

Vstupom do Vamp Pluginu sú audio dáta s jedným alebo viacerými kanálmi. Zvuk nie je prekladaný, takže doplnok získava súbor údajov o ukazovateľoch, jeden na každý kanál. Plugin môže určiť, koľko kanálov bude akceptovať pomocou metód getMinChannelCount a getMaxChannelCount. Počet kanálov, ako aj veľkosť kroku a veľkosť bloku, ktoré sa použijú pri spustení doplnku, sú fixné pri zavolaní inicializačnej metódy pluginu initialise.
bool initialise(size_t inputChannels, size_t stepSize, size_t blockSize);

Ak plugin rozhodne, že hodnoty dodané na inicializáciu sú neprijateľné, mal by vrátiť hodnotu false, aby naznačil, že inicializácia zlyhala. Po inicializácii, na dodanie zvukových údajov a spustenie pluginu hostiteľ opakovane volá metódu process. Metóda process obdrží súbor vstupných ukazovateľov a časovú pečiatku timestamp.

```
FeatureSet process(const float *const *inputBuffers, RealTime timestamp);
```

Zakaždým, keď sa volá metóda process, prejde jeden blok zvuku veľkosti vo vzorkách rovnajúcich sa veľkosti bloku ktorý bol posunutý metóde initialise. Rozdiel v počte vzoriek medzi vstupom pri jednom volaní metódy process a pri ďalšom sa rovná veľkosti kroku.

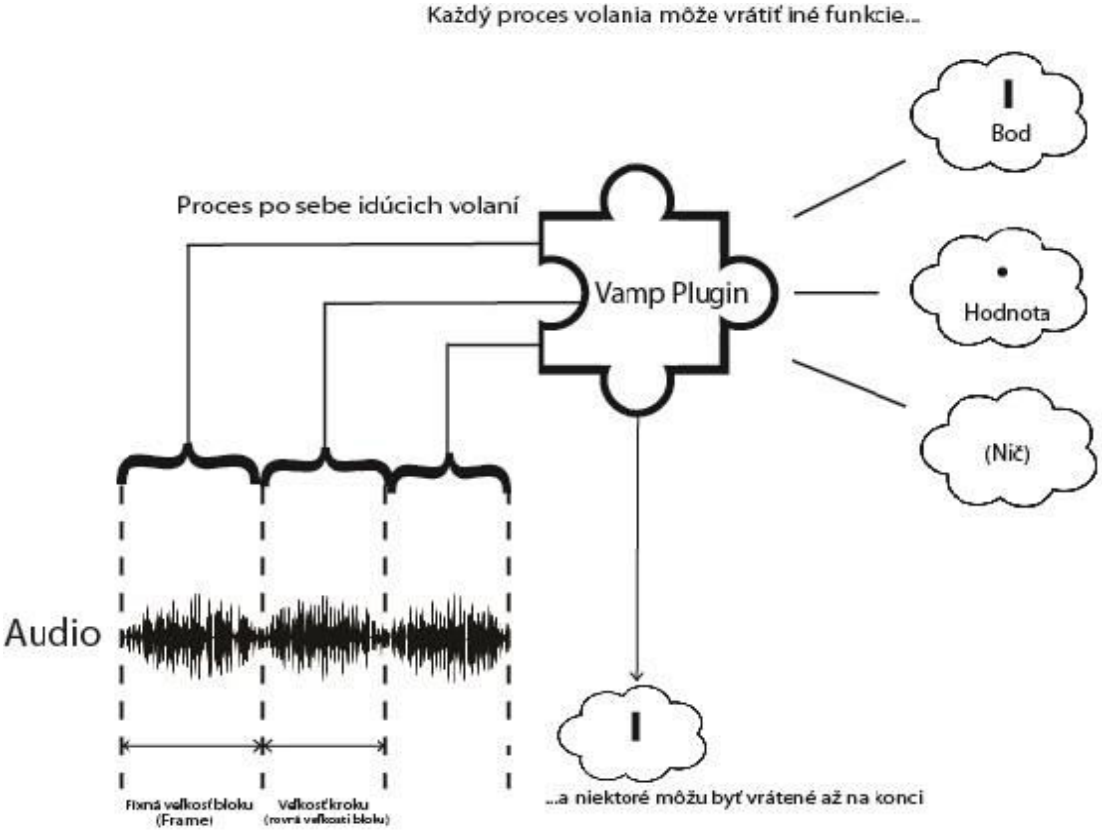
Tak ako pri počte kanálov, plugin môže ovplyvniť veľkosť kroku a veľkosť bloku vrátením požadovaných hodnôt prostredníctvom jeho metód getPreferdStepSize a getPreferdBlockSize. Na rozdiel od počtu kanálov, preferovaná veľkosť kroku a veľkosť bloku sú iba náznaky pre vás, preto by ste mali vždy skontrolovať skutočné hodnoty použité pri metóde initialise, ak sú dôležité pre váš kód. Netreba pre ne určovať preferencie, ak nechcete: čo znamená, že vrátia nulu a hostiteľ bude používať jeho predvolené hodnoty.

Zvuk môže byť poskytnutý buď vo forme *časovej domény* alebo vo *frekvenčnej doméne*. Zvukový vstup časovej domény je konvenčný digitálny zvuk s PCM vzorkovaním s typom vzorky s pohyblivou rádovou čiarkou. Vstup frekvenčnej domény je výsledkom aplikácie oknovej krátkodobej Fourierovej transformácie na každý vstupný blok. Vstupná doména je špecifikovaná pluginom pomocou metódy getInputDomain.

4.2.2.1 Časová doména vstupu (Time Domain Input)

Keď si plugin vyžiada vstup časovej domény, hostiteľ rozdelí vstupný prúd zvuku do série blokov rovnakej veľkosti a prisúdi jednu do každého následného volania metódy initialise. Volanie metódy process môže potom vrátiť funkcie odvodené z tohto vstupného audio bloku podľa jeho rozmeru. Argument inputBuffers(argument vstupného bufferu) pre metódu process bude smerovať do jedného poľa desatinných čísiel(array of floats) pre každý vstupný kanál. Napríklad inputBuffers [0] [blockSize-1] bude posledná zvuková vzorka v aktuálnom bloku pre prvý vstupný kanál.

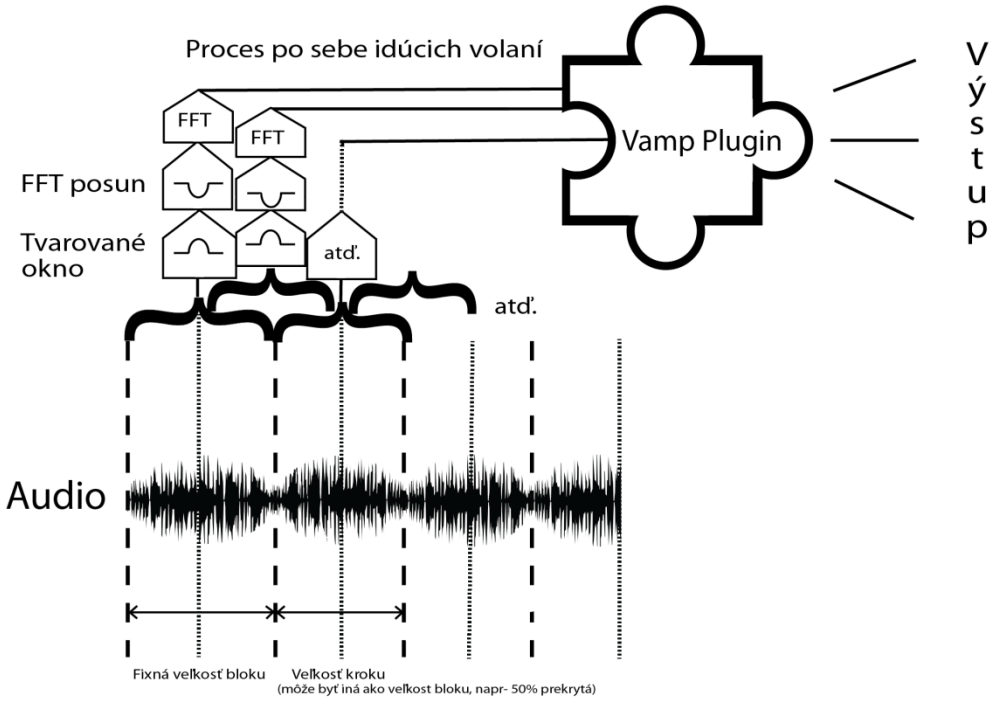
Keď boli všetky audio dáta dodané, hostiteľ zavolá metódu getRemainingFeatures a plugin vráti všetky funkcie, ktoré sú teraz známe a ešte neboli vrátené pri predchádzajúcich volaniach metódy process.



Obr.4.2: Vstup s časovou doménou

Pri dodávaní vstupu časovej domény je najviac obvyklé, aby sa veľkosť kroku rovnala veľkosti bloku, ako je znázornené vyššie. Znamená to, že plugin získava každú vzorku v audio vstupe presne raz, v rade súvislých blokov dát.
Môže však nastať aj iný prípad - plugin môže vrátiť inú hodnotu pre metódu getPreferredStepSize a inú hodnotu pre metódu getPreferredBlockSize, ak by z akéhokoľvek dôvodu uprednostnil prekrývanie alebo nesúvislé bloky. Vamp v súčasnosti neposkytuje žiadne čiastkové vstupné bloky. Ak audio vstup končí v strede bloku, hostiteľ vyplní blok nulovými hodnotami až do veľkosti bloku **.**

4.2.2.2 Frekvenčná doména vstupu (Frequency Domain Input)



Obr.4.3: Vstup s Frekvenčnou doménou

Ak plugin požaduje vstup frekvenčnej domény, každý blok audio vstupu spracováva hostiteľ pomocou oknovanej krátkodobej Fourierovej transformácie pred tým, než dodá zvukové dáta do metódy process. V tejto situácii je najčastejšie, ak sa vstupné bloky prekrývajú - to znamená, že veľkosť kroku je polovica veľkosti bloku alebo menšia. Je to preto, že pôvodné dáta časovej domény musia byť tvarované pomocou kosínusu alebo podobného okna pred aplikáciou Fourierovej transformácie, aby sa zabránilo generovaniu spektrálneho šumu kvôli diskontinuitám na okrajoch vstupných blokov. Toto „oknovanie“ je zodpovednosťou hostiteľa, ale plugin si musí byť vedomý toho, že sa to stane, aby si mohol vybrať rozumné preferované veľkosti krokov a veľkosti blokov.

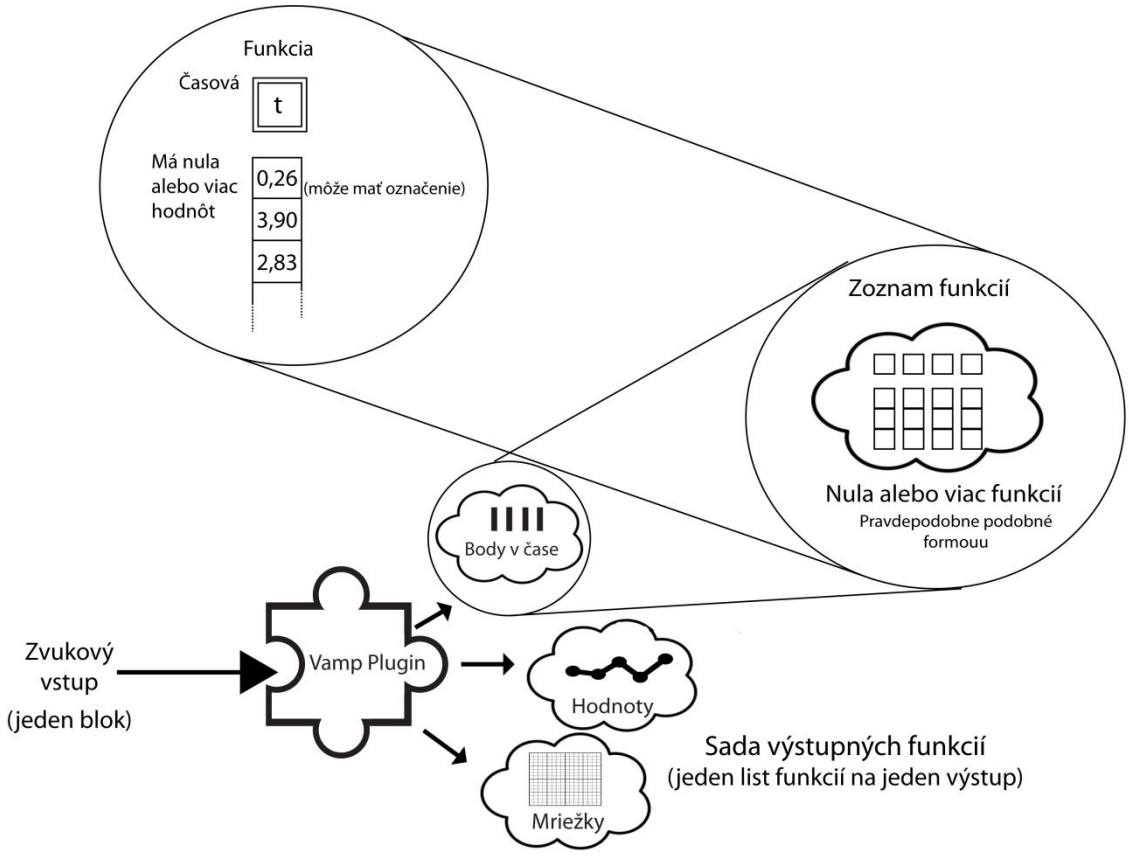
Plugin neobsahuje žiadnu kontrolu nad tým, ktorý tvar okna sa používa, alebo nad akýmikoľvek ďalšími podrobnostami o spracovaní časovej alebo frekvenčnej domény, okrem veľkosti kroku a veľkosti bloku. Ak je potrebná väčšia kontrola, budete musieť namiesto toho požiadať o vstup do časovej domény a vykonať spracovanie v plugine

Pri prijímaní vstupu frekvenčnej domény príkaz inputBuffers pre metódu process bude smerovať na jedno pole čísiel s pohyblivou desatinnou čiarkou(array of floats) pre každý vstupný kanál ako pri vstupe časovej domény, ale „polia float“ tu majú osobité rozloženie. Každý kanál obsahuje blockSize + 2 floats, ktoré tvoria striedavo reálne a imaginárne zložky komplexných výstupných zásobníkov Fourierovej transformácie.

4.2.3 Výstupy (Outputs)

4.2.3.1 Štruktúra funkcií

Plugin môže vrátiť funkcie na dvoch miestach: pri volaní metódy process a pri volaní metódy getRemainingFeatures. Volanie metódy process sa robí opakovane, aby sa pluginu poskytli vstupné dáta. Keď sa všetky vstupné dáta spotrebujú, privolá sa raz metóda getRemainingFeatures a plugin by mal vrátiť akékoľvek iné funkcie, ktoré vypočítal alebo môže teraz vypočítať, ale ešte nevrátil.



Obr.4.4: Výstup Vamp Plugin-u

Návratový typ metódy process a metódy getRemainingFeatures sa nazýva FeatureSet. Jedná sa o STL mapu, ktorej kľúč je výstupné číslo a ktorej hodnota je FeatureList, čo je vektor STL funkčných objektov. Použitie zoznamu funkcií umožňuje pluginu vrátiť funkcie s viac ako jednou časovou pečiatkou(TimeStamp) pri jedinom zavolaní funkcie process alebo vrátiť všetky funkcie pre celý audio vstup v jedinom FeatureSet-e z metódy getRemainingFeatures.

Funkcia Feature má voliteľnú časovú pečiatku a trvanie, vektor nuly alebo viacerých hodnôt a voliteľný štítok. Je potrebné si uvedomiť, že dokonca funkcia s nulovými hodnotami, bez časového označenia a bez štítku by mohla byť platnou funkciou. Môže to znamenať, že sa "niečo stalo v audio bloku prenesenom do tohto volania metódy process" s vysvetlením výrazu "niečoho" v závislosti od toho, ktorý výstup pluginu vrátil túto funkciu.

Veľa o interpretácii vrátených funkcií závisí od toho, s ktorým výstupom je funkcia spojená. Plugin má pevný počet výstupov a musí poskytnúť metódu getOutputDescriptors, ktorá vracia dáta o všetkých z nich ako vektor objektov OutputDescriptor. Deskriptor s nulovým indexom v tomto vektori obsahuje informácie o výstupe, ktorého hodnoty sa nachádzajú s nulovým kľúčom v súboroch funkcií vrátených doplnkom atď.[18]

5 REALIZÁCIA VAMP PLUGINU

Úlohou mojej práce bolo naprogramovať Vamp Plugin v jazyku C++, ktorý zobrazí Spektrogram daného zvukového signálu, taktiež zobrazí Chromagram daného signálu(čo je vlastne Spektrogram rozšírený o ďalšie výpočty) a určí tempo skladby respektíve zobrazí body v ktorých sa tempo zmenilo.

5.1 Efektívna hodnota RMS (Root mean square)

Root mean square (skrátene RMS alebo rms) je definovaná ako druhá odmocnina priemeru štvorca (aritmetický priemer štvorcov množiny čísel). RMS je tiež známe ako kvadratický priemer a je zvláštnym prípadom zovšeobecneného priemeru s exponentom 2. RMS môže byť tiež definované pre kontinuálne meniacu sa funkciu z hľadiska integrálu štvorcov okamžitých hodnôt počas cyklu.

My sa ale na RMS pozrieme z hľadiska zvukového signálu a teda z hľadiska frekvenčného. Hodnotu RMS možno vypočítať vo frekvenčnej oblasti pomocou Parsevalovej vety.
Pre vzorkovaný signál $x[n] = x(t = nT)$, kde T je perióda vzorkovania

$$\sum_{n=1}^{N} x^2[n] = \frac{1}{N}\sum_{m=1}^{N}|X[m]|^2 \tag{5.1}$$

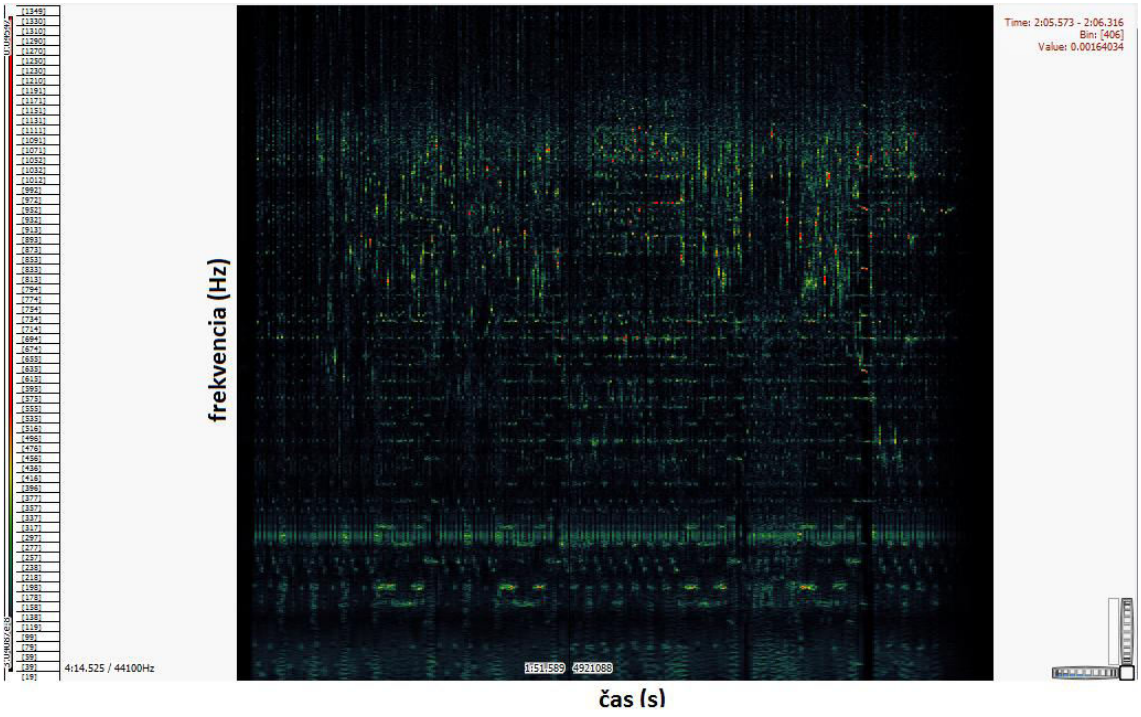
Kde $X[m] = FFT\{x[n]\}$ a *N* je počet vzorkov a FFT koeficientov.
V prípade, kedy je RMS počítaná v časovej doméne je rovnaký ako pri frekvenčnej doméne:

$$RMS\{x[n]\} = \sqrt{\frac{1}{N}\sum_n x^2[n]} = \sqrt{\frac{1}{N^2}\sum_m |X[m]|^2} = \sqrt{\sum_m \left|\frac{X[m]}{N}\right|^2} \tag{5.2}$$

[19]

5.2 Spektrogram

Spektrogram je vizuálne znázornenie spektra frekvencií zvuku alebo iného signálu, ktoré sa mení s časom alebo s inou premennou. Spektrogramy sa niekedy nazývajú spektrálne vodopády, hlasové odtlačky alebo hlasové grafy.



Obr.5.1: Zobrazenie spektrogramu (Skladba AC/DC- Back in Black)

Spektrogramy sa dajú použiť na fonetické rozpoznanie hovorených[20] slov a na analýzu rôznych volaní zvierat. Využívajú sa vo veľkom rozsahu vo vývoji v oblasti hudby, sonaru, radaru a spracovania reči, seizmológie a iných. Bežný formát je graf s dvomi geometrickými rozmermi: jedna os predstavuje čas alebo RPM(Revolutions per minute/ Otáčky za minútu),[21] druhá os je frekvencia. Tretia dimenzia indikujúca amplitúdu konkrétnej frekvencie v konkrétnom čase je reprezentovaná intenzitou alebo farbou každého bodu v obraze.

Spektrogramy sa zvyčajne vytvárajú jedným z dvoch spôsobov: aproximované ako filtračná banka, ktorá vyplýva zo série pásmových filtrov (toto bola jediná cesta pred príchodom moderného digitálneho spracovania signálu) alebo vypočítané z časového signálu pomocou Fourierovej transformácie. Tieto dve metódy v skutočnosti tvoria dve odlišné časovo-frekvenčné reprezentácie, ale za určitých podmienok sú ekvivalentné. Metóda pásmového filtrovania zvyčajne využíva analógové spracovanie na rozdelenie vstupného signálu do frekvenčných pásiem; veľkosť výstupu každého filtra ovláda menič, ktorý zaznamenáva spektrogram ako obrázok na papieri. [22]

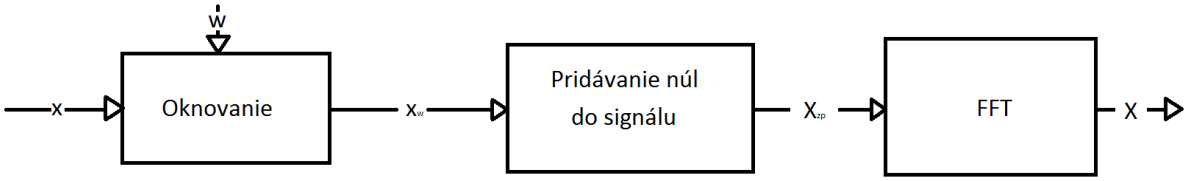
Vytvorenie spektrogramu pomocou FFT je digitálny proces. Digitálne vzorkované dáta v časovej oblasti sú rozdelené na kusy, ktoré sa obyčajne prekrývajú a sú Fourierovou transformáciou pretransformované na výpočet veľkosti frekvenčného spektra pre každý kus. Každý kus potom zodpovedá zvislej čiary v obraze. Tieto spektrá alebo časové grafy sú potom "položené bok po boku", aby vytvorili obraz alebo trojrozmerný povrch[23] alebo sa mierne prekrývajú rôznymi spôsobmi, t.j. oknovanie. Tento proces v podstate zodpovedá výpočtu štvorcovej veľkosti krátkodobej Fourierovej transformácie (STFT) signálu $s(t)$, to je pre okno šírky ω $spectogram(t, \omega) = |\mathrm{STFT}(t, \omega)|^2$ [24]

5.2.1 Spektrálna analýza

Pre oblasti, ktoré nie sú umiestnené v blízkosti tranzientov (prechodov) , je navzorkovaný zvukový signál analyzovaný s cieľom získať jeho reprezentáciu vo frekvenčnej oblasti. Hlavnými krokmi pre spektrálnu analýzu sú Windowing („Oknovanie/Voľba okna“), Zero padding („Pridávanie núl“) a výpočet DFT(„Diskrétna Fourierova transformácia“).
Navzorkovaný vstupný signál $x(n)$ je rozdelený na sériu analyzovaných rámcov o veľkosti N_{frame}

$$x(n + l \cdot N_{hop}) \tag{5.3}$$

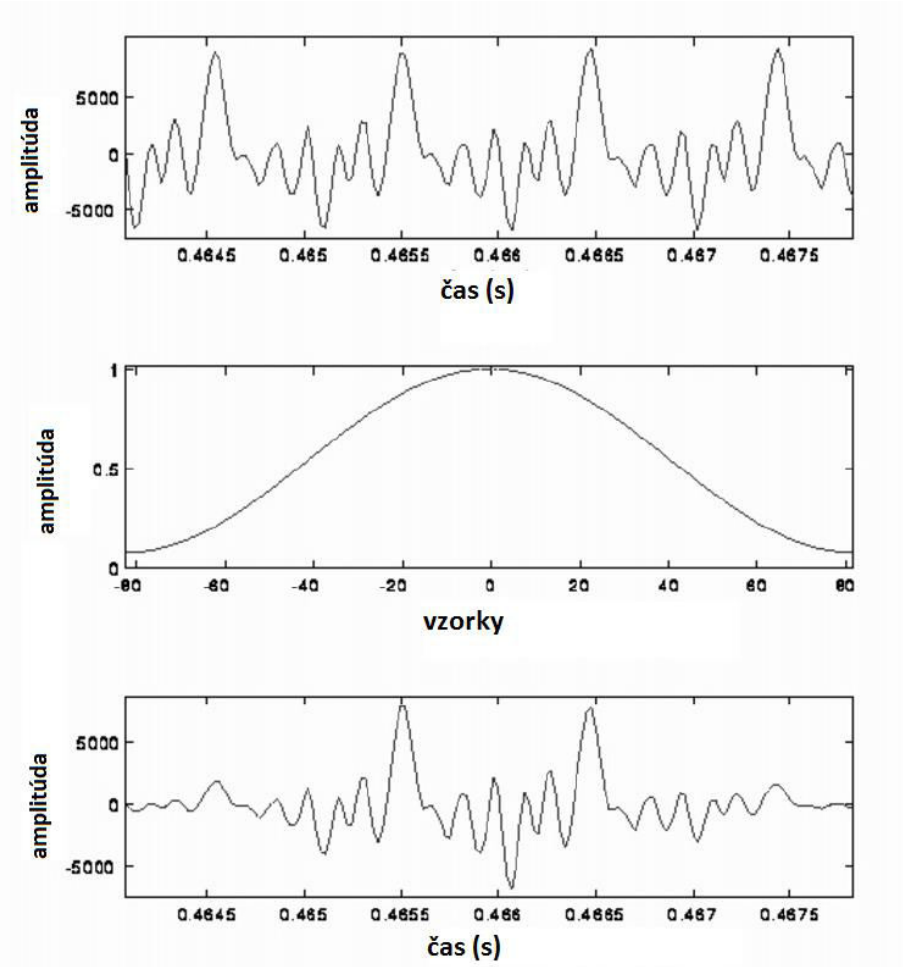
kde n = 0, 1, . . .N_{frame} − 1, l indikuje počet rámcov, ktoré sú analyzované a N_{hop} je časový posun pre každý rámec, meraný vo vzorkách alebo *„hop size“.*



Obr.5.2: Blokový diagram Spektrálnej analýzy

5.2.2 Oknovanie (Windowing)

Prvým krokom na vykonanie spektrálnej analýzy signálu časovej domény je zobrazenie signálu ako môžeme vidieť na obrázku č.5.3.



Obr.5.3: Zvukový rámec (delenie na bloky a oknovanie). **a.** Pozícia zvuku huslí použitého na analýzu aktuálneho rámca. **b.** Hammingovo okno. **c.** Oknovaný signál.

Každý rámec $x(n + l \cdot N_{hop})$ sa násobí oknovacou funkciou (windowing function) $\omega(n)$ pre získanie oknovaného signálu .$x_\omega(n)$.

$$x_\omega(n) = x(n + l \cdot N_{hop}) \cdot \omega(n) \tag{5.4}$$

kde *n*=0....N_{frame}-1.
Všetky štandardné okná sú reálne a symetrické a majú frekvenčné spektrum skrutkovitého tvaru. Voľba typu okna je ovplyvnená hlavne dvomi charakteristikami spektra: šírkou hlavného laloku a najvyššou úrovňou postranného laloku. Najbežnejšie okná sú obdĺžnikové okno (šírka hlavného laloku 2 zásobníky a úroveň bočného laloku rovná -13 dB), Hanningovo okno (hlavný lalok šírky 4 zásobníkov a úroveň bočného laloku rovnajúca sa -23 dB), Hammingovo okno (šírka hlavného laloku 4 zásobníky a bočný lalok s úrovňou rovnajúcou sa -43 dB), Blackman-Harris-ovo okno, Kaiserovo okno atď.
Nakoniec sú dáta spracované „oknovacou funkciou" sústredené do pôvodného času pred výpočtom DFT, aby sme získali okno s nulovou fázou.

$$x_{\omega c}(n) = x_\omega \left(n + \frac{N_{frame}}{2}\right) \tag{5.5}$$

kde $n = -\frac{N_{frame}}{2}, \dots \dots, \frac{N_{frame}}{2} - 1$.

5.2.3 Constant-Q transformácia

V matematike a spracovaní signálov constant-Q transformácia transformuje dátovú sériu na frekvenčnú doménu. Vzťahuje sa na Fourierovu transformáciu [25] a veľmi úzko súvisí s komplexnou transformáciou waveletov Morlet [26].
Transformácia sa môže považovať za sériu logaritmicky rozložených filtrov f_k, pričom k-ty filter má spektrálnu šírku δf_k rovnú násobku predchádzajúcej šírky filtra:

$$\begin{aligned} \delta f_k &= 2^{1/n} \cdot \delta f_{k-1} \\ &= \left(2^{1/n}\right)^k \cdot \delta f_{min} \end{aligned} \tag{5.6}$$

kde δf_k je šírka pásma k-teho filtru, f_{min} je stredná frekvencia najnižšieho filtra a n je počet filtrov na oktávu.

Krátkodobá Fourierova transformácia $x\,[n]$ pre rámec posunutý na vzorku m sa vypočíta nasledovne:

$$X[k,m] = \sum_{n=0}^{N-1} W[n-m]x[n]e^{-j2\pi kn/N} \quad . \tag{5.7}$$

Vzhľadom na dátovú sériu samplovanú pri frekvencii $f_s = 1 / T$, pričom T je samplovacia perióda našich údajov, pre každý frekvenčný zásobník môžeme definovať šírku filtra δf_k a faktor kvality Q rovnicou:

$$Q = \frac{f_k}{\delta f_k} \quad . \tag{5.8}$$

Toto je nižšie zobrazené ako celé číslo cyklov spracovaných na strednej frekvencii f_k. Dĺžka okna pre k-ty zásobník je daná vzťahom:

$$N[k] = \frac{f_s}{\delta f_k} = \frac{S}{f_k} \cdot Q \quad . \tag{5.9}$$

Keďže $\frac{S}{f_k}$ je počet vzoriek spracovaných za cyklus pri frekvencii f_k, Q je počet celých cyklov spracovaných na tejto centrálnej frekvencii. Ekvivalentnú Kernelovu transformáciu možno nájsť pomocou nasledujúcich substitúcií:

Dĺžka okna každého zásobníka je teraz funkciou čísla zásobníku:

$$N = N[k] = Q \cdot \frac{f_s}{f_s} \quad . \tag{5.10}$$

Relatívny výkon každého zásobníku sa zníži pri vyšších frekvenciách, pretože tieto súčty sú v menšom počte. Aby sme to kompenzovali, normalizujeme pomocou $N\,[k]$. Každá oknovacia funkcia bude funkciou dĺžky okna a podobne funkcia čísla okna. Napríklad ekvivalentné Hammingovo okno by bolo:

$$W[k,n] = \alpha - (1-\alpha)\cos\frac{2\pi n}{N[k]} \tag{5.11}$$

Pričom $\alpha = \frac{25}{46}$ a platí že $0 \leq n \leq N[k] - 1$. Z našej digitálnej frekvencie $\frac{2\pi k}{n}$ sa stane $\frac{2\pi Q}{N[k]}$.

Po všetkých týchto modifikáciách dostávame rovnicu constant-Q transformácie

$$X[k] = \frac{1}{N[k]} \sum_{n=0}^{N[k]-1} W[k,n]x[n]e^{\frac{-j2\pi Qn}{N[k]}} \quad . \tag{5.12}$$

5.2.3.1 Rozdiel medzi Fourierovou transformáciou a constant-Q transformáciou

Všeobecne platí, že táto transformácia je vhodná pre hudobné dáta, a to možno pozorovať v niektorých jej výhodách v porovnaní s rýchlou Fourierovou transformáciou. Keďže výstup transformácie je efektívne amplitúda / fáza proti logaritmickej frekvencii, je potrebných menej frekvenčných zásobníkov na efektívne pokrytie daného rozsahu, čo sa ukáže ako užitočné, keď sa frekvencie rozpínajú v rámci niekoľkých oktáv. Keďže rozsah ľudského sluchu pokrýva približne desať oktáv od 20 Hz do približne 20 kHz, toto zníženie výstupných údajov je významné.

Transformácia vykazuje zníženie frekvenčného rozlíšenia pri vyšších frekvenciách, čo je žiaduce pre sluchové aplikácie. Transformácia zrkadlí ľudský sluchový systém, pričom pri nižších frekvenciách je spektrálne rozlíšenie lepšie, zatiaľ čo časové rozlíšenie sa zlepšuje pri vyšších frekvenciách. V spodnej časti klavírnej stupnice (asi 30 Hz) je rozdiel 1 poltón rozdiel približne 1,5 Hz, zatiaľ čo v hornej časti hudobnej stupnice (asi 5 kHz) je rozdiel 1 poltónu rozdiel približne 200 Hz. [27] Takže pre hudobné dáta je ideálne rozlíšenie exponenciálnych frekvencií constant-Q transformácie.

Navyše, harmonické hudobných nôt tvoria vzor charakteristický pre farbu zvuku nástroja v tejto transformácii. Za predpokladu rovnakej relatívnej sily každej harmonickej, ako sa základná frekvencia zmení, relatívna poloha týchto harmonických zostáva konštantná. To môže značne uľahčiť identifikáciu nástrojov. Constant-Q transformácia môže byť tiež použitá na automatické rozpoznanie hudobných kľúčov na základe nahromadených informáciách o sýtosti tónov. [28]

V porovnaní s Fourierovou transformáciou je implementácia tejto transformácie oveľa zložitejšia. Je to spôsobené rôznym počtom vzoriek použitých pri výpočte každého frekvenčného zásobníku, čo tiež ovplyvňuje dĺžku každej implementovanej funkcie okna. [29]

5.2.4 Pridávanie núl (Zero padding)

Aby sme zvýšili frekvenčné rozlíšenie spektrálnej analýzy, musíme zvýšiť veľkosť rámcu N_{frame}. Možnosť ako to urobiť bez zvyšovania počtu vzorkou analyzovaných zo vstupného signálu $x(n + l * N_{hop})$, je získať N_{frame} vzorky z $x(n + l * N_{hop})$ a pridávať nulové vzorky dokým nedosiahneme veľkosť N_{FFT} zabezpečujúcu požadované frekvenčné rozlíšenie $\frac{f_s}{N_{FFT}}$. Počet pridaných núl je potom $N_{FFT} - N_{frame}$ a zero-padding faktor je definovaný ako pomer $\frac{N_{FFT}}{N_{frame}}$. Tento oknovaný signál s pridanými nulami je potom vstupným signálom pre výpočet Fourierovej transformácie.

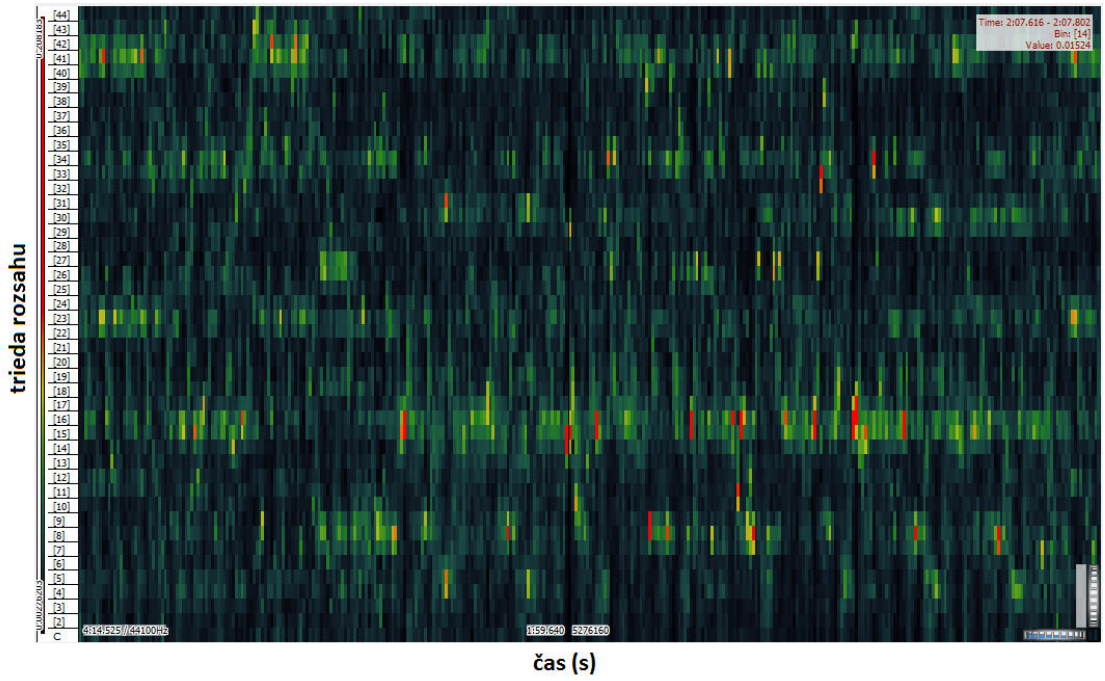
5.2.5 Rozlíšenie a parametre spektrálnej analýzy

Jedným z najdôležitejších parametrov frekvenčnej analýzy je veľkosť rámca analýzy N_{frame}. Kompromis medzi dobrým časovým rozlíšením (pomocou krátkych okien) a dobrým frekvenčným rozlíšením (s použitím dlhých okien) je jedným z problémov analýzy FFT[30]. Pri vykonávaní tónovej analýzy audio signálu je potrebné získať

dobré frekvenčné rozlíšenie pre definovanie veľkých rámcov. Používame veľkosť rámcu N_{frame}=4096 vzorkov čo je T=93ms pri vzorkovacej frekvencii 44,1kHz.

5.3 Chromagram

V kontexte hudby výraz chromatický znak alebo chromagram úzko súvisí s dvanástimi rôznymi triedami rozstupov. Funkcie na základe farebnosti, ktoré sú tiež veľmi blízke profilom triedy rozstupov(pitch class profiles), sú mocným nástrojom na analýzu hudby, ktorej výškové rozstupy môžu byť zmysluplne roztriedené (často do dvanástich kategórií) a ktorých ladenie sa približuje k rovnako temperovanej stupnici. Jedna hlavná vlastnosť funkcií farebnosti (chroma fukncií) je to, že zachycujú harmonické a melodické charakteristiky hudby, pričom sú odolné voči zmenám vo farbe a nástrojoch.



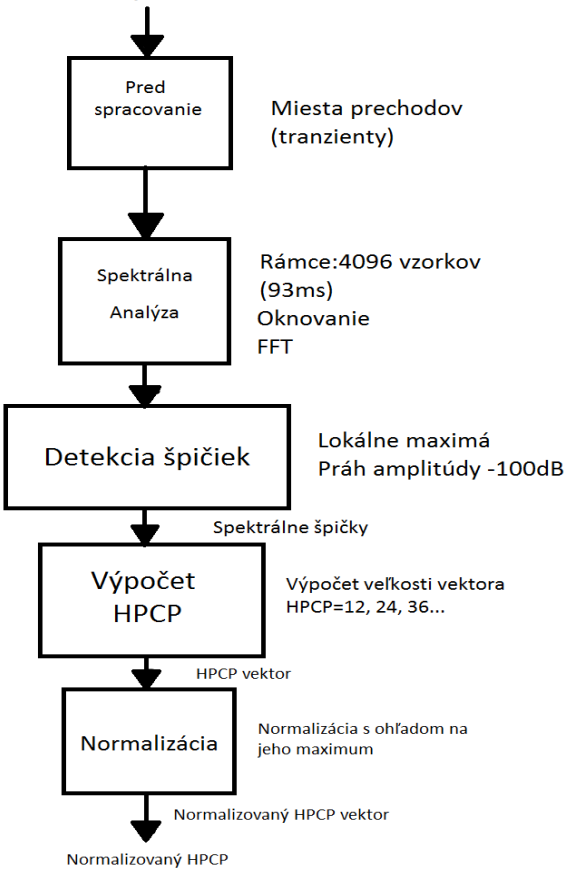
Obr.5.4: Zobrazenie Chromagramu (skladba: AC/DC- Back in Black)

Pozorovaním sa zistilo, že ľudia vnímajú zvuk dvoch tónov ako podobný vo farbe, ak sú od seba vzdialené o oktávu. Na základe tohto pozorovania môže byť tónový rozstup rozdelený na dve zložky, ktoré sa označujú ako výška tónu a farba (chroma). [31] Za predpokladu rovnovážnej stupnice je dvanásť chroma hodnôt reprezentovaných súborom tónov

{C, C♯, D, D♯, E ,F, F♯, G, G♯, A, A♯, B}

ktorý pozostáva z dvanástich atribútov hláskovania hudobného rozsahu, ako sa používa v západnej hudobnej notácii.
Vzhľadom na hudobnú reprezentáciu (napr. notová osnova alebo zvukový záznam) je hlavnou myšlienkou funkcií farby zhromaždiť pre dané lokálne časové okno (napr. v bitoch alebo v sekundách) všetky informácie, ktoré sa vzťahujú na danú farbu do jedného koeficientu. Posunutie časového okna cez reprezentáciu hudby má za následok postupnosť chromatických znakov, z ktorých každý vyjadruje rozloženie obsahu v časovom okne na dvanásť chromatických pásikov. Výsledná reprezentácia vlastností farby v čase sa tiež označuje ako chromagram.
Môj plugin s funkciou chromagramu vypočítava konštantnú Q spektrálnu transformáciu (rovnako ako v plugine Spektrogramu) a potom zakomponuje všetky hodnoty frekvenčných zásobníkov do jednej oktávy, pričom každý zásobník obsahuje súčet magnitúd zo zodpovedajúceho zásobníka vo všetkých oktávach. Počet hodnôt v každom vektorovom prvku vrátenom pluginom je teda rovnaký ako počet zásobníkov na oktávu konfigurovaných pre základnú konštantnú Q transformáciu.Pri zostrojovaní chromagramu musíme vykonať niekoľko výpočetných operácií a postupov.
Kostra výpočtu a zostrojenia chromagramu je zobrazená v obrázku č.5.4.

Obr.5.4: Blokový diagram výpočtu HPCP



1. Vloženie vstupného signálu(zvuková nahrávka)
2. Výpočet spektrálnej analýzy pre získanie frekvenčných častí zvukového signálu.
3. Použitie Fourierovej transformácie na prevod signálu do formy Spektrogramu. (Fourierova transformácia je typom časovo-frekvenčnej analýzy.)
4. Vykonať filtrovanie frekvencií s použitím frekvenčného rozsahu od 100Hz do 5000Hz.
5. Vykonať detekciu vrcholov/špičiek. Zvažované sú iba hodnoty lokálnych maxím spektra.
6. Vypočítať referenčnú frekvenciu. Odhad odchýlky vzhľadom ku hodnote 440Hz.
7. Vykonať mapovanie triedy rozsahov vzhľadom ku odhadovanej referenčnej frekvencii. (Toto je postup určenia hodnoty triedy rozsahov z frekvenčných hodnôt. Používa sa schéma váženia s kosínovou funkciou, pričom zohľadňuje prítomnosť harmonických frekvencií berúc do úvahy celkom 8 harmonických pre každú frekvenciu. Pre namapovanie hodnoty na jednu tretinu poltónu musí byť veľkosť distribučných vektorov triedy rozsahov rovná 36)
8. .Normalizácia funkcie rámec po rámci rozdelením maximálnej hodnoty aby sa eliminovala závislosť na celkovej hlasitosti. Ako výsledok dostávame normalizovaný Harmonický profil tried rozsahov(Chromagram).

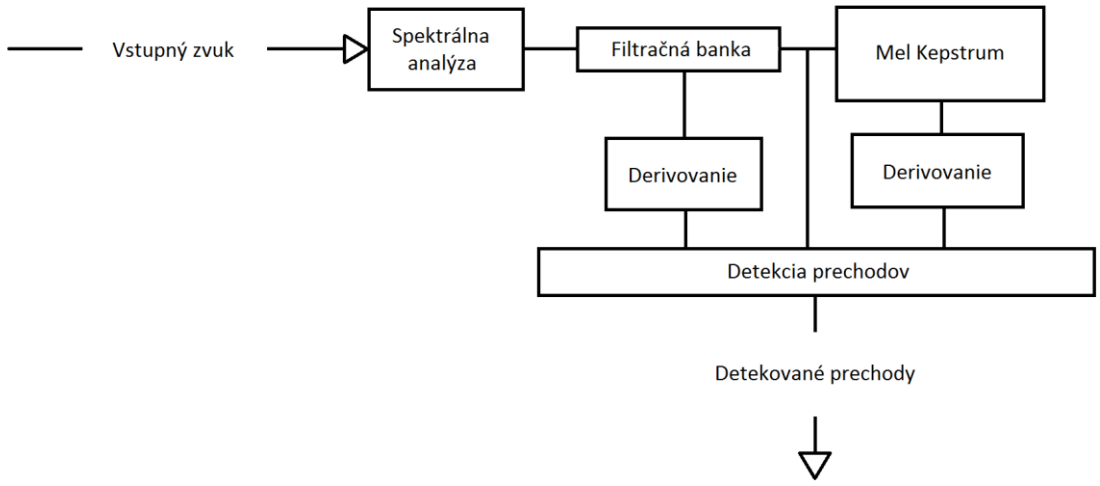
5.3.1 Predspracovanie (Pre-Processing)

Cieľom tohto kroku je pripraviť signál na výpočet distribučného vektora triedy skreslenia. Najskôr sa vykoná spektrálna analýza signálu (vytvorenie spektrogramu) aby sme získali reprezentáciu signálu vo frekvenčnej oblasti a niektoré ďalšie kroky, ktorými učiníme signál odolným voči hluku, vrátane procesu umiestnenia prechodov(tranzientov) a výberu frekvencie.

5.3.1.1 Lokalizácia prechodov

Ako prvý aplikujeme algoritmus na detekciu prechodov na odstránenie oblastí, v ktorých je harmonická štruktúra hlučná, takže oblasti, ktoré sú umiestnené 50ms pred a po prechodoch, sa neanalyzujú. Toto predspracovanie tiež znižuje výpočtové náklady na výpočet HPCP(Harmonic Pitch Class Profile) a pritom neovplyvňuje ani rozlíšenie ani efektívnosť deskriptorov tonality, takže presnosť odhadu ladenia zostáva podobná aj pri vyradení prechodov.[32]
V tejto metóde sú prechodné miesta lokalizované analýzou vývoja niekoľkých spektrálnych vlastností:

1.bankových filtračných energií
2. Mel cepstrum koeficientov a ich derivátov
Banka filtrov používaná v algoritme má 42 pásiem s frekvenciami medzi 40Hz a 20kHz, podľa stupnice Mel. Používa sa niekoľko jednoduchých pravidiel kombinujúcich tieto vstupy s cieľom nájsť body maximálneho nárastu sklonu.



Obr.5.5: Blokový diagram procesu hľadania prechodov

5.3.1.2 Spektrálna analýza

Existuje množstvo techník počítania funkcií sýtosti zo zvukových signálov, väčšinou sú založené na deformácii a balení magnitúdového spektra signálu. Jednou z nich je tá naša, ktorá využíva constant-Q transformáciu, spektrálnu analýzu kde sú kanály frekvenčnej domény logaritmicky rozmiestnené tak tesne pri sebe, že pripomínajú rozlíšenie frekvencie ľudského sluchového systému. Constant-Q transformáciu X_{cq} digitálneho signálu $x(n)$ môžeme vypočítať pomocou rovnice:

$$X_{cq}(k) = \sum_{n=0}^{N(k)-1} w(n,k)x(n)e^{-j2\pi f_k n} \quad (5.13)$$

kde analyzované okno w a jeho dĺžka N sú funkcie frekvenčného zásobníku pozície n. Stredná frekvencia k-teho zásobníku $f_k = 2^{k/\beta} f_{min}$, je definovaná vzhľadom ku frekvenciám rovnako temperovanej stupnici. β je počet zásobníkov na oktávu, čím sa definuje rozlíšenie analýzy, a f_{min} ktoré definuje bod začiatku analýzy frekvencie. Z $X_{cq}(k)$ môžeme vypočítať vektor sýtosti pre daný rámec vzorcom:

$$y(b) = \sum_{i=0}^{O-1} \left| X_{cq}(b + i\beta) \right| \quad (5.14)$$

kde $b \in 1, \beta$ je číslo zásobníku sýtosti, a O je celkový počet zvažovaných oktáv. Tak isto ako s krátkodobou Fourierovou transformáciou tak aj vektory sýtosti môžu byť

vypočítané postupne na čiastočne prekrývajúcich sa blokoch signálu čo vedie k vzniku chromagramu.[33]

5.3.1.3 Detekcia špičiek

Algoritmus detekcie špičiek bol vyvinutý v kontexte rámca Sinusoidal Modeling Synthesis (SMS), ktorý sa používal hlavne na reprezentovanie monofónnych signálov. [34] Fourierova veta uvádza, že rámec o N-vzorkách môže byť dokonale zastúpený N frekvenčnými zložkami a sínusový model predpokladá, že spektrum *X(k)* môže byť reprezentované menším počtom frekvenčných zložiek nazývaných sínusoidy. Sínusoida je frekvenčná zložka, ktorá je stabilná vo frekvencii aj amplitúde. Každá sínusoida predstavuje časť s dobre definovanou reprezentáciou, ktorou je Fourierova transformácia okna použitého na analýzu. Existujú isté interakcie medzi frekvenčnými komponentmi, ktoré sťažujú odhad čiastočnej hodnoty analýzou jediného spektra. V SMS je definovaný „vrchol" ako lokálne maximum v spektre magnitúdy, kde jediným obmedzením je to, že jeho frekvencia patrí do určitého rozsahu a jeho veľkosť je vyššia ako daná prahová hodnota. Kvôli vzorkovacej povahe spektra je každý vrchol presný len do polovice spektrálneho zásobníku. Spektrálny zásobník reprezentuje frekvenčný interval $\frac{f_s}{N_{FFT}}$ Hz kde fs je vzorkovacia frekvencia a N_{FFT} je veľkosť Fourierovej transformácie.[35]

Pridávanie núl(zero padding) zvyšuje frekvenčné rozlíšenie a tým aj presnosť jednoduchého detektora špičiek. Pri použití obdĺžnikového okna je požadovaný faktor nulovania 1000 ak chceme zvýšiť frekvenčné rozlíšenie na úrovni 0,1% šírky hlavného laloku okna.
Na zníženie tohto požadovaného faktora pridávania núl sa použije ďalší postup, pri ktorom sa vykonáva kvadratická spektrálna interpolácia s použitím len tých vzoriek, ktoré sú blízke frekvenčnej vzorke s maximálnou veľkosťou. V tomto postupe sa tri najbližšie body k frekvencii maximálneho rozsahu považujú za parabolu definovanú nasledujúcou rovnicou:

$$y(x) = a(x - p)^2 + b \tag{5.15}$$

kde p je stred paraboly, a je miera jeho konkávnosti a b je odchýlka.

5.3.1.4 Frekvenčné filtrovanie

Rovnako ako mnohé z postupov na určenie triedy rozsahov sme vykonali výber detekovaných spektrálnych špičiek podľa ich frekvencie. Zvažujeme len tie spektrálne špičky, ktorých frekvencia patrí frekvenčnému intervalu $f_i \epsilon [100{,}5000]\text{Hz}$,

bez ohľadu na vysoké frekvencie v našej analýze. Dôvodom je, že prevládajúce zvukové objekty sú v tejto oblasti hlučnejšie, kvôli nárazovému a inštrumentálnemu šumu (fúkaniu, trenie struny atď.). Výstupom tohto postupu je súbor spektrálnych špičiek $\{a_i, f_i\}, i = 1 \ldots . nPeaks$ ktoré majú hodnoty amplitúdy a_i a frekvenčné hodnoty f_i.
Tieto informácie sú vstupnými informáciami pre nasledujúce kroky výpočtu Profilu Triedy Rozsahov(PCP).

5.3.2 Stanovenie referenčnej frekvencie

Základom profilov triedy rozstupov je namapovanie frekvenčných hodnôt na triedy rozstupov podľa rovnovážnej stupnice a použitie danej referenčnej frekvencie f_{ref}. Dané kúsky signálu nie sú vždy naladené na štandardnú referenčnú frekvenciu 440Hz. Aby tónové charakteristiky nezáviseli od frekvencie ladenia, je potrebné odhadnúť túto referenčnú frekvenciu, to znamená frekvenciu použitú na vyladenie analyzovaného kusu. Postup stanovenia referenčnej frekvencie sa môže vykonať dvoma rôznymi spôsobmi: niektoré metódy odhadnú referenčnú frekvenciu pred mapovaním hodnôt frekvencie na hodnoty triedy rozstupov; iné algoritmy posunú rozdelenie funkcií triedy rozstupov ako krok následného spracovania, aby boli funkcie naladené na správnu referenčnú frekvenciu. V nasledujúcom postupe budeme používať prvý prístup, ktorý vykoná odhad referenčnej frekvencie pred výpočtom vektora rozdelenia triedy rozstupov. Referenčná frekvencia sa odhaduje pre každý analytický rámec analýzou odchýlky spektrálnych špičiek signálu vzhľadom na štandardnú referenčnú frekvenciu 440 Hz. Potom sa dosiahne celková hodnota kombináciou odhadov rámcov.

5.3.2.1 Okamžitá referenčná frekvencia

Nota je zložená z niekoľkých harmonických, ktorých frekvencie f_n sú násobkom základnej frekvencie f_0.

$$f_n = n \cdot f_0 \tag{5.16}$$

kde *n*=1...*N*.
Ak je základná frekvencia rozladená daným frekvenčným faktorom α tak sa frekvencie f_n zmenia
na $f_n^{'}$.

$$f_n^{'} = n \cdot \alpha \cdot f_0 \tag{5.17}$$

Interval (vyjadrený v poltónoch) medzi každou harmonickou frekvenciou a štandardnou referenčnou frekvenciou (440 Hz) môžeme potom vypočítať nasledovne:

$$\beta_n^{'} = 12 \cdot \log_2\left(\frac{n \cdot \alpha \cdot f_0}{440}\right) = \beta_n + 12 \cdot \log_2 \alpha = \beta_n + d \tag{5.18}$$

Interval medzi každou harmonickou a základnou referenčnou frekvenciou 440Hz možno rozdeliť na dva termíny: interval medzi základnou frekvenciou a 440Hz a interval medzi každou harmonickou a základnou frekvenciou takže dostaneme nasledujúci výraz pre $\beta_n^{'}$:

$$\beta_n^{'} = 12 \cdot \log_2\left(\frac{f_0}{440}\right) + 12 \cdot \log_2 n + d \tag{5.19}$$

$$\beta_n^{'} = \beta_0 + dev_n + d \tag{5.20}$$

Pre každú harmonickú definujeme nasledujúci pojem:

$$d_n = d + dev_n \tag{5.21}$$

Pojem $dev_n = 12 \cdot \log_2 n$ závisí počte harmonických. Hodnotu tejto odchýlky môžeme zložiť do jednej oktávy nasledovne:

$$dev_n^{'} = 12 \cdot \left(\log_2 n - round(\log_2 n)\right) \tag{5.22}$$

Harmonické $2f_0, 4f_0, 8f_0, \ldots$ sú presne zladené so základnou frekvenciou (napr. dev_n = 0), pretože vytvorený interval je rovnako temperovaná oktáva. Tento rozladovací faktor môžeme vypočítať pre rôzne harmonické dokonalého harmonického súboru, ako je uvedené v tabuľke č.5.1:

Harmonické	Frekvencia	dev_n (v centoch)
1	f_0	0
2	$2 \cdot f_0$	0
3	$3 \cdot f_0$	1,95
4	$4 \cdot f_0$	0
5	$5 \cdot f_0$	-13,69
6	$6 \cdot f_0$	1,95
7	$7 \cdot f_0$	-31,17
8	$8 \cdot f_0$	0

Tabuľka 5.1: Rozladovací faktor meraný v centoch, medzi prvými 8 harmonickými komplexného tónu a jeho základnou frekvenciou f_0.

Výhodou referenčného frekvenčného výpočtu je odhad faktoru rozlaďovania d v skutočnom (monofónnom alebo polyfonickom) spektre, kde významné špičky spektrogramu zodpovedajú hlavne harmonickým jednej alebo viacerých nôt. Potom sa rozlaďovací faktor *d* môže odhadnúť analýzou odchýlky spektrálnych špičiek vzhľadom ku štandardnej referenčnej frekvencii.[36]

5.3.3 Harmonický profil triedy rozstupov (Harmonic Pitch Class Profile)

Harmonický profil triedy rozstupov (HPCP) je funkcia distribúcie triedy rozstupov založená na profile triedy rozstupov (PCP). Tento vektor meria intenzitu každého z dvanástich polí diatónovej stupnice a získava sa mapovaním každého frekvenčného zásobníka spektra na danú triedu rozstupov. HPCP zavádza niektoré úpravy týkajúce sa výpočtu PCP: najprv sme zaviedli váhu do výpočtu funkcie; po druhé, zvažujeme prítomnosť harmonických; za tretie, v zásobníkoch HPCP používame vyššie rozlíšenie (zníženie úrovne kvantifikácie na menej ako polovicu). Zvažujeme teda iba tie spektrálne špičky, ktorých frekvencia patrí do intervalu $f_i[100, 5000]$Hz, bez ohľadu na vysoké frekvencie v našej analýze. HPCP vektor je potom definovaný nasledujúcim vzorcom:

$$HPCP(n) = \sum_{i=1}^{nPeaks} \omega(n, f_i) \cdot a_i^2 \qquad (5.23)$$

kde $n = 1 \ldots . size$, a_i *a* f_i *sú* hodnoty lineárnej veličiny a frekvencie čísla vrcholu i, $nPeaks$ je číslo spektrálnych špičiek ktoré zvažujeme, n je HPCP zásobník, $size$ je veľkosť HPCP vektoru (počet zásobníkov:12, 24, 36,...), a $w(n, f)$ je váha frekvencie f_i ak zvažujeme HPCP zásobník n.[37]

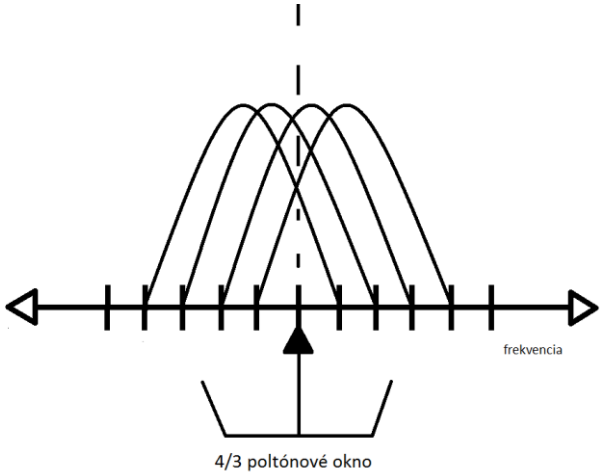
5.3.3.1 Funkcia váženia

Funkcia váženia $w\ (n, f_i)$ je definovaná nasledovne: namiesto prispievania do jedného zásobníka HPCP (napríklad najbližšieho), každá frekvencia f_i prispieva do zásobníka HPCP, ktorý je obsiahnutý v určitom okne v okolí hodnoty tejto frekvencie, ako je znázornené na obrázku 3.9. Pre každý z týchto zásobníkov je príspevok špičky i (štvorček maximálnej lineárnej amplitúdy $|\ a_{\ i}| = 2$) vážený pomocou funkcie $cos2$ okolo frekvencie zásobníka. Hodnota hmotnosti závisí od frekvenčnej vzdialenosti medzi f_i a stredovej frekvencie bin n, f_i, meraná v poltónoch, nasledovne:

$$f_n = f_{ref} \cdot 2^{\frac{n}{size}} \tag{5.24}$$

kde $n = 1 \ldots size$.

Tento postup váženia minimalizuje chyby odhadu, ktoré zisťujeme, keď sú v spektre prítomné rozdiely v ladení a neharmonizmus. Tieto faktory môžu vyvolať chyby pri mapovaní hodnôt frekvencie do zásobníkov HPCP.



Obr.5.6: Vážiaca funkcia

5.3.3.2 Váženie harmonických frekvencií

Spektrum noty sa skladá z niekoľkých harmonických, ktorých frekvencie sú násobkami základnej frekvencie (f, 2 · f, 3 · f, 4 · f atď.). Keď hráme jednu notu, spektrálne zložky sa objavujú na frekvenciách rôznych harmonických. Táto skutočnosť ovplyvňuje hodnoty HPCP, kde sa zvyšuje hodnota v rôznych zásobníkoch. Ak vezmeme do úvahy temperovanú stupnicu, môže sa hodnota i_n spojená s n-tou harmonickou nuly (ktorú môžeme nazvať n-tá trieda harmonického rozstupu) vypočítať takto:

$$i_n = mod[(i_1 + 12 \cdot \log_2(n)), 12] \tag{5.25}$$

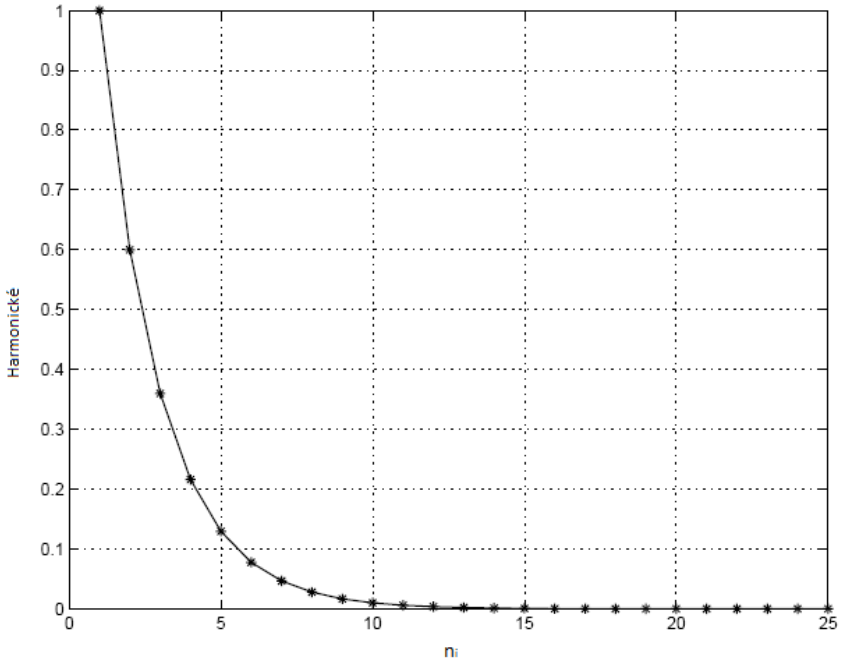
kde i_1 je trieda rozstupu, ktorá zodpovedá základnej frekvencii noty.

Aby sa harmonické podieľali na triede rozstupov ich základnej frekvencie, zavádzame funkciu váženia: každá frekvencia špičky f_i sa podieľa na frekvenciách f_i , ktoré majú harmonickú frekvenciu $(f_i, \frac{f_i}{2}, \frac{f_i}{3}, \frac{f_i}{4}, \ldots \frac{f_i}{nHarmonická})$.

Tento podiel znižujeme spolu s frekvenciou použitím nasledovnej funkcie:

$$\omega_{harm}(n) = s^{n-1} \tag{5.26}$$

kde s <1, aby sme simulovali, že amplitúda spektra klesá s frekvenciou. Táto funkcia je ukázaná v obrázku č.5.7 (kde s je nadstavené na hodnotu 0,6).
V ideálnom prípade by mala hodnota s závisieť na farbe zvuku nástroja.
[38][39][40]
Váha rôznych harmonických (s=0,6):



Obr.5.7: Vážiaca funkcia príspevkov harmonických frekvencií

5.3.3.3 Spektrálne bielenie

Funkcia váženia pre každú frekvenciu by mala byť prispôsobená spektrálnej obálke analyzovaného signálu. Aby sme tak urobili, vložíme do celého procesu výpočtu HPCP krok predbežného spracovania(pre-processing), v ktorom sa spektrum normalizuje podľa spektrálnej obálky, aby sa zmenilo na ploché spektrum. Pomocou tejto normalizácie stôp sa noty vo vysokých oktávach rovnomerne podieľajú na konečnom vektore HPCP ako tie, ktoré sú na nízkych rozstupoch a výsledky nie sú ovplyvnené rôznymi postupmi vyrovnávania.[41]

Tabuľka 5.2: Príspevky prvých 6 harmonických noty

Harmonické	Frekvencia	dev_n(v centoch)
1	f_0	1
2	$2 \cdot f_0$	s
3	$3 \cdot f_0$	s^2
4	$4 \cdot f_0$	s^3
5	$5 \cdot f_0$	s^4
6	$6 \cdot f_0$	s^5

5.3.4 Následné spracovanie (Post processing)

5.3.4.1 Normalizácia

Pre každý rámec analýzy sú hodnoty HPCP normalizované s ohľadom na svoju maximálnu hodnotu, aby sa zachovala relatívna relevancia každého z HPCP zásobníkov. Tento normalizačný proces spolu s fázou detekcie špičiek poskytuje nezávislosť na dynamike a celkovom objeme, ako aj na prítomnosť mäkkého šumu. Najprv sa vyberajú iba spektrálne špičky s vyššou hodnotou ako je prahová hodnota, takže ak je energia veľmi nízka, nedochádza k detekcii špičiek a vypočítaný vektor HPCP je plochý. Po druhé, ak sa spektrálna amplitúda vynásobí faktorom, výsledný vektor HPCP sa taktiež zmenší. Vplyv tohto faktora stupnice je potom eliminovaný normalizačným procesom.[28]

5.3.5 Štandard MIDI ladenia

Keďže v plugine s funkciou Spektrogramu sme pracovali s rozsahmi frekvencií v Hz ale pri plugine s funkciou Chromagramu už pracujeme s MIDI jednotkami, v tejto časti si objasníme prevod frekvencií na MIDI jednotky a naopak.
MIDI Tuning Standard (MTS) je špecifikácia precízneho hudobného rozstupu dohodnutého združením výrobcov MIDI v protokole MIDI. Služba MTS umožňuje hromadné hlásenie výpisu ladiaceho signálu, pričom poskytuje ladenie každej zo 128 nôt a ladiacu správu pre jednotlivé poznámky pri prehrávaní.
Ak f je frekvencia tak príslušná hodnota frekvenčných dát d sa môže vypočítať pomocou rovnice:

$$d = 69 + 12\log_2\left(\frac{f}{440\text{Hz}}\right) \quad . \tag{5.27}$$

Množstvo $\log_2\left(\frac{f}{440\text{Hz}}\right)$ je počet oktáv nad koncertným A 440-Hz (je záporný, ak je frekvencia pod týmto rozstupom). Vynásobením číslom 12 sa udáva počet polí nad touto frekvenciou. Pridaním 69 sa udáva počet poltónov nad C, päť oktáv pod stredným C. Vzhľadom na to, že 440 Hz je široko používaný štandard koncertného A (napr. USA, Veľká Británia) a keďže je v MIDI vyjadrený celým číslom 69 (deväť poltónov nad stredným C, čo je 60), dostávame reálne číslo, ktoré vyjadruje rozstup v súlade s MIDI a celočíselným zápisom, známy ako MIDI číslo noty.
Prevod z MIDI čísla noty (d) na frekvenciu (f) je daný nasledujúcim vzorcom:

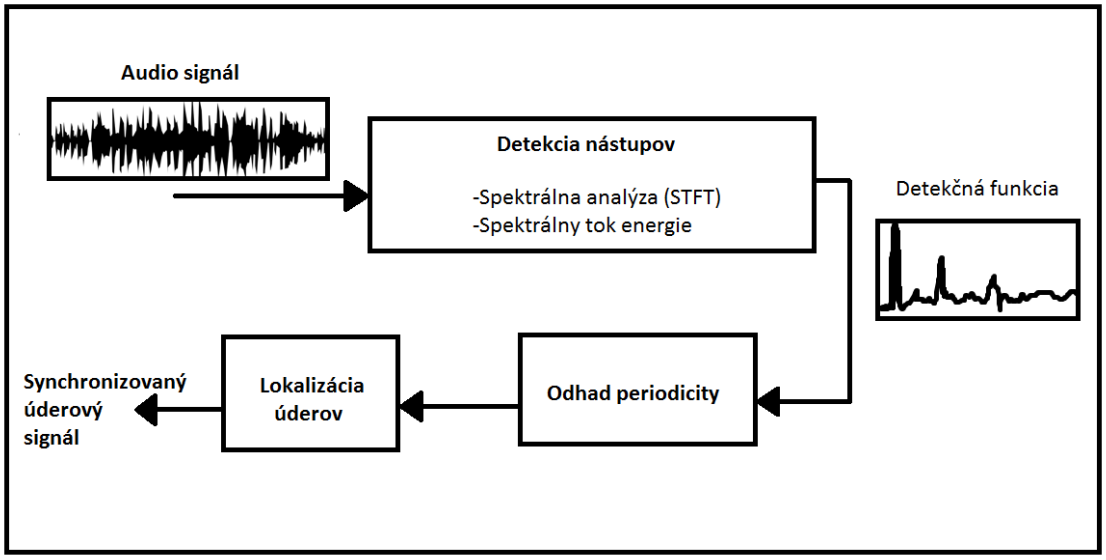
$$f = 2^{(d-69)/12} \cdot 440\text{Hz} \quad .[42] \tag{5.28}$$

Formát dátových frekvencií umožňuje presné zaznamenanie frekvencií, ktoré sa líšia od rovnakého temperamentu. Frekvenčné údaje sú definované v [jednotkách], ktoré sú zlomky poltónového. Frekvenčný rozsah začína na MIDI note č.0,ktorá zodpovedá tónu C = 8,1758 Hz a siaha až ku MIDI note č.127, ktorá zodpovedá tónu G = 12 543,875 Hz. Prvý bajt frekvenčného dátového slova špecifikuje najvyšší rovnako temperovaný poltón, ktorý nepresahuje frekvenciu. Nasledujúce dva bajty (14 bitov) určujú podiel 100 centov nad poltónom, na ktorom leží frekvencia. Účinné rozlíšenie = 100 centov / 214 = 0,0061 centov. "[43]
Toto vyššie rozlíšenie umožňuje logaritmické znázornenie rozstupu, v ktorom je poltón rozdelený na 1282 = 214 = 16384 častí, čo znamená, že oktáva je rozdelená na 196608 (logaritmicky) rovnakých častí. Tieto diely sú vo veľkosti presne 100/16384 centov (približne 0,0061 centov), čo je ďaleko pod prahom vnímania rozstupu ľudského sluchu čo umožňuje veľmi presné zobrazenie rozstupu tónov.

5.4 Určovanie tempa a úderov

Tento plugin má za úlohu zistiť tempo danej skladby, respektíve zobraziť graficky zmenu tempa (zrýchlenie/spomalenie) ak nejaká nastane.
Tempo odhad je základným problémom pri získavaní hudobných informácií (MIR), ktoré sú životne dôležité pre aplikácie v hudbe, ako je napríklad hudobná podobnosť a hudobné odporúčacie systémy, poloautomatické úpravy zvuku, automatické doprovody, polyfónne transkripcie,
úderovo-synchrónne zvukové efekty a počítačom podporované DJ systémy.
Automatická analýza rytmu sa typicky skladá z dvoch hlavných úloh: odhadu tempa a sledovanie úderov. Cieľom odhadu tempa je automaticky určiť rýchlosť hudobných úderov v čase. Údery môžu byť definované ako miesta v čase, v ktorých si ľudia "klepkajú" nohou pri počúvaní hudby. Úlohou sledovania úderov je určiť miesta týchto úderov v čase, čo je podstatne ťažšie v hudbe, v ktorej sa tempo časom mení [44].

Predpokladá sa, že tempo zvukového signálu je konštantné počas trvania analytického okna a že sa nakoniec pomaly vyvíja z jedného do druhého. Navyše predpokladáme, že tempo sa pohybuje medzi 60 a 200 BPM, bez straty všeobecnosti, pretože akúkoľvek inú hodnotu možno mapovať do tohto rozsahu. Navrhovaný algoritmus sa skladá z troch hlavných krokov (pozri obrázok 1):



Obr.5.8: Blokový diagram algoritmu určovania tempa

- **Detekcia nástupu**: pozostáva z výpočtu detekčnej funkcie založenej na spektrálnom toku energie vstupného audio signálu;
- **Odhad periodicity**: periodicita detekčnej funkcie sa odhaduje pomocou techník detekcie rozstupov;
- **Odhad polohy úderov**: pozícia zodpovedajúcich úderov sa získa z krížovej korelácie medzi detekčnou funkciou a umelým pulse-train-om. [45]

5.4.1 Detekcia nástupu

Cieľ detekcie nástupu spočíva v extrakcii detekčnej funkcie, ktorá bude indikovať umiestnenie najvýznamnejších vlastností zvukového signálu, ako sú zmeny nôt, harmonické zmeny a perkusné udalosti. V skutočnosti sú tieto udalosti obzvlášť dôležité v procese vnímania úderov. Noty sa ľahko zakryjú celkovou energiou signálu kontinuálnych tónov vyššej amplitúdy [46], zatiaľ čo sú pravdepodobnejšie detekované po ich oddelení vo frekvenčných kanáloch.

5.4.1.1 Spektrálna analýza a spektrálny tok energie

Vstupný zvukový signál sa analyzuje pomocou decimovanej verzie krátkodobej Fourierovej transformácie (STFT), tj krátke signálové segmenty sa extrahujú v pravidelných časových intervaloch, násobia sa analýzou a transformujú sa do frekvenčnej oblasti pomocou Fourierovej transformácie. Z toho dostávame rovnicu:

$$\tilde{X}(f,m) = \sum_{n=0}^{N-1} w(n)x(n+mM)e^{-j2\pi fn} \quad (5.29)$$

kde $x\ (n)$ označuje zvukový signál, $w\ (n)$ okno s konečnou analýzou veľkosti N danú v samploch, M vyjadruje $Hop\ Size$ danú v samploch, m index rámca a f frekvenciu. Spektrálny tok energie definujeme $E\ (f,k)$ ako aproximáciu derivácie obsahu frekvencie signálu vzhľadom na čas:

$$E(f,k) = \sum_m h(m-k)G(f,m) \quad (5.30)$$

kde $h\ (m)$ sa približuje diferenciátorovému filtru s:

$$H(e^{j2\pi f}) \simeq j2\pi f \quad (5.31)$$

a transformácia:

$$G(f,m) = \mathcal{F}\{|\tilde{X}(f,m)|\} \quad (5.32)$$

sa získa pomocou dvoch krokov: *nízkopásmové filtrovanie* $|\tilde{X}(f,m)|$ pre vykonanie energetickej integrácie podobným spôsobom ako v sluchovom systéme, pričom sa zdôrazňujú najnovšie vstupy, ale maskujú rýchle modulácie [47] a *nelineárnou kompresiou*. Úder sa väčšinou vyskytuje pri nástupoch nôt, takže najskôr musíme rozlíšiť vrcholy "true beat" od falošných, aby sme získali správnu funkciu detekcie. Okrem toho pracujeme za predpokladu, že tieto nežiaduce vrcholy sú oveľa menšie v amplitúde v porovnaní s vrcholmi nástupu nôt. Algoritmus výberu špičiek, ktorý vyberá špičky nad dynamický prah vypočítaný pomocou mediánu filtra, je jednoduchým a efektívnym riešením tohto problému. Stredový filter je nelineárna technika, ktorá vypočítava bodové stredné číslo v okienku s dĺžkou $2i + 1$ tvorenou podmnožinou $v\ (k)$, takže stredná prahová krivka je daná výrazom:

$$\theta(k) = C \cdot median(g_k) \quad (5.33)$$

Kde $g_k = \{V_{k-i}, \ldots . V_k \ldots \ldots, V_{k+i}\}$ a C je preddefinovaný faktor škálovania na umelé zvýšenie prahovej krivky mierne nad rovnovážnou úrovňou signálu. Na zaistenie presnej detekcie musí byť dĺžka stredného filtra dlhšia ako priemerná šírka špičiek detekčnej funkcie.

5.4.1.2 Odhad periodicity

Funkcia detekcie na výstupe v štádiu detekcie nástupov môže byť považovaná za kvázi-periodický a hlučný impulzný vlak, ktorý vykazuje veľké vrcholy pri nástupoch nôt. Ďalším krokom je odhadnutie tempa zvukového signálu, t.j. periodicita

impulzov nástupov nôt. Používajú sa dve metódy odvíjajúce sa od tradičných techník určovania rozstupov: *spektrálny produkt* a *auto korelačná funkcia*.
My sa bližšie pozrieme na autokoleračnú funkciu, ktorú používame aj v našom plugine.[48]

-Auto koleračná funkcia

Je klasickou metódou v odhade periodicity. Neformalizovaná deterministická auto korelačná funkcia $p\ (k)$ sa vypočíta takto:

$$r(\tau) = \sum_k p(k+\tau)p(k) \tag{5.34}$$

Opäť platí, že $60\ < T\ < 200\mathrm{BPM}$. Preto sa pri výpočte autokorelácie vypočítavajú len hodnoty $r\ (t)$ zodpovedajúce rozsahu od 300ms do 1s. Na nájdenie odhadovaného tempa T sa analyzuje oneskorenie troch najväčších vrcholov $r\ (t)$ a hľadá sa vzájomný vzťah medzi nimi. V prípade, že sa nenašiel žiaden vzťah, je oneskorenie najväčšej špičky považované za periódu úderu.

5.4.1.3 Poloha úderov

Ak chceme nájsť umiestnenie úderu, použijeme metódu založenú na myšlienke hrebeňového filtra. Vytvoríme umelú impulznú dráhu $q\ (t)$ tempa T, ktorú sme predtým vypočítali a krížovo korelovali s $p\ (k)$. Táto operácia má nízke výpočtové náklady, pretože korelácia sa hodnotí iba na indexoch zodpovedajúcich maximám $p\ (k)$. Potom zavoláme t_0, časový index, kde je táto krížová korelácia maximálna a považujeme ho za východiskovú pozíciu úderu. Pri druhom a ďalších po sebe nasledujúcich úderoch v analytickom okne sa do predchádzajúcej pozície úderu pridá perióda T a zodpovedajúci vrchol v $p\ (k)$ sa hľadá v oblasti $t_i - D$. Ak nie je nájdený žiaden vrchol, úder sa umiestni do jeho očakávanej polohy ti. Keď nastane posledný úder v danom okne, jeho umiestnenie sa uloží, aby sa zabezpečila kontinuita s prvým úderom nového analytického okna. Ak je tempo nového okna analýzy odlišné o viac ako 10% od predchádzajúceho tempa, odhaduje sa nová fáza. Špičky sa vyhľadávajú s použitím novej doby trvania bez odkazovania na predchádzajúcu fázu úderu.

5.5 Využitie funkcií pluginov v praxi

5.5.1 Spektrogram

Prvotné analógové spektrogramy sa aplikovali na širokú škálu oblastí, vrátane štúdia volaní vtákov , pričom súčasný výskum pokračoval v používaní moderných digitálnych zariadení a aplikoval sa na všetky zvuky zvierat. Súčasné používanie digitálneho spektrogramu je zvlášť užitočné pri štúdiu frekvenčnej modulácie (FM) v hovore zvierat. Konkrétne sú so spektrogramom najľahšie vizualizované rozlišujúce FM charakteristiky cvrlikania, širokopásmové kliknutia a sociálna harmonizácia.
Spektrogramy sú nápomocné pri prekonávaní deficitu reči a pri výučbe reči pre časť obyvateľstva, ktorá je hlboko hluchá . Štúdie fonetiky a syntézy reči sú často uľahčené použitím spektrogramov. Reverzáciou procesu vytvárania spektrogramu je možné vytvoriť signál, ktorého spektrogram je ľubovoľný obraz. Táto technika môže byť použitá na skrytie obrazu v zvuku a bola použitá niekoľkými umelcami elektronickej hudby. Niektorá moderná hudba sa vytvára pomocou spektrogramov ako stredného média, zmenou intenzity rôznych frekvencií v čase alebo dokonca vytváraním nových, ich kreslením a potom inverznou transformáciou.
Spektrogramy môžu byť použité na analýzu výsledkov odovzdávania skúšobného signálu cez signálny procesor, ako je filter, aby sa skontroloval jeho výkon. Spektrogramy s vysokým rozlíšením sa používajú pri vývoji RF a mikrovlnných systémov .V dnešnej dobe sa spektrogramy používajú na zobrazenie parametrov rozptylu meraných vektorom sieťových analyzátorov. Americký geologický prieskum momentálne poskytuje zobrazovanie spektrogramov zo seizmických staníc v reálnom čase. Spektrogramy môžu byť taktiež použité s rekurentnými neurónovými sieťami na rozpoznávanie reči. V hudbe sa spektrogramy používajú či už na identifikáciu nástrojov použitých v skladbe, pri vizuálnom zobrazení spevu a dynamiky jeho jednotlivých častí s ktorou môžeme následne pracovať alebo na celkové zobrazenie dynamiky určitej skladby vďaka ktorému môžeme vidieť, ktoré pasáže skladby sú prebudené a následne ich upraviť.[49]

5.5.1.1 Porovananie Spektrogramov z programov Matlab a Sonic Visualiser

Spektrogram z programu Matlab je v prvom rade oveľa jednoduchší pre vytvorenie, respektíve je už vytvorený a stačí si ho iba privolať vhodnou funkciou. Spektrogram z programu Sonic Visualiser je vo svojej podstate rovnaký ako spektrogram, ktorý môžeme vytvoriť v programe Matlab, avšak využitie v Sonic

Visualiseri je prehladnejšie, môžeme z daného zobrazenia odčítať hodnoty spektra pre zvolený zásobník a práca v tomto programe je celkovo viac prispôsobená laikom a ľudom so záujmom o modifikáciu či skúmanie zvukového signálu.
V programe matlab sa využíva zobrazenie spektrogramu skôr všeobecne pre grafické zobrazenie akéhokoľvek signálu vrátane zvukového, pričom ale nemôžeme z daného zobrazenia vyčítať presné hodnoty daných zásobníkov. Naopak výhodou spektrogramu v programe Matlab je jeho ostrejšie zobrazenie.

5.5.2 Chromagram

Identifikáciu rozsahov, ktoré sa líšia aspoň o oktávu, chroma vlastnosti (darebné vlastnosti) ukazujú vysoký stupeň odolnosti voči zmenám vo farbe zvuku a úzko súvisia s hudobným aspektom harmónie. To je dôvod, prečo sú chroma funkcie dobre zavedeným nástrojom na spracovanie a analýzu hudobných dát. Napríklad v podstate každý postup rozpoznávania akordov sa opiera o nejaký druh chroma reprezentácie. Chroma funkcie sa tiež stali de facto štandardom pre úlohy, ako je zosúladenie hudby a synchronizácia, ako aj analýza štruktúry zvuku. Nakoniec, vlastnosti chroma sa ukázali ako veľmi mocné pri zobrazení strednej úrovne v obsahu na báze audio vyhľadávania, ako je identifikácia titulnej piesne alebo zvuková zhoda.[50]

5.5.3 Určovanie tempa

Tempo odhad má mnoho aplikácií, ako je napríklad výroba beatmixov zložených z po sebe idúcich hudobných skladieb spracovaných dokopy pomocou vyrovnávania tempa a rozťahovania skladieb v čase. V DJ-ských aplikáciách možno použiť metrické informácie na automatické vyhľadanie bodov vhodných na vytvorenie slučiek. Vizuálnu využitie možno pridať do hudobných prehrávačov s úderovo synchrónnymi vizuálnymi efektmi, ako sú virtuálne tanečné znaky. Medzi ďalšie aplikácie patrí nájdenie hudby s určitým tempom z knižníc digitálnej hudby s cieľom prispôsobiť ich nálade poslucháča alebo poskytnúť vhodnú motiváciu pre rôzne fázy športových aktivít. Okrem toho môžu byť automaticky extrahované údery skladby použité na umožnenie hudobne synchronizovanej extrakcie znakov napríklad na účely štrukturálnej analýzy alebo na identifikáciu cover verzií skladieb.[51]

6 VYUŽITIE PLUGINOV V PROGRAME SONIC ANOTATOR

Sonic Annotator (predtým známy ako "Runner") je program na báze príkazového riadku pre dávkové extrahovanie zvukových funkcií z viacerých zvukových súborov. Základnou myšlienkou je odstrániť proces extrakcie funkcií z extrakčných metód pomocou Vamp pluginov pre extrakciu funkcií. Výstupné formáty sú RDF, CSV a ďalšie avšak nové výstupné formáty môžu byť pridané s miernym množstvom programovacích prác. Sonic Annotator je koncipovaný tak, že má v rámci projektu OMRAS2 dve hlavné využitia. Po prvé, je užitočný pre vytvorenie dát zverejniteľných v rámci sémantického webu pomocou Hudobnej ontologie a Ontológie zvukových funkcií. Po druhé, môže byť použitý na vytvorenie zoznamov funkcií pre použitie s AudioDb.

Najzrejmejšie zamýšľané použitie Sonic Annotator-u slúži na zverejnenie funkcií údajov o zbierke zvukov - daných súborom zvukových nahrávok, pravdepodobne uložených v jednom mieste , a taktiež umožňuje súbor špecifikácií funkcie automaticky extrahovať a uverejňovať na použitie tretími stranami. Typické funkcie môžu zahŕňať odhad tempa a odhady ladenia, lokalizácie úderov, segmentácie atď. - súbor dostupných funkcií nezávisí od Sonic Annotator-u, ale od oblasti dostupných Vamp pluginov.

Ak chcete používať Sonic Annotator, musíte mu povedať tri veci:
1.aké audio súbory použiť na extrahovanie funkcií
2. aké funkcie získať
3. ako a kde napísať výsledky.
Môžete mu tiež voliteľne povedať, aby zhrnul všetky funkcie.[52]

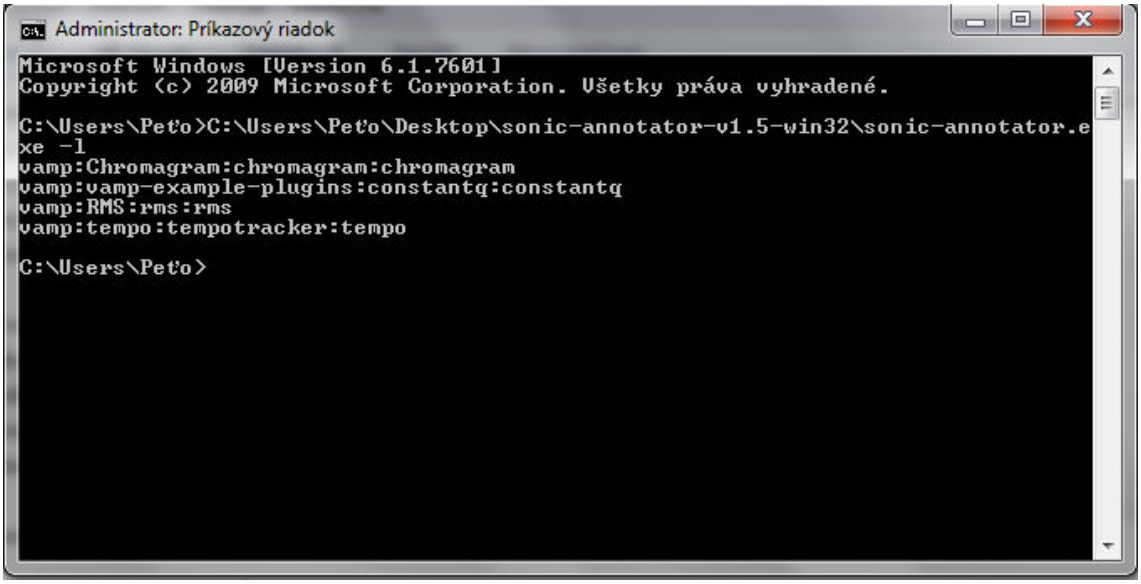
6.1 Aké audio súbory použiť na extrahovanie funkcií

Sonic Annotator prijíma zoznam zvukových súborov na príkazovom riadku. Akýkoľvek argument, ktorý nie je chápaný ako podporovaná možnosť príkazového riadku, bude považovaný za názov zvukového súboru. Môže byť uvedený ľubovoľný počet súborov.
Niektoré bežné formáty zvukových súborov sú podporované, vrátane MP3, Ogg a niekoľkých formátov PCM, ako sú WAV a AIFF. AAC je podporovaný iba v systéme OS / X a iba vtedy, ak nie je chránený systémom DRM. WMA nie je podporovaná.

Súborové cesty nemusia byť lokálne; môžete tiež poskytnúť vzdialené HTTP alebo FTP adresy pre Sonic Annotator na načítanie.
Sonic Annotator tiež prijíma názvy súborov playlistu (s príponou .m3u) a spracuje každý súbor nachádzajúci sa v zozname skladieb.
Obmedzenie aktuálnej verzie softvéru Sonic Annotator v systéme Windows spočíva v tom, že sa namiesto spätného lomítka ("\") vyžaduje ako línia oddeľovania cesty ("/") predbežné lomítko, aby sa zabránilo zápisu nesprávnych adries URL do výstupu v režime zapisovača RDF. Napríklad C: /audio/testfile.wav.
Napokon môžete namiesto súboru poskytnúť lokálnu cestu k adresáru spolu s voľbou -r (rekurzívna), aby Sonic Annotator spracoval každý zvukový súbor nachádzajúci sa v tomto adresári alebo niektorom z jeho podadresárov.

6.2 Aké funkcie extrahovať

Sonic Annotator aplikuje "transformuje" na vstupné zvukové súbory, kde transformácia (v tejto terminológii) pozostáva z pluginu Vamp spolu s určitou sadu parametrov a kontextu špecifikovaného spustenia vrátane veľkosti kroku a bloku, vzorkovacej frekvencie atď.
Ak chcete Sonic Annotator používať normálne, musíte vytvoriť súbor, ktorý opisuje vlastnosti transformácie, ktorú chcete použiť, a potom o tom informujte Sonic Annotator dodaním názvu súboru transformácie na príkazovom riadku s voľbou -t. K dispozícii je aj rýchly spôsob použitia predvolenej konfigurácie doplnku.
Transformácie sú zvyčajne popísané v RDF, po transformácii časti ontológie pluginov Vamp. Transformácia môže používať ľubovoľný plugin Vamp, ktorý je v súčasnosti nainštalovaný a dostupný v systéme.
Zoznam dostupných výstupov pluginov môžete získať spustením programu Sonic Annotator pomocou možnosti -l alebo --list:



Obr.6.1: Zoznam pluginov zobrazený cez Sonic Annotator

6.3 Ako a kam zapísať výsledky

Sonic Annotator podporuje rôzne výstupné moduly (a pre vývojárov je jednoduché pridať nové). Musíte si vybrať aspoň jeden z výstupných modulov použitím voľby *-w* (writer). Každý modul má vlastnú sadu parametrov, ktorú je možné nastaviť na príkazovom riadku, ako aj vlastné predvolené pravidlá o tom, kde je možné napísať výsledky.

Typy výstupov:
Csv- Zapíše výsledky do dátových súborov oddelených čiarkou. Vygenerované údaje pozostávajú z jedného riadka pre každú funkciu výsledku, obsahujúcej časovú pečiatku funkcie, dĺžku funkcie, ak je prítomná, všetky hodnoty zásobníku v danom poradí, za ktorým nasleduje štítok funkcie, ak je prítomný. Ak je zadaná voľba *-csv-one-file* alebo *-csv-stdout*, pred každým z vyššie uvedených sa zobrazí ďalší stĺpec obsahujúci názov zvukového súboru, z ktorého bola funkcia extrahovaná, ak sa líši od názvu súboru predchádzajúceho riadku. Ak chcete potlačiť tento ďalší stĺpec, použite voľbu *--csv-omit-filenames*.

Rdf- Jeden súbor je vytvorený pre každý vstupný zvukový súbor obsahujúci funkcie extrahované všetkými transformáciami použitými na tento súbor, pomenovaný po vstupnom audio súbore s príponou .n3, umiestnený v rovnakom adresári ako audio súbor. Aplikácia Sonic Annotator použije opis RDF doplnku, ak je k dispozícii na zvýšenie jeho výstupu (napríklad identifikácia časov začiatku tónu ako časov

začiatku tónu, ak RDF doplnok hovorí, že to je to, čo produkuje, namiesto toho, aby ich písal ako obyčajné udalosti). Najlepšie výsledky získate, ak je dokument RDF dodaný s vašimi pluginmy (napríklad vamp-example-plugins.n3) a máte ho nainštalovaný na rovnakom mieste ako vaše Vamp pluginy. Ak chcete tento zvýšený výstup prepísať a napísať jasné udalosti pre všetky funkcie, použite *-rdf-plain.*
Výstup RDF bude obsahovať vlastnosť *available_as* spájajúcu výsledky s pôvodným URI audio signálu. V predvolenom nastavení sa bude zobrazovať URI súboru alebo zdroja obsahujúceho zvuk, ktorý spracoval Sonic Annotator, napríklad umiestnenie súboru: /// na disku. Ak chcete toto prepísať, napríklad na spracovanie lokálnej kópie súboru pri generovaní RDF, ktoré popisuje jeho kópiu dostupnú v sieti, môžete použiť voľbu *--rdf-signál-uri*, aby ste určili URI alternatívneho signálu.

Midi- Napíše výsledky do súborov MIDI. Všetky funkcie sú napísané ako MIDI noty. Ak má atribút aspoň jednu hodnotu, jeho prvá hodnota sa použije ako rozsah noty, druhá hodnota (ak je k dispozícii) pre rýchlosť. Ak má daná funkcia jednotky Hz, potom jej rozstup bude premenený z frekvencie na celočíselnú hodnotu v rozsahu MIDI, inak bude napísaný priamo.
Viaceré (až 16) transformácie môžu byť zapísané do jedného súboru MIDI, kde budú mať oddelené čísla MIDI kanálov.[53]

6.4 Využitie pluginov prostredníctvom programu Sonic Annotator

Sonic Anotator môžeme použiť na vytvorenie RDF súboru, ktorý obsahuje kostru transformácie (pluginu). Tento RDF súbor môžeme potom v Sonic Anotatore použiť na úpravu akéhokoľvek podporovaného audio súboru. V podstate nám Sonic Visualiser poskytuje aplikáciu Vamp pluginov bez vizuálneho zobrazenia t.j. ak aplikujeme napríklad plugin určovania tempa tak Sonic Anotator vypíše v ktorom časovom okamihu (v sekundách) sa zmenilo tempo a vypíše aj danú zmenu, všetko však iba v číselných hodnotách. Tieto hodnoty môžeme neskôr použiť pre ďalšie výpočty bez náročného odčítavania z vizuálneho zobrazenia ako je to v programe Sonic Visualiser. Výsledné hodnoty transformácie vypísané programom môžeme uložiť do súborov s rôznou koncovkou v závislosti na ďalšom využití(csv, rdf, midi atď).

ZÁVER

V závere mojej semestrálnej práce by som sa chcel vyjadriť k samotnej práci na plugine a ku dosiahnutým výsledkom. Úlohou tejto semestrálnej práce bolo naprogramovať Vamp Plugin v prostredí C++, ktorý bude vedieť vypočítať a vizuálne zobraziť Spektrogram a od neho odvíjajúci sa Chromagram vstupného signálu, bude vedieť vypočítať Strednú efektívnu hodnôt s parametrom segmentácie a určí tempo (prípadne jeho zmeny) daného signálu. Najskôr som v práci rozobral pojem Získavanie informácii z hudby, ktorého pochopenie je nutné k ďalšej práci na plugine. V tejto časti som uviedol ako sa získavajú informácie z hudby, rôzne metódy a spôsoby využitia. Taktiež som rozobral obsahovo orientované metódy na spravovanie hudby súčasťou ktorých je aj parametrizácia hudobnej nahrávky. Znalosti tejto problematiky boli pri programovaní Vamp Pluginu nevyhnutné. Po uvedení konkrétnych príkladov parametrizačných programov, do ktorých som následne implementoval realizovaný plugin som tiež podrobne rozobral samotnú štruktúru Vamp Pluginu, z čoho sa skladá a čo je dôležité vedieť pri jeho programovaní. Po udelení tejto témy bakalárskej práce bolo pre mňa výzvou dosiahnuť pozitívne výsledky, keďže som s programovaním nemal doposiaľ príliš veľké skúsenosti. Napriek tomu sa mi podarilo úspešne zrealizovať fungujúci Vamp plugin a ako výsledok mojej práce uvádzam 4 pluginy, ktoré dokážu vypočítať efektívnu hodnotu signálu s voľbou veľkosti segmentu, taktiež zobrazí Spektrogram signálu s rôznymi možnosťami nadstavenia parametrov, Chromagram s rôznymi nastaviteľnými parametrami a v neposlednom rade vypočíta tempo daného vstupného zvukového signálu a zobrazí každú zmenu v tempe skladby. Dané pluginy sú plne funkčné, nastaviteľné a majú veľa možností využitia v hudobnom priemysle či už pri spracovaní hlasových prejavov alebo pri rozpoznávaní nástrojov obsiahnutých v skladbe.

RESUMÉ

At the end of my semester work, I would like to comment on the plug-in itself and the results. The role of the semester work was programming Vamp plugin in C ++ to be able to calculate and visualize spectrogram and from him, linked to Chromagram input signal, will be able to calculate the mean effective values of parameters segmentation and determine the pace (or changes) the signal. I first dealt with the concept of Getting Information from Music, which understanding is necessary for further work on plugins. In this section, I mentioned how to get information from music, different methods and ways of using it. I also devised content-oriented methods for music management, which include parameterisation of music recording. Knowledge of this issue was essential to Vamp Plugin programming. After placing concrete examples of parameter setting programs to which I subsequently implemented the plugin I realized also closely analyzed the structure itself Vamp plugins of which it is composed and what is important to know when programming. After having given this bachelor thesis, I was challenged to achieve positive results, as I did not have too much experience with programming. Nevertheless, I managed to successfully implement the functioning Vamp plugin and as a result of my work I show 4 plugins that can calculate the effective value of the signal with a choice of sizes segment also shows spectrogram signal with different options adjusted parameters, Chromagram with various adjustable parameters and not least the calculated rate of the input audio signal and displays each change in the track temperature. The plugins are fully functional, adjustable, and have a lot of possibilities to use in the music industry, either in speech processing or recognizing the tools contained in the song.

7 BIBLIOGRAFIA

[1] Downie, J. Stephen. Music information retrieval (Chapter 7,Page 295-296). In *Annual Review of Information Science and Technology 37*, ed. Blaise Cronin, 295-340. Medford, NJ: Information Today, 2003

[2] Music information retrieval. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2017-12-12]. Dostupné z: https://en.wikipedia.org/wiki/Music_information_retrieval

[3] Music Informational Retrieval. *SlideShare* [online]. Italy: Alberto de Bortoli, 2010 [cit. 2017-12-12]. Dostupné z: https://www.slideshare.net/albertodebortoli/music-information-retrieval

[4] Downie, J. Stephen. Music information retrieval (Chapter 7,Page 297-301). In *Annual Review of Information Science and Technology 37*, ed. Blaise Cronin, 295-340. Medford, NJ: Information Today, 2003.

[5] Music information retrieval. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2017-12-12]. Dostupné z: https://en.wikipedia.org/wiki/Music_information_retrieval

[6] KLAPURI, Anssi. a Manuel. DAVY. *Signal processing methods for music transcription*. New York: Springer, c2006. ISBN 9780387306674.

[7] RAŚ, Zbigniew. a Alicja A. WIECZORKOWSKA. *Advances in music information retrieval*. Berlin: Springer Verlag, c2010. Studies in computational intelligence, v. 274. ISBN 3642116736.

[8] ISO/1EC International Standart IS 11172-3. *Information Technology – Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5Mbit/s-part 3: Audio*"

[9] D. Pan. *A Tutorial on MPEG/Audio Compression.*IEEE Multimedia Journal, summer 1995.

[10] D.A. Reynolds. *Speaker identification and verification using Gaussian mixture speaker models.* Speech Communication, 17;98-108, 1995.

[11] J.T. Foote. *The TreeQ Package.* Available from ftp://svr-ftp.eng.cam.ac.uk/pub/comp.speech/tools/treeq1.3.tar.gz

[12] J.T. Foote. *Content- Based Retrieval of Music and Audio.* Proc. Of SPIE, Multimedia Storage and Archiving Systems 2, pp 138-147, 1997.

[13] S.B. David and P. Mermelstein. *Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences.* IEEE Trans. Acoustics, SPeech, Signal Proc., ASSP-28 (4), 1980.

[14] L.R. Rabiner and B.H. Juang. *Fundamentals of Speech Recognition,* Prentice-Hall, Englewood Cliffs, NJ, 1993.
[15] P. Ciaccia, M.Patella and P.Zezula. *M-tree: An Efficient Access Method for Similarity Search in Metric Spaces.* Proc. Of VLDB, 1997.
[16] GULDAN, Michal. *Automatická analýza zvukovej informácie*. Žilina, 2006. Diplomová práca. ŽILINSKÁ UNIVERZITA V ŽILINE. Vedoucí práce Ing. Roman Jarina, PhD.
[17] *Charm: Sonic Visualiser* [online]. London, England: King's College London, 2009 [cit. 2017-12-12]. Dostupné z: http://www.charm.rhul.ac.uk/analysing/p9_0_1.html
[18] *Vamp Plugins: Vamp Plugin* [online]. London, England: University of London, Queen Mary, 2009 [cit. 2017-12-12]. Dostupné z: http://www.vamp-plugins.org/guide.pdf
[19] CHEN, C. H. *Signal processing handbook*. 1. New York: Dekker, c1988. ISBN 0824779568.
[20] JL Flanagan, Speech Analysis, Synthesis and Perception, Springer-Verlag, New York, 1972
[21] "What's an Order?". *siemens.com*. 11 July 2016. Retrieved 7 April 2018
[22] Spectrograph". *www.sfu.ca*. Retrieved 7 April 2018
[23] "Spectrograms". *ccrma.stanford.edu*. Retrieved 7 April 2018
[24] STFT Spectrograms VI - NI LabVIEW 8.6 Help". *zone.ni.com*. Retrieved 7 April2018.
[25] Judith C. Brown, Calculation of a constant Q spectral transform, *J. Acoust. Soc. Am.*, 89(1):425–434, 1991.
[26] Continuous Wavelet Transform "When the mother wavelet can be interpreted as a windowed sinusoid (such as the Morlet wavelet), the wavelet transform can be interpreted as a constant-Q Fourier transform. Before the theory of wavelets, constant-Q Fourier transforms (such as obtained from a classic third-octave filter bank) were not easy to invert, because the basis signals were not orthogonal."
[27] WOLF, Joe. Note names, MIDI numbers and frequencies. *The University New South Wales*[online]. United Kingdom [cit. 2018-05-27]. Dostupné z: https://newt.phys.unsw.edu.au/jw/notes.html
[28] Hendrik Purwins, Benjamin Blankertz and Klaus Obermayer, A New Method for Tracking Modulations in Tonal Music in Audio Data Format, *International Joint Conference on Neural Network (IJCNN'00).*, 6:270-275, 2000.
[29] Benjamin Blankertz, The Constant Q Transform, 1999.

[30] Amatriain, X., Bonada, J., Loscos, A., and Serra, X. (2002). Spectral processing. In Z¨olzer, U., editor, *DAXF-Digital Audio Effects*, pages 373–438. John Wiley & Sons.

[31] Shepard, Roger N. (1964). "Circularity in judgments of relative pitch". *Journal of the Acoustical Society of America*. **36** (212): 2346–2353.

[32] LEE, Newton. *Digital Da Vinci: computers in music*. New York: Springer, 2014

[33] Serra, X. (1996). Musical sound modeling with sinusoids plus noise. In Poli, G. D., Picialli, A., Pope, S. T., and Roads, C., editors, *Musical Signal Processing*. Swets & Zeitlinger.

[34] Amatriain, X., Bonada, J., Loscos, A., and Serra, X. (2002). Spectral processing. In Z¨olzer, U., editor, *DAXF-Digital Audio Effects*, pages 373–438. John Wiley & Sons.

[35] GOMEZ GUTIERREZ, Emilio. *TONAL DESCRIPTION OF MUSIC AUDIO SIGNALS* [online]. Barcelona, 2006 [cit. 2018-05-27]. Dostupné z: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.76.8192&rep=rep1&type=pdf. Dizertácia. THE DEPARTMENT OF TECHNOLOGY OF THE UNIVERSITAT POMPEU FABRA FOR THE PROGRAM IN COMPUTER SCIENCE AND DIGITAL COMMUNICATION IN PARTIAL FULFILMENT OF THE REQUIREMENTS. Vedoucí práce Dr. Xavier Serra.

[36] Fujishima, T. (1999). Realtime chord recognition of musical sound: a system using common lisp music. In ICMA, editor, *International Computer Music Conference*, pages 464–467, Beijing, China.

[37] Rossing, T. D. (1989). *The sience of sound*. Addison-Wesley, second edition.

[38] Fletcher, N. H. and Rossing, T. D. (1991). *The physics of musical instruments*. Springer-Verlag.

[39] Morse, P. M. (1983). *Vibration and sound*. American Institute of Physics for the Acoustical Society ofAmerica, second (paperback) printing edition.

[40] Schwarz, D. and Rodet, X. (1999). Spectral envelope estimation and representation for sound analysissynthesis.

[41] In *International Computer Music Conference*, pages 351–354, Beijing, China.

[42] MIDI tuning standard. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-05-27]. Dostupné z: https://en.wikipedia.org/wiki/MIDI_tuning_standard

[43] *Oficial MIDI specifications* [online]. [cit. 2018-05-27]. Dostupné z: https://www.midi.org/specifications-old/item/the-midi-1-0-specification

[44] Sebastian Bo ¨ck, Florian Krebs and Ger- hard Widmer. "Accurate Tempo Estimation based on Recurrent Neural Networks and Resonating

Comb Filters", 16th International Society for Music Information Retrieval Conference, 2015

[45] TEMPO AND BEAT ESTIMATION OF MUSICAL SIGNALS Miguel Alonso, Bertrand David , Gael¨ Richard ENST-GET, Departement ´ TSI 46, rue Barrault, Paris 75634 cedex 13, France {malonso,bedavid,grichard}@tsi.enst.fr

[46] Laroche, J. "Efficient Tempo and Beat Tracking in Audio Recordings", J. Audio. Eng. Soc., vol. 51, No. 4, pp. 226–233, April 2003

[47] Todd, N. P. McA., "The Auditory 'Primal Sketch': A Multiscale model of rhythmic grouping", Journal of New Music Research, vol. 23, No. 1, pp. 25 – 70, 1994.

[48] Alonso M., David B. and Richard G., "A Study of Tempo Tracking Algorithms from Polyphonic Music Signals", Proceedings of the 4th. COST 276 Workshop, Bordeaux, France. March 2003.

[49] Spectrogram: Applications. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 27 May 2018 [cit. 2018-05-28]. Dostupné z: https://en.wikipedia.org/wiki/Spectrogram

[50] Chroma features: Apllications. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 9 April 2018 [cit. 2018-05-28]. Dostupné z: https://en.wikipedia.org/wiki/Chroma_feature

[51] *Music Tempo Estimation with k-NN Regression* [online]. 2009, , 7 [cit. 2018-05-28]. Dostupné z: https://www.cs.tut.fi/sgn/arg/klap/eronen-tempo-2009.pdf

[52] KNOPKE, Ian a Chris CANNAM. *Sonic Annotator* [online]. [cit. 2018-05-27]. DOI: Ian Knopke and Chris Cannam. Dostupné z: https://www.vamp-plugins.org/sonic-annotator/runner.pdf

[53] *Soundsoftware* [online]. United Kingdom [cit. 2018-05-27]. Dostupné z: https://code.soundsoftware.ac.uk/projects/sonic-annotator/wik

Zoznam príloh

1. CD
2. Zabalený súbor vo forme .rar s pluginmy pre program Sonic Visualiser (vo forme DLL)
3. Návod na inštaláciu a obsluhu pluginov

Obsah príloh

1.CD

a) Elektronická verzia práce
b) Pluginy pre program Sonic Visualiser (vo forme DLL)
c) Návod na inštaláciu a obsluhu pluginov

2.Zabalený súbor .rar

a) Spektrogram
b) Chromagram
c) Určovanie tempa
d) RMS