VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2019 Tereza Búliková

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

SYNCHRONIZACE NOTOVÉHO ZÁPISU S HUDEBNÍ NAHRÁVKOU
AUDIO-TO-SCORE ALIGNMENT TOOL

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Tereza Búliková**
AUTHOR

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

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Audio inženýrství**
Ústav telekomunikací

Studentka: Tereza Búliková ***ID:*** 186039
Ročník: 3 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

Synchronizace notového zápisu s hudební nahrávkou

POKYNY PRO VYPRACOVÁNÍ:

Vytvořte systém, který bude schopen synchronizovat jak jednotlivé druhy interpretací (ve formátech WAV nebo MIDI) vzájemně, tak přímo skladbu s notovým zápisem neboli Audio-to-score alignment (např. MusicXML, MIDI, CMF, …). Systém bude následně otestován na skladbách jak klasické, tak populární hudby.

DOPORUČENÁ LITERATURA:

[1] MONTECCHIO, N.; CONT, A. A unified approach to real time audio-to-score and audio-to-audio alignment using sequential Montecarlo inference techniques. Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2011: s. 197-200.

[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: 1.2.2019 ***Termín odevzdání:*** 27.5.2019

Vedoucí práce: Ing. Tomáš Kiska
Konzultant:

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

ABSTRAKT

Práce se zabývá získáním spektrálních a chroma vlastností z audio nahrávek, které jsou použity pro synchronizační algoritmus Borcení časové osy. Tento algoritmus je poté využit pro tvorbu synchronizačních programů Audio-to-audio a Audio-to-score.

KLÍČOVÁ SLOVA

Hudební synchronizace, spektrogram, chromagram, Borcení časové osy, Skryté Markovovy modely, Audio-to-audio synchronizace, Audio-to-score synchronizace

ABSTRACT

This thesis deals with obtaining spectra and chroma features from audio records. Features are used in synchronization algorithm Dynamic Time Warping. This algorithm is used to create synchronization programs Audio-to-audio and Audio-to-score alignment.

KEYWORDS

Music alignment, spectrogram, chromagram, Dynamic Time Warping, Hidden Markov models, Audio-to-audio alignment, Audio-to-score alignment

BÚLIKOVÁ, T. *Synchronizace notového zápisu s hudební nahrávkou*. Brno, 2019, 54 s. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: Ing. Tomáš Kiska

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Synchronizace notového zápisu s hudební nahrávkou“ jsem vypracoval(a) samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor(ka) uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil(a) autorská práva třetích osob, zejména jsem nezasáhl(a) nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom(a) následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora(-ky)

PODĚKOVÁNÍ

Ráda bych poděkovala vedoucímu diplomové práce panu Ing. Tomáši Kiskovi, za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno . . . .

podpis autora(-ky)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

PODĚKOVÁNÍ

Výzkum popsaný v této bakalářské práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

Brno ....

podpis autora(-ky)

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM VÝPISŮ

ÚVOD

Tato práce se bude věnovat vybraným audio signálům, získáním a zpracováním jejich vlastností. Bude uveden přehled a popis nejpoužívanějších algoritmů v oblasti audio signálů, z nichž algoritmus *Borcení časové osy* využijeme při programování zadaných synchronizačních programů.

První program se bude zabývat synchronizací nahrávky k nahrávce, kdy jedna nahrávka bude považována za referenční a druhá za testovanou. Program bude mít za úkol nahrávky sjednotit a seřadit s prvky druhé nahrávky tak, aby odpovídala prvkům nahrávky první (např. začátek sloky nebo refrénu). Uživatel bude moci sledovat průběh jejich synchronizace na dvou posuvnících a bude obeznámen s procentuální podobností vybraných skladeb.

Druhý program se bude zabývat synchronizací audio nahrávky k notovému zápisu. Zatímco se nahrávka bude přehrávat, program bude obsahovat ukazatel, který bude poukazovat na aktuální takt a dobu notového zápisu.

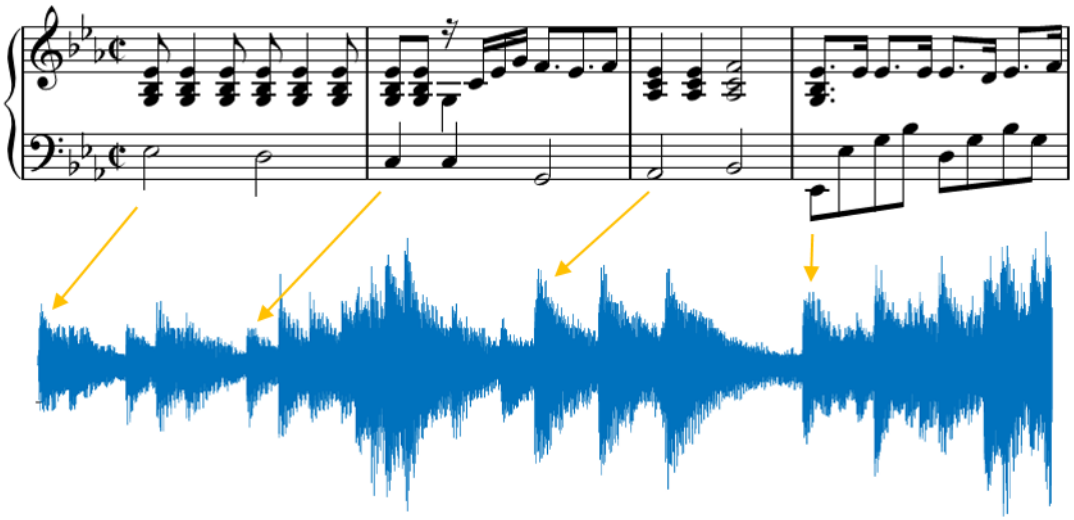
Budou uvedena bloková schémata obou programů a popis jednotlivých bloků a funkcí programů. Programy budou následně testovány na skladbách klasické i populární hudby. Pro jednotlivé testy bude uvedena úspěšnost synchronizace a případné důvody nízké úspěšnosti.

1 HUDEBNÍ SYNCHRONIZACE

Hudba může být zaznamenána několika různými způsoby. Může být zapsána do not a následně předváděna či může být zhotovena jako nahrávka. Autor se snaží vložit do svého díla určitou myšlenku vyjádřenou například v daném seskupení hudebních nástrojů, ve změnách hlasitosti nebo tempa. Záleží však na interpretovi, jestli bude autorovu myšlenku následovat, nebo zda bude myšlenku měnit a experimentovat.

Proto zpravidla pro jedno hudební dílo neexistuje pouze jedno zpracování. Ať se jedná o díla klasické hudby, která bývají v hojném počtu hrána po celém světě od žáků základních uměleckých škol až po prestižní filharmonie nebo skladby populární hudby, každé zpracování se jistým způsobem liší od jiného.

Hudební synchronizace se zabývá sladěním těchto odlišností, porovnáváním průběhů dvou i více nahrávek, nahrávky a jejího notového zápisu, porovnáním akordů, atd. Obrázek 1.1 graficky naznačuje hudební synchronizaci mezi nahrávkou a notovým zápisem. Každá nota odpovídá určitému časovému průběhu v nahrávce. Abychom dokázali hudební nahrávky nebo notový zápis provázat a seřadit, potřebujeme z formátů hudebního zápisu získat specifické informace.



Obr. 1.1: Notový zápis skladby a jeho oscilogram.

1.1 Získávání audio informace

Existuje několik druhů informací o hudební nahrávce, které se následně využívají pro synchronizační algoritmy. Výběr zvukové informace záleží právě na daném synchronizačním algoritmu. Tato práce se bude zabývat informacemi o spektrálních vlastnostech a jejich převedením na *chroma* vlastnosti (*chroma* – ze starořeckého *khrôma*, ve významu barva, barevnost, jasnost).

Lidský mozek vnímá výšku tónu periodicky a to tak, že dva tóny lišící se o oktávu, mozek označí jako *barevně* podobné [1]. Na obrázku 1.2 je spektrogram dvou oktáv. Liší se pouze fundamentálním tónem, vyšší harmonické jsou však shodné, to proto, že vzdálenost dvou oktáv je v poměru 2:1. Tón tedy můžeme rozdělit na dvě části – na tónovou výšku a jeho *chroma* vlastnost [2].

1.1.1 Znázornění výšek tónů, spektrogram

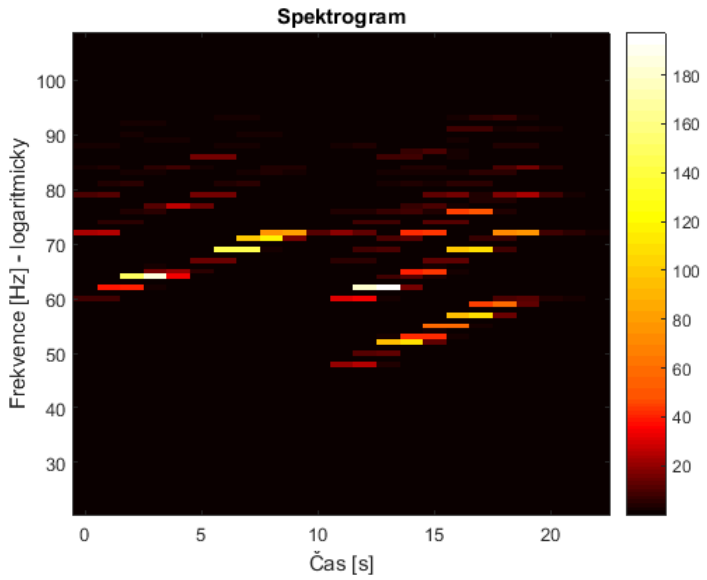
Výška tónu je jedna ze základních čtyř vlastností tónu (mezi další patří délka, barva, hlasitost), díky kterým jsme schopni posuzovat rozdílnost tónů. Výška tónu je určena frekvencí. Čím je frekvence vyšší, tím i lidské ucho slyší tón jako *vyšší*. Do frekvenčního pásma 20 Hz až 20 kHz, které odpovídá frekvencím slyšitelným pro lidské ucho, se vejde jedenáct oktáv.

Pro zobrazení výšek audio nahrávky používáme spektrogram. Spektrogram zobrazuje spektrum jednotlivých frekvencí měnících se v čase a jejich intenzitu. Osa *ypsilonových* souřadnic může mít dvě reprezentace, lineární a logaritmickou. Jestliže zobrazujeme frekvenci na ose y v lineárním měřítku, frekvenční pásma se na časové ose rozdělí na stejně velké intervaly. Pro logaritmické měřítko se interval frekvencí exponenciálně zvyšuje. Převedení signálu na spektrogram je ztrátovým procesem.[1]

Spektrogram z původní audio informace lze získat dvěma způsoby. Prvním způsobem je analogové řešení. Frekvence se rozdělí na zvolený počet intervalů pomocí pásmových filtrů, kde výstup filtru řídí převodník. Jejich obálky tvoří vodorovné pruhy spektrogramu. Pro jednotlivé intervaly můžeme volit různé vzorkovací frekvence (např. 20 050 Hz, 44 100 Hz, 96 000 Hz). Popis řešení takového spektrogramu najdeme v lit. [3].

Digitálně se spektrogram vytváří pomocí STFT, neboli *krátkodobé Fourierovy transformace*, která rozdělí signál na stejné úseky předem daným symetrickým oknem a v každém úseku se vypočítá *diskrétní Fourierovu transformaci* signálu. STFT může být aplikována v reálném čase. Metodu řešení a podrobné vzorce najdeme zde [4].

[1] Ztrátový proces je takový proces, při kterém přicházíme o informace a vlastnosti, které jsou zatěžující pro systém, ale nemají vliv na funkčnost a správnost dalších procesů.



Obr. 1.2: Spektrogram v logaritmickém měřítku pro diatonickou stupnici v jedno-čárkové a malé oktávě.

Obrázek 1.2 zobrazuje spektrogram nahrávky diatonické stupnice[2] nejprve zahrané v jedno-čárkové oktávě, poté v malé oktávě. Na ose x vidíme délku jednotlivých vzorků, na ose y hodnotu frekvence a sytost barvy zobrazuje intenzitu vzorku.

V této práci je použit spektrogram vytvořený analogovým způsobem v logaritmickém zobrazení. Pokud budeme uvažovat rozložení klaviatury, začneme na tónu $A0$ a rozsah zakončí tón $C8$, dostaneme dohromady 88 tónů, které odpovídají zápisu MIDI notace. Hodnota nejnižšího tónu je 21 a nejvyššího 108. Signál se tedy snažíme rozdělit na 88 frekvenčních intervalů a střední frekvence pak odpovídá frekvenci tónu v intervalu.

Dále můžeme definovat tzv. *výškové třídy*, kde jedna třída zahrnuje stejnou skupinu oktáv $\{ \dots C0, C1, C2 \dots \}$ [5]. *Výšková třída* je sled výšek tónů, které mají stejnou *chroma* vlastnost.

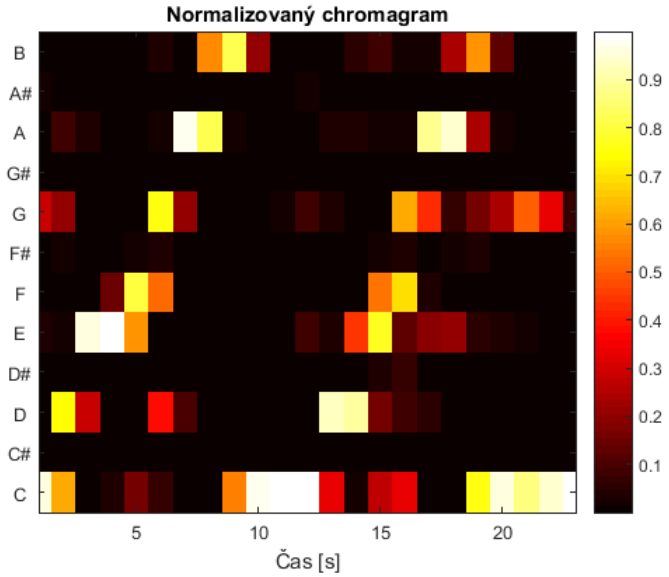
1.1.2 Chroma vlastnosti, chromagram

Dvojnásobek základní frekvence tónu nese stejné jméno jako tón základní, lišící se pouze o přívlastek polohy tónu (subkontra $A0$ (27,5 Hz), malé $A3$ (220 Hz), jedno-čárkové $A4$ (440 Hz), atd.). Oktávu pak lze rozdělit na 12 půltónů $\{ C, C\sharp, D \dots H \}$, kterým odpovídá *chroma* vektor $\{ x(1) = C, x(2) = C\sharp, \dots x(12) = H \}$ [2].

[2]Diatonické stupnice jsou tvořeny ze sedmi tónů, vzdálených od sebe o celé tóny nebo půltóny.

Chroma vlastnosti sjednocují v časovém oknu spektrální informace do jedné proměnné. Pokud budeme časové okno posouvat přes celý signál, dostaneme posloupnost několika *chroma* vlastností a každá zobrazuje rozložení výšek v časovém okně ve 12 rozměrovém *chroma* vektoru. *Chroma* vlastnosti zachycují melodickou a harmonickou podstatu, která se nemění se změnou barvy tónu. Chromagram je zobrazení energie *výškových tříd* v *chroma* vektoru [5].

Chromagram můžeme získat rozdělením signálů na frekvenční intervaly nebo *krátkodobou Fourierovou transformací*, výhodné je použít předchozí výpočty pro spektrogram. Pro analogové řešení se v každém intervalu spočítá průměrný výkon konvolucí vybraného intervalu s obdélníkovým oknem – pro každý tón z *výškové třídy* je spočítán výkon v jednotlivých pozicích *chroma* vektoru. Energie je dána součtem výkonů propočítaných vektorů. Podrobně popsáno zde [6]. Pro *Fourierovu transforamci* je výkon spočítán autokorelací signálu.



Obr. 1.3: Normalizovaný chromagram pro diatonickou stupnici v jedno-čárkové a malé oktávě.

Chromagram na obrázku 1.3 je normalizovaný chromagram. Normalizaci, někdy označovanou jako logaritmickou kompresi, využíváme pro dynamické změny v audio signálu a tiché pasáže s velmi nízkou energií, jako jsou pomlky [5]. Tišší, ale významnější pasáže, mohou být dynamicky překryté hlasitějšími nebo nežádoucím hlukem. Normalizace se snaží vyvážit tyto rozdíly nejméně na stejné hodnoty, ideálně tiché pasáže zesílit.

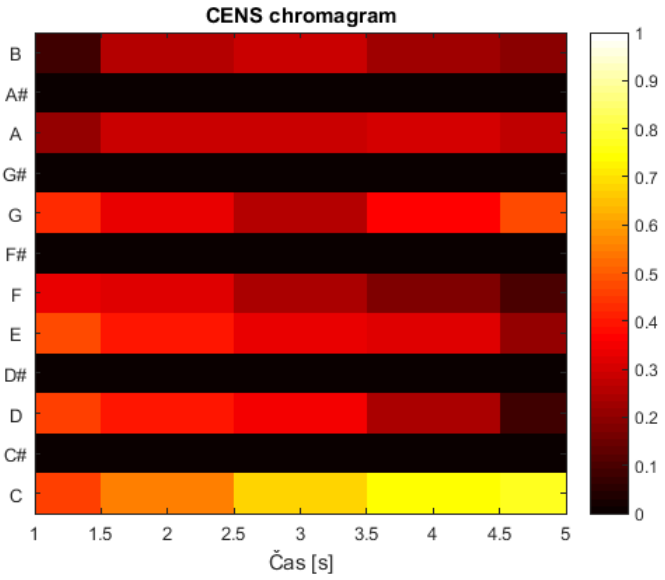
Míra komprese není stanovena a musí se volit s ohledem na audio signál. Hodnota komprese by měla být kompromisem mezi maximálním zesílením slabých sig-

nálů a minimálním zesílením signálů hlasitých nebo nežádoucích. Normalizace je nahrazení vektoru normalizovaným vektorem propočítaným v *eukleidovské metrice*. Převedení na normalizovaný vektor řeší [5].

Důležitým parametrem pro správné zobrazení chromagramu je ladění audio nahrávky. Pokud nahrávka není laděna podle komorního *A* (440 Hz), které odpovídá v MIDI notaci hodnotě 69, je nutné vypočítat odchylku ladění, podle které se přepočítají střední frekvence 88 frekvenčních intervalů. Při neuvážení odlišného ladění nahrávky se energie může zobrazit náhodně a nezávisle na *chroma* vektorech.

Chromagram můžeme dále upravovat pro získání dalších *chroma* vlastností, které se využívají pro hudební synchronizaci. *CENS* (*Chroma Energy Normalized Statistics*) vlastnosti, obrázek 1.4, jsou posloupnosti vlastností, které zobrazují krátkodobou statistiku rozložené energie v *chroma* vektoru [2]. Pro počítání *CENS* vlastností nejprve normalizujeme *chroma* vektory (opět můžeme využít předchozích výpočtů), aplikujeme kvantizaci na zvolených úrovních (mírách komprese), hodnoty vyhladíme okénkovou funkcí o zvolené délce a snížíme vzorkovací frekvenci o zvolený násobek [2].

Problémy mohou nastat pro různé hudební efekty a špatně vyladěné nahrávky. V audio signálu se mohou vyskytovat různé výkyvy amplitudy signálu (trylky, tremola, atp.), které jsou obtížné pro výpočet, proto se někdy vyhlazování provede jako první.



Obr. 1.4: *CENS* chromagram pro diatonickou stupnici v jedno-čárkové a malé oktávě.

Hodnoty vlastností *CENS* chromagramů se dají vhodně použít pro výpočet po-

dobnostních funkcí hudební synchronizace popsaných v následující kapitole.

1.2 Algoritmy pro hudební synchronizaci

Reálné signály se často snažíme převádět do různých signálových modelů. Tyto modely jsou výhodné především proto, že poskytují základní informace pro teoretický rozbor signálově zpracovávaného systému, který může být využit pro zpracování jiných systémů, například simulace dolní propusti.[3] Dále nesou informaci o zdroji signálu, který nemusí být při zpracování k dispozici, ale můžeme ho zpětně simulovat, ale především umožňují vytvářet pravděpodobnostní, rozpoznávací a identifikační systémy [7].

Modely systémů můžeme rozdělit na deterministické a stochastické [7]. Deterministický model je předvídatelný a funguje tak, že ze stejných vstupních podmínek vždy dostaneme stejné výstupní podmínky. Deterministickým modelem je například synchronizační algoritmus Borcení časové osy neboli *Dynamic Time Warping*.

Další model je stochastický, jehož prvky nebo vztahy jsou náhodné. Tento model může být nazván také statistický. Signál je považován za náhodný proces různých parametrů, které ale nadále mohou být zpracovávány deterministicky. Často využívaný stochastický model se nazývá Skrytý Markovův Model, neboli (*Hidden Markov Models*). Tento model může být využit samostatně nebo je implementován do dalších modelů, které se přizpůsobují konkrétnímu využití.

Hlavní myšlenkou srovnávání je to, že můžeme vyčíslit, jak moc jsou, či nejsou dvě posloupnosti se sledovanými znaky podobné, přestože délka každé posloupnosti se liší. Výběr metody úzce souvisí s její aplikací a s dostupností dat potřebných k aplikaci [8].

1.2.1 Metoda borcení časové osy

Synchronizační algoritmus *Dynamic Time Warping*, zkráceně DTW, se používá pro časové seřazení a synchronizaci dvou posloupností o nestejných délkách [9]. Původně byla vytvořena pro porovnávání řečových signálů pro automatické rozpoznávání řeči. Pro případ hudební synchronizace se jedná o dvě posloupnosti obsahující například *chroma* vlastnosti nahrávek.

Posloupnost o počátečních hodnotách X(původní) a Y(původní) se změní na posloupnost hodnot X(smrštěná/deformovaná) a Y(původní). Pro operaci, která smršťuje nebo roztahuje části audio signálů, se používá název *time warping*. Slovo *dynamic* se přidává v případě, že posloupnosti se sledovanými vlastnostmi reprezentují audio signál [8].

[3] Frekvenčí filtr, který se požívá pro filtraci audio signálů.

Nechť je jedna posloupnost definována jako $X = (x_1, x_2 ... x_n)$, kde N je délka této posloupnosti, obdobně pro druhou posloupnost platí $Y = (y_1, y_2 ... y_m)$, M je délkou druhé posloupnosti [5]. Každá tato posloupnost reprezentuje jednu podobu nahrávky. Každá posloupnost může být jinak dlouhá, přestože si hudebně odpovídají. DTW hledá kompromis mezi tempem nahrávek a uspořádává prvky posloupnosti vynecháváním některých vzorků nebo naopak podržením jejich hodnot.

Abychom posloupnosti mohli seřadit, potřebujeme sjednotit jednotlivé prvky první posloupnosti k prvkům druhé. To se provádí pomocí rozdílů vzdáleností mezi každými prvky obou posloupností. Odtud dostaneme tzv. *vzdálenostní* matici těchto vzdáleností, která se skládá z $M \times N$ prvků – $\boldsymbol{D}(n, m)$. Její indexování začíná na počátku signálů ($\boldsymbol{D}(0, 0)$) a končí na konci obou posloupností s indexy $\boldsymbol{D}(N-1, M-1)$. Jako vedlejší produkt synchronizačního algoritmu můžeme získat hodnotu největšího rozdílu mezi prvky.

Hodnoty vzdáleností uspořádané za sebou vytvoří tzv. *warping path* – vzdálenostní cestu (obr. 1.5), posloupnost $P = (p_1 ... p_L)$, kde L je její délka. *Vzdálenostní* cesta má tyto 3 předpoklady [9]:
1. Její hranice musí být omezené, musí začínat prvním prvkem obou posloupností X, Y a končit na konci těchto posloupností,

$$p_1 = (1, 1), \tag{1.1}$$

$$p_L = (N, M). \tag{1.2}$$

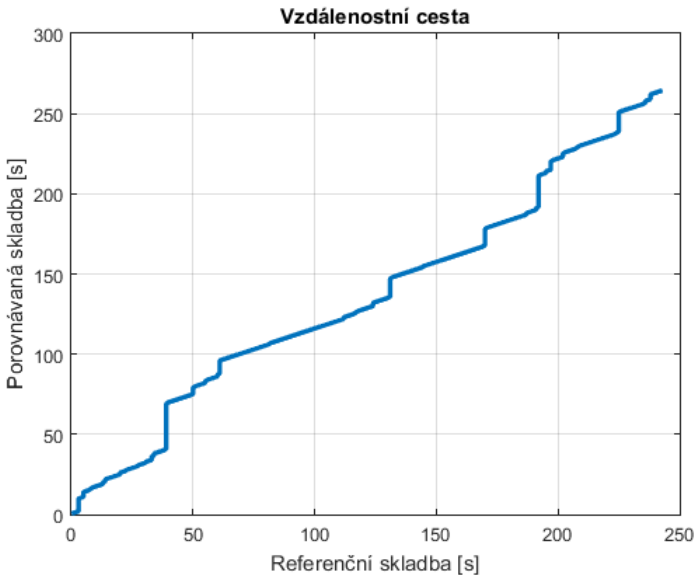
2. Musí být kauzální, to znamená, že posloupnost musí pokračovat stále kupředu a nesmí se vracet v čase, neboli nesmí reagovat na následek dříve než na příčinu,

$$n_1 \leq n_2 \leq ... \leq n_L \wedge m_1 \leq m_2 \leq ... \leq m_L. \tag{1.3}$$

3. Posloupnost musí být spojitá, nesmí vynechat žádný prvek předem uvedených posloupností. To pro rozdíl dalšího a aktuálního prvku zajistí velikost kroku

$$p_{l+1} - p_l \in \{(1, 0), (0, 1), (1, 1)\}, l \in [1 : L - 1]. \tag{1.4}$$

Vzdálenostní cesta nemusí existovat pouze jedna. Jednotlivé buňky *vzdálenostní* cesty zaručují, že každý prvek vektoru X má vztah k nejméně jednomu prvku vektoru Y [5]. *Vzdálenostní* cestu získáme propočítáním všech možných cest ve *vzdálenostní* matici a za optimální je zvolena ta s nejmenší celkovou hodnotou vzdálenosti.



Obr. 1.5: *Vzdálenostní* cesta.

Výsledkem funkce DTW je také tzv. *cenová* matice [9]. Má stejné rozměry jako matice *vzdálenostní*, ale každý její prvek obsahuje kumulovanou hodnotu optimální *vzdáleností* cesty ke specifickému prvku matice. *Cenovou* matici počítáme podle vzorce [9]:

$$\boldsymbol{C}(n,m) = \boldsymbol{D}(n,m) + min \begin{cases} \boldsymbol{C}(n-1,m-1) \\ \boldsymbol{C}(n-1,m) \\ \boldsymbol{C}(n,m-1) \end{cases} . \tag{1.5}$$

Každý prvek *cenové* matice obsahuje minimální hodnotu k dosažení tohoto prvku. Při výpočtu této matice musí být ukládány indexy každého prvku předešlých buněk, které byly vybrány jako minimální hodnota. *Vzdálenostní* cesta se pak odvozuje z aktuálního prvku k začátku matice [9].

DTW algoritmus tak můžeme shrnout jako soubor podmínek, kterými jsou: *cenová* i *vzdálenostní* matice musí začínat od prvního prvku, pro indexy *cenové* matice menší než nula je výsledkem nekonečno (nahrávky nejsou podobné), je rekurzivní – využívá sama sebe pro definování matice, délky obou signálů jsou konečné. Pokud se vydáme zpátky po *vzdálenostní* cestě odečítáním 1, musíme dojít k počáteční hodnotě [9].

Jelikož je tento algoritmus využíván v několika oborech, vznikly předdefinované funkce, které uživatel může využívat, aniž by musel znát celý postup algoritmu. Každá funkce má svůj vlastní postup pro řešení borcení časové osy a každá se liší

v použití. Tato práce bude využívat DTW podle Itakury, ale dalším příkladem je např. DTW podle Sakoe-Chiba nebo Algoritmus Smith-Waterman [8].

DTW podle Itakury

Tento algoritmus je schopen počítat s libovolně dlouhými posloupnostmi, u kterých nemusíme předpokládat většinovou shodu vlastností. Začínáme v uzlu $(1,1)$, tedy definujeme cenovou matici na tyto indexy a položíme ji rovno tzv. *dissimilarity* funkci. *Dissimilarity* funkce tady představuje funkci *eukleidovské vzdálenosti* dvou vzorků v posloupnosti neboli říká, jak moc jsou vzorky rozdílné [8]:

$$\boldsymbol{C}(1,1) = d(n(1), m(1)), \text{ kde} \tag{1.6}$$

$$d(a,b) = \sqrt{(\sum_{l=1}^{K}(a_l - b_l)^2}, \tag{1.7}$$

a,b zde zastupují obecné posloupnosti, K je délka vektorů.

Prvním uzlem je tedy uzel s indexy $(1,1)$, tudíž jeho předcházející hodnota odpovídá fiktivnímu uzlu s indexy $(0,0)$. Do *vzdálenostní* matice $\boldsymbol{D}$ ukládáme předcházející hodnotu každého uzlu. *Omezení lokální cesty* neboli *cenovou* matici vyjádříme jako:

$\boldsymbol{C}(n,m) = min\{\boldsymbol{C}(n-1,m-1), \boldsymbol{C}(n-2,m-1), \boldsymbol{C}(n,m-1)\} + d(x(n), y(m))$,
pokud $\boldsymbol{D}(n,m-1) \neq (n,m-2)$
a
$\boldsymbol{C}(n,m) = min\{\boldsymbol{C}(n-1,m-1), \boldsymbol{C}(n-2,m-1)\} + d(x(n), y(m))$,
pokud $\boldsymbol{D}(n,m-1) = (n,m-2)$ [8].

Můžeme si všimnout, že vertikální změny nejsou zcela dovoleny. Dále je možno vynechat prvek vektoru posloupnosti, která se nachází na vertikální ose (obr. 1.5), díky tomu, že přecházející uzel můžeme indexovat $(n-2, m-1)$. Posledním postřehem je fakt, že nejsou povoleny dva posuvy v ose horizontální kvůli omezení $\boldsymbol{D}(n,m-1) \neq (n,m-2)$. Důsledek tohoto omezení je, že konečný prvek referenčního signálu je v poloviční a dvojnásobné mezi porovnávaného signálu – $\frac{1}{2}M \leq N \leq 2M$ [8].

Algoritmus se opakuje pro každý uzel až do uzlu posledního (N,M), kde můžeme najít *cenu* obou posloupností. Zpáteční cestou získáváme optimální cestu synchronizace s nejvhodnějším předešlým krokem.

Itakurovo borcení časové osy se nenachází v Matlabu jako vestavěná funkce, uživatel ji však může získat s přihlédnutím na všechny podmínky autorských práv.

Do Matlabu může být přidána jako součástí nějaké knihovny nebo jako samostatná funkce. Správné volání funkce je:

[DTWcost, BestPath] = dynamicTimeWarpingItakura(B,A,tonorm).

Vstupními hodnotami jsou vektory A a B, kde A je referenční posloupnost a B porovnávaná. *Tonorm* je automaticky nastaven na hodnotu 1 a znamená to, že *cena* bude normalizovaná neboli bude podělená délkou optimální cesty. Výstupem jsou dvě veličiny, *DTWcost* ukazuje hodnotu *ceny* (její hodnota je nekonečno, pokud signály nelze synchronizovat) a *BestPath* je vektor uzlů při zpětném vracení se od konce vektorů do počátku optimální cestou [*převzato z funkce* HELP *programu Matlab*].

1.2.2 Skryté Markovovy modely

Markovův model popisuje matematické řešení pro vytvoření modelu, jehož výsledky jsou zčásti náhodné a zčásti kontrolované. Skrytý Markovův model vytváří systém Markovova procesu, jehož stavy jsou nepozorovatelné. Pomocí skrytého Markovova modelu řešíme 3 problémy [10]:

1. Je dán model a posloupnost pozorování. Určujeme pravděpodobnost pozorované posloupnosti vzhledem k modelu.
2. Je dán model a posloupnost pozorování. Nalézáme optimální stav posloupnosti pro základní Markovův proces. Chceme odhalit skryté části Skrytého Markovova modelu.
3. Je dána posloupnost pozorování a dimenze. Nalézáme model, který je maximem pravděpodobnosti posloupnosti.

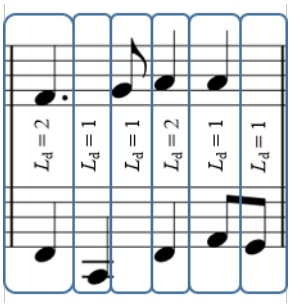
Z těchto 3 problémů, je nutné vyřešit problém č. 2. Řešení problému závisí na nalezení nejpravděpodobnější posloupnosti stavů. Cílem je najít největší pravděpodobnost stavu d_t a v $t = t$ a všechny takovéto stavy spojit.

Algoritmus *Hidden semi-Markov models*, který zde bude popsán, je převzán z literatury [11] a skládá se ze tří modelů a dohromady vytváří systém synchronizace audio nahrávky a notového zápisu. Pravděpodobnost každého stavu HSMM je určena *Latent harmonic allocation* (LHA), což je Bayesovský model[4] harmonické *mixtury*. HSMM odpovídá jednomu instrumentu = hlasu v partituře a pravděpodobnost trvání stavu zaručuje tempový model *Linear dynamics system* (LDS).

[4]Bayesovský model je proces odvozování závěrů o parametrech pomocí pozorovaného vzorku dat, při kterém se využívá Bayesova vzorce. Cílem inference je odhad parametrů, predikce dat a porovnávání modelů [12].

Úkolem algoritmu je odhadnout tempo, časové výkyvy, barvu, dynamiku, výšku tónu, posloupnost stavů a dozvuk. Odhad je rozdělen do dvou etap. Hlasy v partituře, které budou považovány za vedoucí složku, se nazývají *part* a očekáváme u nich trochu jiný průběh než v ostatních hlasech.

Prvním aspektem pro správnou synchronizaci je použít tzv. *dereverberaci* na audio nahrávku, neboli zbavení se dozvuku, vytvořeného akustikou místnosti. Notový zápis je reprezentován jednotlivými notami a začátky a konci jejich trvání. Takové rozdělení partitury na stavy je na obr. 1.6. Aby mohl být vytvořen model posloupnosti stavů, nejprve je vytvořena hladká tempová křivka celé skladby založená na LDS a následně je pro každý *part* určena délka trvání pomocí HSMM, ovládaný taktéž modelem LDS.



Obr. 1.6: Model notového zápisu převedený do posloupnosti stavů.

Tempová křivka T_d je logaritmus hodnoty audio rámců v *tatum* jednom stavu d, kde *tatum* je největší společný dělitel všech not, vyskytujících se v partituře. Celočíselná délka v *tatumech* stavu d je veličina L_d znázorněna na obr. 1.6. Ukázka příkladu: pokud je v notovém zápisu nejkratší nota osminová a aktuální stav obsahuje notu čtvrťovou, bude délka L_d rovna 2 (ve čtvrťové notě jsou obsaženy 2 noty osminové). Přesné vyjádření rovnic pro vypočítání tempové křivky je uveden zde [11].

Notový zápis je tedy posloupnost jednotlivých stavů. Model může být kompletně popsán pomocí 4 proměnných: posloupností stavů, počátečním stavem hustoty pravděpodobnostní funkce (PDF), změnou stavu PDF a délkou stavu PDF. Stavové posloupnosti musí být shodné s celkovou tempovou křivkou a s jednotlivými přechody mezi stavy.

Po získání posloupnosti stavů se vytváří pro všechny *party* CQT spektrogram (spektrogram nahrávky po odstranění dozvuku založený na neparametrickém Bayesovského modelu a nezáporné konvoluci) vybráním pozice v notovém zápisu, příslušného *partu*, kombinace výšky tónu a instrumentu, indexu harmonie a vyzářené energetické kvantum pro každou hodnotu prvku audio signálu.

Na HMM algoritmu je postaveno několik dalších modelů.

Viterbi algoritmus

Algoritmus se zabývá hledáním optimální posloupnosti stavů skrytých Markovových modelů, přičemž výsledkem je posloupnost pozorovaných stavů. Pojem *optimální* je v tomto případě definován na základě pravděpodobnosti.[8]

Baum-Welch algoritmus

Namísto hledání posloupnosti se zde počítá suma posloupností všech stavů, které jsou schopny generovat danou posloupnost pozorování.[8]

HMM training

Cílem algoritmu je *naučit se* parametry Skrytého Markovova modelu. Může se tak dít za dvou podmínek a to za tzv. *učením pod dohledem*, kde využíváme znalost statistických vlastností prvků a dopočítáváme pravděpodobnosti stavů nebo za *učením bez dohledu*, kde dohledáváme vlastnosti pomocí nových prvků.[8]

1.3 Aplikace hudební synchronizace

Aplikace hudební synchronizace můžeme dělit podle vstupních informací, které dále zpracováváme. Může se jednat o audio nahrávku, notový zápis nebo soubor MIDI informací. Podle tohoto vstupního argumentu určujeme, který synchronizační algoritmus bude nejvhodnější použít.

Úkolem aplikace zůstává požadované sjednocení těchto vstupních – testovaných parametrů a zvolených referenčních parametrů. Byly proto vytvořeny dva základní modely, které se dají různě modifikovat a pokryjí široké pole synchronizační problematiky.

1.3.1 Audio-to-audio

Sjednocení nahrávky s nahrávkou je proces získávání odpovídajících si bodů v čase mezi dvěma audio signály podobného nebo stejného obsahu. Toto sjednocení požaduje algoritmus, který je schopný sledovat příbuzné body specifických vlastností v časovém měřítku. Využití této synchronizace je užitečné v několika ohledech.

Algoritmus umožňuje rychlé vyhledávání částí nahrávky, synchronizaci studiové nahrávky s originální nebo upravenou verzí, porovnání rozdílných temp dvou nahrávek nebo synchronizaci při post-produkci audio signálu, jako zařazení jednotlivých částí do uceleného výsledku.

Dále může být tento systém využit pro umělou inteligenci, která bude schopna rozpoznat audio informace nebo pro myšlenku virtuálních učitelů hudebních nástrojů a virtuálních orchestrů.

1.3.2 Audio-to-score

Audio-to-score synchronizace neboli přiřazení nahrávky k jejímu notovému zápisu může být prováděna ve dvou časových rovinách. Pro reálný čas mluvíme o *sledování notového zápisu* a pro nereálný tzv. *offline* systém, se uvádí synchronizace *náhrávka-k-notám*.

Využití tohoto sjednocení se opět týká několika oborů. Systém je schopen propojovat notaci a hudební provedení a umožňuje muzikologům vytvářet notový zápis během poslechu, porovnává dvě různá provedení, používá sjednocení jako nalezení a přiřazení nejlépe vyhovujícího notového zápisu k zadané nahrávce nebo naopak, nalézá automatický doprovod ke skladbě, může vytvářet notový zápis včetně různých informací nebo vytváří učební a tréninkové programy pro hudebníky.

1.3.3 Další využití

Synchronizační algoritmy se dají využít i pro jiné než základní systémy synchronizace. Zvláštním případem je například sjednocení textu písně s probíhající nahrávkou. Jedna z možných realizací je skrze rozpoznávání akordů. Tento případ se dá v praxi využít pro vytváření karaoke programů a videí.

Další oblastí použití je sjednocení audio nahrávky s videoobrazem, hojně využívané ve filmovém průmyslu. DTW i Markovovy modely se často využívají v medicínských oborech.

2 PROGRAMY PRO HUDEBNÍ SYNCHRONIZACI

Programy pro hudební synchronizaci byly vytvořené v uživatelském rozhraní programu Matlab R2015a a následně v něm testovány. Při vytváření programu byly použity knihovny Signal Processing Toolbox [13], Chroma toolbox [14], Mirtoolbox [15], MIDI Toolbox [16] a Audio Analysis library [18]. Dále byl pro převedení formátů a pro testování použit freewarový program MuseScore 3 [19].

Přesto, že je v teoretické části práce popsáno více algoritmů, kterými se dá provádět synchronizace, vzhledem ke zjednodušení bylo zvoleno, že oba programy budou založeny na DTW algoritmu. Toto rozhodnutí mohlo být učiněno zejména proto, že program MuseScore 3 umí převádět všechny notové formáty na formát wav, který je vstupním parametrem právě pro DTW.

2.1 Audio-to-audio synchronizace

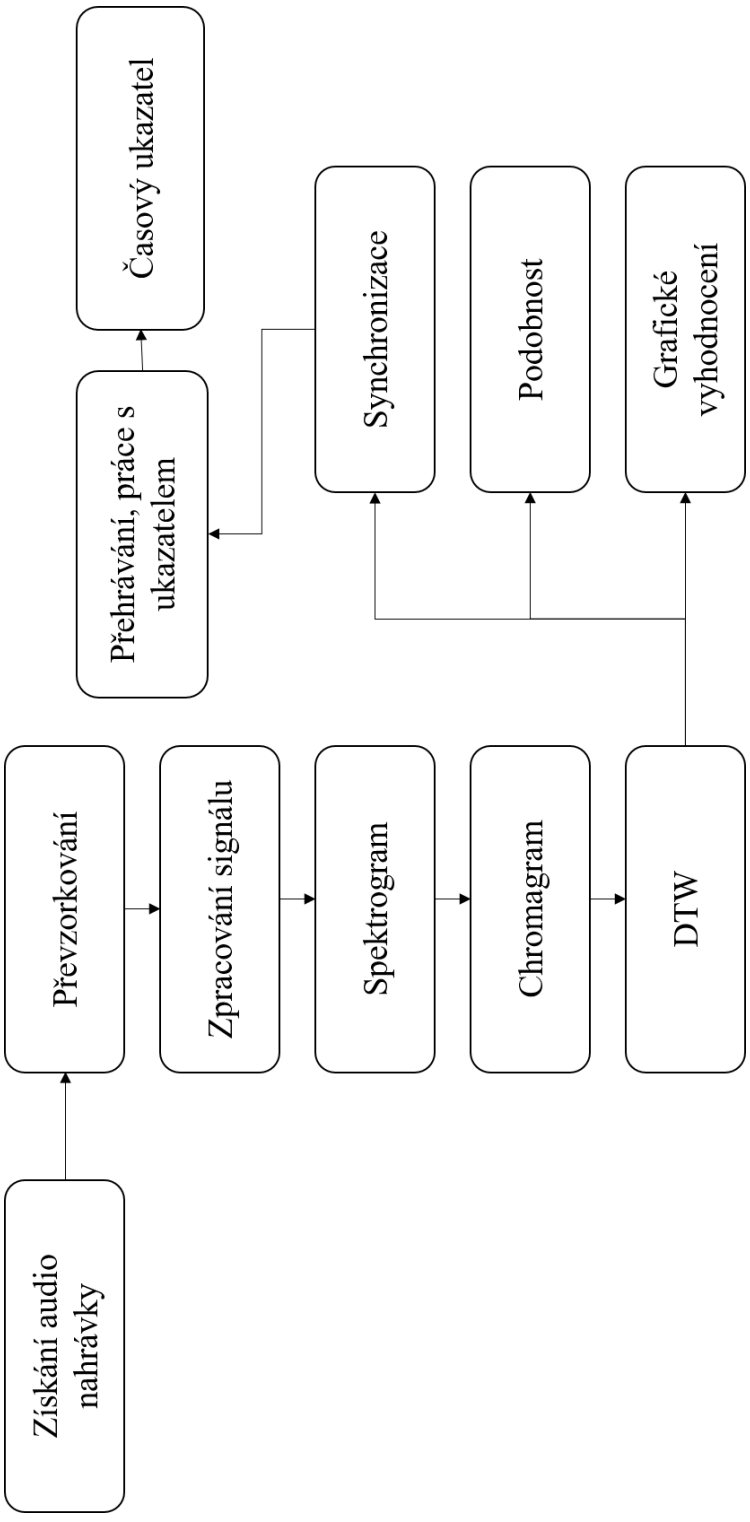
Úkolem práce bylo, aby vytvořený program dokázal sjednotit dvě podobné nahrávky, vypočítal jejich podobnost a ukázal graf jejich podobnosti. Nahrávky měly být volené uživatelem. Po zvolení nahrávek se provede synchronizační algoritmus a dodá uživateli graf a hodnotu podobnosti. V uživatelském rozhraní se mají nacházet dva ukazatele, které se budou pohybovat závisle na sobě podle podobnostní funkce a znázorňovat odlišnosti v nahrávkách.

2.1.1 Blokové schéma

Blokové schéma na obr. 2.1 popisuje funkci programu. Po spuštění programu musí uživatel načíst dvě písně, které si přeje synchronizovat. Jedna z písní je referenční a jedna porovnávaná. Výběr referenční skladby uživatel provede tak, že tuto píseň nahraje pomocí tlačítka v horní polovině uživatelského rozhraní.

Při načítání písně do programu se skladby převzorkují na 22 050 Hz z důvodu následné práce s touto hodnotou v dalších blocích. Pokud jsou obě skladby správně načteny, následuje získávání různých vlastností, které jsou vstupními hodnotami pro DTW algoritmus.

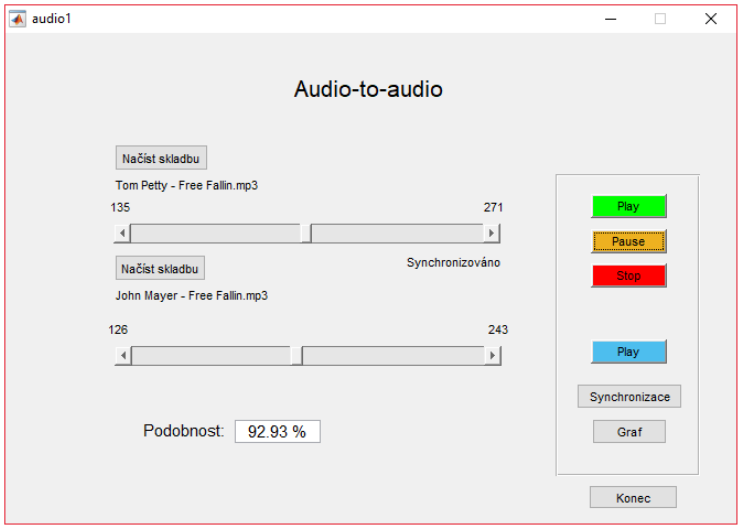
Úspěšné provedení algoritmu odblokuje přehrávací tlačítka a uživatel nyní může začít přehrávat písně a sledovat časové ukazatele a posuvníky jejich synchronizace. Dále má uživatel možnost zobrazit graf *vzdálenostní* cesty synchronizovaných písní. V uživatelském rozhraní je také údaj, který zobrazuje hodnotu podobnosti obou písní.



Obr. 2.1: Blokové schéma programu Audio-to-audio.

2.1.2 Řešení programu

Na obr. 2.2 vidíme ukázku programu Audio-to-audio při porovnávání dvou skladeb. Následně je uveden podrobný popis funkce programu.



Obr. 2.2: Ukázka programu Audio-to-audio.

Načtení písně

Prvním krokem je vybrání písně uživatelem. Píseň je do programu načtena funkcí *audioread*, která ukládá dvě informace – vzorkovací frekvenci nahrávky a matici, ve které počet řádků odpovídá vzorkům a počet sloupců počtu kanálů (standardně má nahrávka 2 kanály pro stereo bázi). Povolené formáty pro nahrání písně jsou wav a mp3.

Název písně a její délka v sekundách se zobrazí v uživatelském rozhraní nad příslušným posuvníkem. Jestliže uživatel zruší načítání písně, textový blok nad posuvníkem oznámí uživateli, že načtení proběhlo nesprávně a tlačítko synchronizace se zablokuje. Během načítání se skladba převzorkovává ze své původní vzorkovací frekvence na hodnotu 22 050 Hz. Toto převzorkování je důležité při přehrávání skladeb a jeho účel je takový, že pokud je skladba spuštěna od pozastavené části, vstupním parametrem pro přehrávání od této části je právě vzorkovací frekvence vynásobená s hodnotou pozastavení v sekundách.

V tomto bloku se také nastaví minimum a maximum posuvníku a také jeho krok. Krok posuvníku má dvě hodnoty – první hodnota odpovídá posunu při tažení jezdce myší a druhá představuje krok při kliknutí myší do dráhy jezdce [20]. Program však s těmito hodnotami pracuje jen pasivně, tedy že krokem posuvníku nastavuje velikost jezdce posuvníku. Pokud uživatel bude jezdcem posouvat nebo klikat do

dráhy, jezdec se sice posune, ale nemá to vliv na přehrávání skladby a na posunutí se z jiného časového okamžiku skladby.

Synchronizace

Dalším krokem je aplikace podobnostní funkce na převzorkované nahrávky. Přes ukazatele jsou vyvolány proměnné do příslušného bloku, kde se funkce počítá. Nahrávky jsou tedy vstupními argumenty pro tuto funkci. Aby se mohla aplikovat funkce *Dynamic Time Warping*, je třeba z nahrávek získat jeho spektrální a *chroma* vlastnosti.

Prvně se nahrávka převede do *miraudio* bloku pomocí funkce z Mirtoolboxu [15], jehož vstupní parametry zaručují, že se vlastnosti budou propočítávat pro vzorkovací frekvenci 20 050 Hz a v režimu mono. Režim mono ve funkci *miraudio* nesčítá dva signály do jednoho kanálu, ale odděluje kanály od sebe. Poté se provedou výpočty spektrálních a *chroma* vlastností za pomocí funkcí z Chroma toolboxu [14].

Nastavení těchto funkcí je převzato z lit. [2] a je vidět na výpisu 2.1. Blok *paramPitch.winLenSTMSP* určuje, jak přesně celý výpočet proběhne. Jedná se o nastavení délky obdélníkového okna (popsáno v sekci Chroma vlastnosti, chromagram), ve kterém se propočítává výkon. Podle lit. [2] vzorec (2.1) určuje aplikovaný časový úsek, kde w odpovídá délce okna a f_{vz} vzorkovací frekvenci:

$$T = \frac{w}{f_{vz}} = \frac{4410}{22050} = 200\,\text{ms}. \tag{2.1}$$

Hodnota 200 ms odpovídá zpoždění mezi jednotlivými časovými úseky. Tato hodnota může být přenastavená podle toho, jak přesnou synchronizaci uživatel potřebuje. Pro hodnotu 20 ms, tedy délku okna 441, je zpoždění natolik malé, že jej lidské ucho nedokáže zaznamenat. Tato hodnota odpovídá ideální synchronizaci, jejím problémem je však zatížení na výpočet. Zatímco pro délku okna 4410 je vektor hodnot podobnostní funkce pouze dvojnásobek reálné délky skladby, pro délku 441 je to dvacetinásobek. Tento rozdíl se promítne i do délky času provedení celé synchronizace a do zatížení procesoru počítače použitého pro spuštění programu. Přesto se DTW algoritmus počítá právě s touto délkou okna, protože je její přesnost přesně vhodná pro aplikaci programu.

Mimo nastavení přesnosti jsou na výpise 2.1 použity ještě dvě hodnoty týkající se parametrů pro počítání *CENS* vlastností. Jsou to *paramCENS.winLenSmooth*, která nastavuje hodnotu vyhlazení a *paramCENS.downsampSmooth*, která určuje násobek snížení vzorkovací frekvence.

Během provádění synchronizace se uživateli zobrazí dva *načítací* ukazatele. Tyto ukazatele informují uživatele o průběhu. Převádění písně do *miraudio* bloku má vložený ukazatel a informuje uživatele o uplynulém čase, kdy byla skladba úspěšně do bloku převedena. Druhý ukazatel monituruje průběh celé synchronizace, počínaje kliknutím na tlačítko DTW až po všechny výstupy tohoto bloku.

Výpis 2.1: Příklad nastavení vlastností.

```
paramPitch.winLenSTMSP=441;
paramCENS.winLenSmooth=21;
paramCENS.downsampSmooth=5;
```

Po získaní vlastností se vypočte podobnostní funkce *dynamicTimeWarpingItakura* z knihovny Audio Analysis Library [18]. Vstupní hodnoty funkce jsou v tomto pevném pořadí: testovaná skladba, referenční skladba a hodnota normalizace. Normalizace je nastavena na hodnotu 1, čímž zaručuje, že hodnota vzdálenosti je normalizovaná, neboli dělena délkou optimální cesty. Výstupem podobnostní funkce je matice optimální *vzdálenostní* cesty (výpis 2.2) a největší hodnota *vzdálenostní* matice.

Výpis 2.2: Příklad optimální *vzdálenostní* cesty.

```
%Příklad optimální cesty, kde první sloupec
%odpovídá X(deformovaná) a druhý Y(původní)
  1     1
  3     2
  5     3
  7     4
  9     5
 11     6
  .     .
  .     .
  .     .
399   367
401   368
402   369
403   370
404   371
```

Podobnost

Podobnost je v tomto případě počítána na základě znalosti definice integrálního počtu a grafu *vzdálenostní* cesty. Určitý integrál v mezích $< a, b >$ na ose x vyjadřuje obsah plochy pod funkční křivkou. Protože známe hodnotu každého bodu posloupnosti i její délku, tedy funkční hodnotu v každém bodě, může konečnou posloupnost místo integrálu zastupovat suma.

Matice *vzdálenostní* cesty je tedy odpovídajícím vstupem pro tuto úvahu. Posloupnost referenční skladby se v grafu *vzdálenostní* cesty zobrazuje na x-ové ose a její funkční křivka má lineární závislost $y = x$. Můžeme tedy určit referenční obsah pod touto křivkou například jako obsah pravoúhlého trojúhelníku.

Následuje výpočet rozdílu prvků *vzdálenostní* matice se stejným řádkovým indexem v absolutní hodnotě. Tímto zjistíme vzdálenost jednotlivých prvků. Výpočet rozdílů se provádí pro každou dvojici *vzdálenostní* matice a výsledek se sčítá pod názvem testovaný obsah. Absolutní hodnota je zde použita z důvodu, že prvky posuzované skladby mohou být větší a zároveň menší od referenčních prvků.

Podobnost je určena rozdílem referenčního a testovaného obsahu a je přepočítána na procenta. Výsledek se uživateli zobrazí přímo v uživatelském rozhraní. Úspěšnost 100 % se zobrazí pouze v případě, že uživatel nahrál stejné písně do obou výběrů skladeb.

Graf

Po stisknutí tlačítka *Graf* se uživateli promítne v samostatném okně graf *vzdálenostní cesty*. Na ose x nalezneme referenční nahrávku a její průběh bude mít lineární závislost (modrá čerchovaná čára). Osa y je vyhrazena pro testovanou nahrávku a vykresluje odchylky od referenční nahrávky (červená čára). Podobu tohoto grafu lze vidět např. na obr. 2.6 v kapitole Vyhodnocení.

Přehrávání

V uživatelském rozhraní se nacházejí dvě tlačítka pro funkci přehrávání. Zelené tlačítko *Play* se týká prvního posuvníku a tedy referenční skladby. Po jeho stlačení se začne přehrávat referenční skladba, zároveň se začnou pohybovat oba jezdci posuvníků. První jezdec se posouvá každou sekundu o hodnotu $\frac{1}{délka\ skladby}$. O každou sekundu se přičte i časový ukazatel, který počítá odehrané sekundy písně.

Druhý jezdec je závislý na I matici. Nejprve se přečte hodnota jezdce prvního posuvníku, vynásobí se délkou skladby a přibližnou délkou rozšíření signálu po DTW funkci. Takto zjistíme index řádku ve druhém sloupci matice. Do proměnné se uloží hodnota stejného řádku, nicméně prvního sloupce. Tato hodnota odpovídá posunutí porovnávané skladby vzhledem k referenční. Zpětně ji podělíme délkou skladby

a délkou rozšíření a tento výsledek uložíme jako pozici jezdce pro druhý posuvník. Jezdec posuvníku se tak může pohybovat rychleji, nebo pomaleji vzhledem k výsledkům DTW. Časový ukazatel druhého posuvníku je také závislý na DTW algoritmu a jeho hodnota vychází ze stejných proměnných. Pouze na konci je podělena pouze délkou rozšíření.

Modré tlačítko *Play* by mělo logicky mít opačnou funkci. Po jeho stlačení se sice začne přehrávat porovnávaná skladba, avšak posuvník nepřebírá referenční funkci, tedy neukazuje postup porovnávané skladby v časové posloupnosti a nepřirovnává referenční skladbu k porovnávané. Toto zobrazení by bylo velice snadno programovatelné, avšak by utrpěl už tak vysoký výpočetní čas.[1] Musela by totiž být spočítána druhá DTW funkce, teď s opačnými parametry, po jejíž spočítání bychom dostali výstup synchronizace referenční písně k porovnávané. Potom bychom mohli po stlačení tlačítka pozorovat plynoucí jezdec porovnávané skladby a měnící se jezdec skladby referenční.

Kvůli náročnosti na výpočetní čas bylo tedy od této myšlenky upuštěno a modré tlačítko *Play* tak plní v podstatě jen přehrávací funkci pro porovnávanou píseň. Všechny ukazatele se ale hýbou stejně, jako kdyby bylo stlačeno zelené tlačítko. Modrým tlačítkem tak uživatel může pouze poslechově kontrolovat, jestli se obě skladby nacházejí ve stejném místě.

V průběhu přehrávání jedné nebo druhé písně jsou obě přehrávací tlačítka zablokována, aby nedošlo k simultánnímu přehrávání obou skladeb. Po pozastavení nebo zastavení se tlačítka opět zpřístupní.

Zastavení

Tlačítko *Pause* má funkci klasického pozastavení skladby bez ztráty hodnot. Posuvníky setrvají na místě, kde se právě nacházely, když bylo tlačítko spuštěno a po stlačení jednoho z tlačítek *Play* se skladba přehrává a posuvník posouvá od tohoto místa.

Tlačítko *Stop* pak zastaví skladbu a vrátí oba posuvníky na nulovou pozici. Pokud spustíme jedno z tlačítek *Play*, přehrávání i pozice ukazatelů startují od nuly.

Konec

Tlačítko *Konec* ukončí uživatelské rozhraní programu, vyčistí *Command Window* v rozhraní Matlabu a zavře případně otevřený graf *vzdálenostní* cesty.

[1] Výpočetní čas pro DTW synchronizaci je zhruba 10 minut vzhledem k délce skladby. Je to uzpůsobeno zejména nízkým výkonem procesoru počítače a verzí Matlabu.

2.2 Audio-to-score synchronizace

Audio-to-score synchronizace má za úkol nahrát do programu audio nahrávku a její notový zápis vytvořený ve formátu, který nese hodnotu informace o průběhu notového zápisu. Použitím vhodného synchronizačního algoritmu se notový zápis sjednotí s nahrávkou a umožní uživateli při přehrávání nahrávky sledovat její průběh v notách. Ukazatel by měl být schopen ukazovat jednotlivé začátky taktů vybrané nahrávky.

2.2.1 Blokové schéma

Funkce tohoto programu je popsána pomocí blokového schématu na (obr. 2.3). Na začátku uživatel vybere referenční skladbu, která může být uložena ve formátu wav nebo mp3. Uživatel dále musí vybrat formát notového zápisu (xml, mxl nebo midi). Při nahrávání notového zápisu se spustí externí program MuseScore 3, který všechny formáty převede na pomocný formát wav, případně midi pro další práci v programu.

Když u obou proměnných došlo ke správnému načtení, následuje tlačítko pro synchronizaci. Bez synchronizace není možné skladbu přehrávat, ani sledovat ukazatele synchronizace, jelikož její výstupní proměnné pracují se všemi ostatními funkcemi. Z wav formátů se vypočítají příslušné vlastnosti, které vstupují do DTW algoritmu.

Po synchronizaci se zpřístupní tlačítko pro přehrávání a po jeho stlačení může uživatel začít sledovat přirovnání skladby k taktu a době. Toto sledování se neděje v reálném čase a proto, když uživatel chce získat synchronizační informace, musí v daný okamžik znovu ukazatele aktualizovat.

2.2.2 Řešení programu

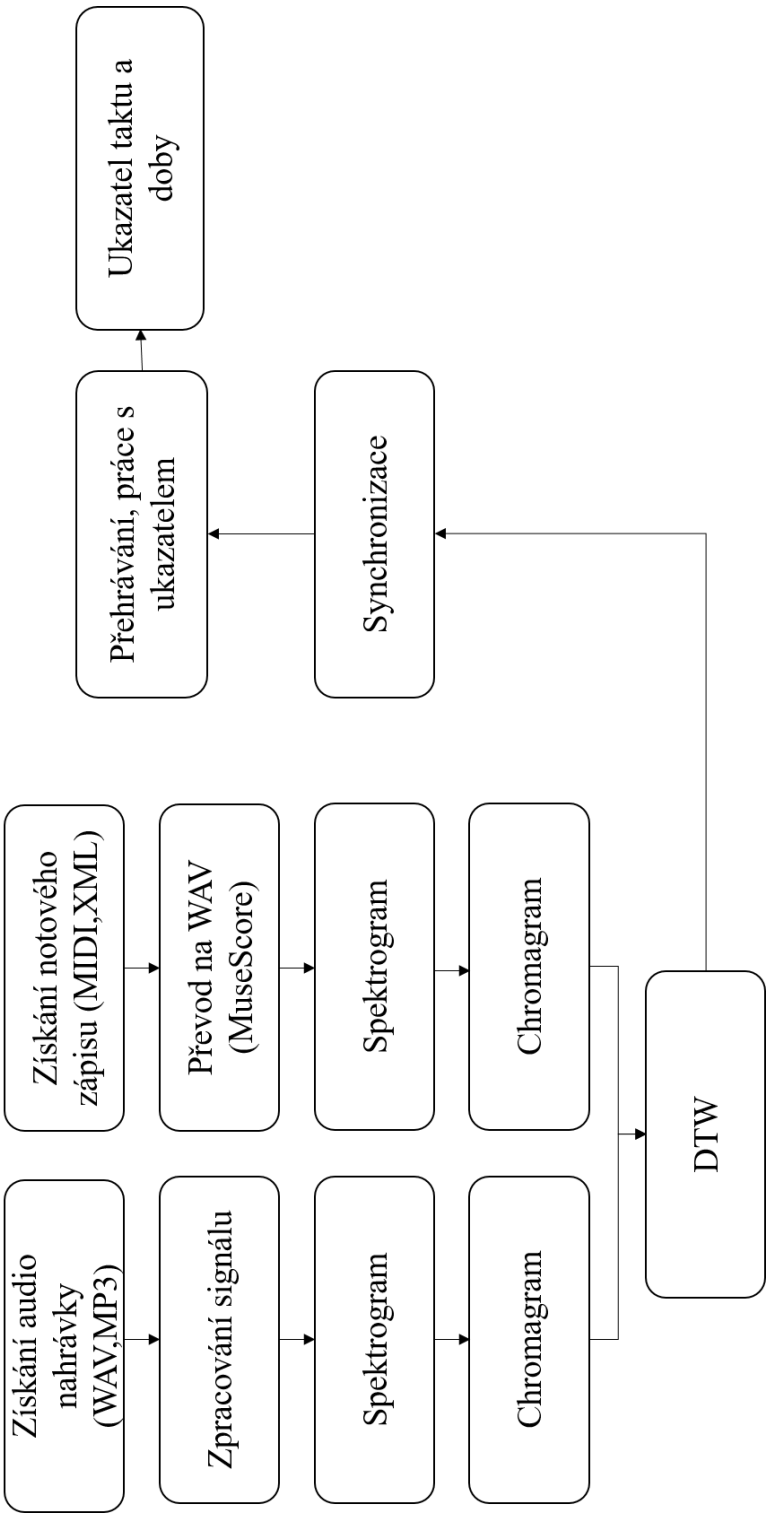
Nyní se budeme zabývat podrobným popisem funkce programu Audio-to-score, jehož ukázku při porovnávání skladby a not lze vidět na obr. 2.4.

Načtení písně

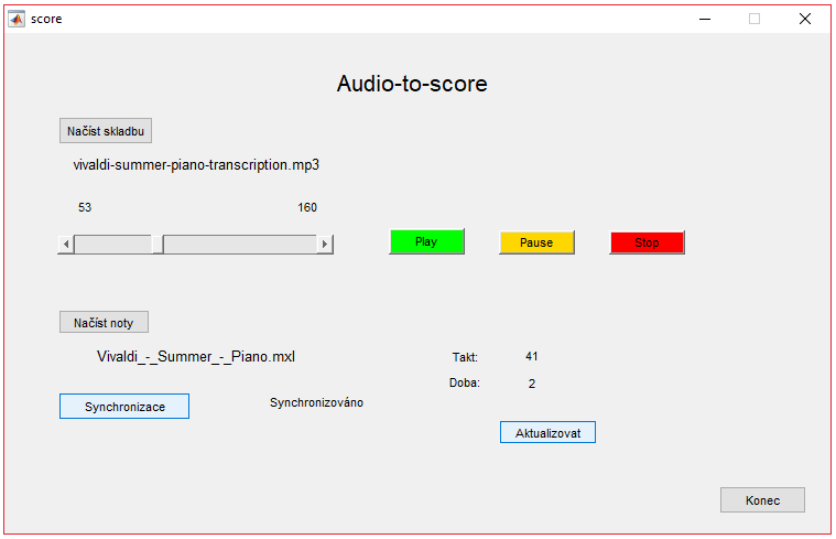
Načtení písně do programu je velmi podobná algoritmu načtení, který byl použit pro program Audio-to-audio (2.1.2). Potřebné proměnné se ukládají, aby mohly být zpracovávány v jiných blocích.

Načtení not

Načíst noty je umožněno ve třech formátech (xml, mxl a midi). Formát mxl je pouze zkomprimovanou verzí formátu xml, se kterým se ale pracuje totožně, proto dále budeme uvažovat dva formáty. Přestože pro oba formáty existuje několik toolboxů



Obr. 2.3: Blokové schéma programu Audio-to-score.



Obr. 2.4: Ukázka programu Audio-to-score.

a pomocných funkcí, které je dokážou do Matlabu nahrát a dále s nimi pracovat, žádná z nich nedokáže být dále zpracována pomocí DTW algoritmu. Proto při nahrání not Matlab pracuje s externím freewarovým programem MuseScore3 doplněný o plugin Batch conventor, který umožňuje konvertovat několik formátů do formátu požadovaného. V tomto případě je využita konverze z xml, příp. midi formátu do formátu wav. Pro wav formát funkce počítá tempo skladby v BPM a tento formát můžeme zpracovat v DTW synchronizaci.

Formáty jsou však přes funkci *switch* načítány odlišně. Pokud se jedná o midi, do proměnné *nmat* se ukládá sloupcová matice s notovými událostmi jdoucími po sobě. Data z této matice můžeme ale získat také přes tzv. *metadata*, která obsahují obecné vlastnosti písně, jako je název, tempo, či předznamenání. Z *metadat* je nutné získat čitatel taktového předznamenání, který je využíván pro indikaci taktu a doby v programu. Pro nalezení předznamenání je použita funkce *meter* z knihovny MIDItoolbox. Její omezení v používání je uvedeno v kap. 2.4.2.

Formáty mxl a xml jsou pouze formálně odděleny na dva samostatné *case* oddíly. Zpracování funkce však funguje na stejném principu. Protože každé notové zpracování může mít specifický zápis, není vlastnost taktového předznamenání uložena vždy ve stejné proměnné. Proto byla zvolena varianta, že formáty xml a mxl budou přes program MuseScore3 převedeny do midi a dále funkce bude postupovat stejně jako v případě midi formátů.

Noty ať už ve formátu midi, či xml musí být uloženy ve složce, ze které je spouštěn program Audio-to-score. Při konverzi přes MuseScore3 se do stejné složky uloží pomocný wav nebo midi soubor, se kterým pak program pracuje.

Název vybraných not se zobrazí pod příslušným tlačítkem. Pokud by se během

načítání not nepovedlo funkci správně provést nebo by uživatel zrušil výběr, tlačítko pro synchronizaci se zablokuje a nebude možné na něj kliknout, pokud uživatel nezopakuje výběr. Celý proces načítání je opět sledován ukazatelem progresu.

Synchronizace

Zde je použit stejný algoritmus jako v programu Audio-to-audio (2.1.2). Synchronizaci nelze provést v případě, že noty nebo skladba nebyly správně načteny či vůbec zvoleny.

Aktualizace

Stěžejním bodem tohoto programu je přiřazení časového okamžiku písně k notovému zápisu. Tuto synchronizaci zajišťuje tlačítko *Aktualizovat*. Tlačitko pracuje na principu přepočítávání temp skladby a notového zápisu a se *vzdálenostní* maticí z použitého DTW algoritmu.

V průběhu přehrávání se nad ukazatelem skladby odpočítává čas v sekundách od stlačení tlačítka *Play* a to buď od začátku – nuly nebo od posledního pozastaveného místa. V okamžiku, kdy je stlačeno tlačítko *Aktualizovat*, se do proměnné uloží aktuální odpočtený čas a je vynásoben dvaceti, což odpovídá přibližnému prodloužení délky skladby při DTW synchronizaci.

S touto proměnnou indexujeme pozici ve *vzdálenostní* matici. Odpočtený čas násobený dvaceti najdeme ve vzdáleností matici ve druhém sloupci a přečteme hodnotu z prvního sloupce, která odpovídá stejné pozici. Tuto hodnotu uložíme a podělíme dvaceti, abychom se zpětně dostali na stejný rozměr.

Získanou hodnotu následně násobíme tempem not, které jsme vypočítali během načítání. Protože tempo se počítá v BPM, je vyděleno šedesáti, abychom stále počítali se stejným rozměrem (sekundy). Vynásobená hodnota teď odpovídá celkovému počtu dob v odehraném úseku skladby, neboli počet dob za sekundu (BPM/60) násobíme odpočítanou hodnotou v sekundách od spuštění přehrávání, převedenou skrz *vzdálenostní* matici.

Při načítání notového zápisu do programu pomocí příslušné funkce získáme čitatel taktového předznamenání, který vyjadřuje, kolik dob je součástí jednoho taktu. Vydělením celkového počtu dob a počtu dob v jednom taktu, získáme odpovídající takt notového zápisu.

Dále je třeba určit, ve které době taktu se nacházíme. Vyjádříme čtyřmístný zbytek po dělení celkového počtu dob a dob v jednom taktu a stejně přesný zbytek po rozdělení jednoho taktu příslušným čitatelem taktového předznamenání. Pro dostatečnou přesnost byla zvolena právě tato čtyři místa za desetinou čárkou jako zbytek po dělení. Tyto dvě čísla porovnáváme s podmínkou: pokud je první zbytek

menší než druhý, jedná se o první dobu, pokud je první zbytek větší než druhý, ale zároveň menší než dvojnásobek druhého zbytku, jedná se o druhou dobu, apod. Celé porovnávání probíhá ve smyčce *for* a několika vnořených *if* funkcí. Je zde i opatření pro opakující se doby v taktu, což může být způsobeno pozdržením hodnot ve *vzdálenostní* matici, nebo pro poslední doby v taktu, jejichž některé zbytky po dělení by už odpovídali první době taktu dalšího.

Čísla taktu a doby se vypisují do uživatelského rozhraní tohoto programu. Pro každou novou synchronizaci programu je třeba stisknout tlačítko *Aktualizovat*, které přepočte okamžitou hodnotu přehrávání.

Přehrávání

Při přehrávání skladby má tradičně uživatel na výběr ze 3 možností. *Play* slouží pro přehrávání skladby, a to buď od začátku, nebo od dané části. Toto tlačítko je během samotného přehrávání vypnuto, aby nedocházelo k vícenásobnému přehrávání. Při přehrávání se odpočítává čas skladby v sekundách nad jejím ukazatelem. Dále *Pause* pro pozastavení skladby, po kterém může opět skladba pokračovat v přehrávání, nebo *Stop*, které vrátí všechny ukazatele na nulovou hodnotu.

Konec

Toto tlačítko slouží k ukončení programu. Zavře uživatelské rozhraní, vymaže soubory, které byly potřebné pro funkčnost programu a vymaže vzniklé příkazové řádky v *Command Window*.

2.3 Uživatelské rozhraní

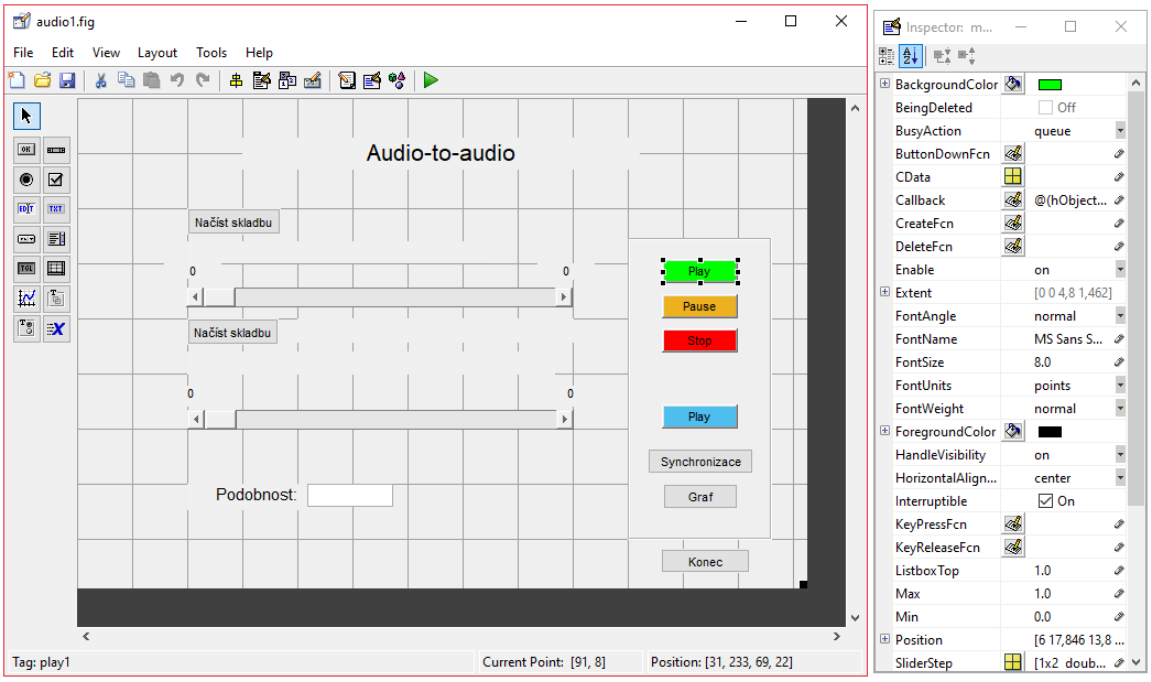
Uživatelské rozhraní slouží pro vizualizaci programů a pro snazší komunikaci uživatele s programem. Uživatel se nemusí zabývat čtením a porozuměním programátorského kódu. Práce s programem by naopak díky rozhraní měla být intuitivní. Základními požadavky na uživatelská rozhraní jsou: jednoduchost, provázanost a komplexnost. Jednoduchost má zajistit rychlý a jednoduchý pohyb v aplikaci, provázanost zaručuje návaznost kroků a komplexnost ošetřuje všechny eventuality. [20]

Grafické uživatelské rozhraní (*GUI – Graphical User Interface*) je vytvořeno v programu Matlab pomocí integrovaného nástroje pro tvorbu grafických prvků [20]. Tento nástroj je nazýván *GUIDE* (*Graphical User Interface Development Environment*) a obsahuje většinu grafických prvků potřebných pro chod programu [20].

Grafické prvky se dají ovládat a upravovat ze dvou souborů, které funkce *GUIDE* vytváří. Prvním je *m-script*, kde se prvky upravují pomocí různých funkcí a nastavením parametrů. Do grafů a textových bloků můžeme ukládat hodnoty nebo je zpětně do programu načíst, také můžeme nastavit chování ukazatelů a tlačítek.

Druhý soubor je tzv. *průvodce*, který má na starosti hlavně grafickou stránku programu. V tomto prostředí není třeba všechny prvky, které potřebujeme, vytvářet ručně, ale je možnost si vybrat z předem připravených objektů. Objektům lze přidělit výchozí nastavení, které se zobrazí vždy, když uživatel spustí program. Nastavení se provádí tzv. *Property Inspectorem* neboli oknem pro řízení vlastností [20], kde můžeme nastavovat například výchozí text pro textové bloky, názvy tlačítek, barvy i velikosti.

Na obr. 2.5 vidíme pracovní prostředí při vytváření uživatelského rozhraní pomocí *GUIDE* v Matlabu.



Obr. 2.5: Ukázka vytváření programu prostřednictvím funkce *GUIDE* a *Property Inspectoru.*

2.4 Vyhodnocení

2.4.1 Audio-to-audio

Testovaní nahrávek proběhne pomocí měření času skladby v sekundách ve vybraných časových okamžicích. Nejprve bude jedna nahrávka zvolena jako referenční a druhá testovaná a budeme sledovat odpočty jednotlivých skladeb v daných okamžicích. Následně se pořadí nahrávek otočí a z hodnot předchozího testu budeme získané hodnoty porovnávat ke zvoleným. Pokud bude předem zvolený okamžik 60 s u referenční písně a k němu bude programem přiřazena hodnota 57 vteřin testované skladby, při otočení skladeb bude referenční hodnota 57 s a budeme zkoumat hodnotu k ní přiřazenou. Časové okamžiky, ve kterých bude probíhat testování, budou rozděleny na 3 předem dané a 2 náhodné. Bude vyhodnocena odchylka prvních dvou testů v procentech se stupnicí popsanou níže. Pokud tento test určí úspěšnost nad 90 %, pokračujeme poslechovým testem.

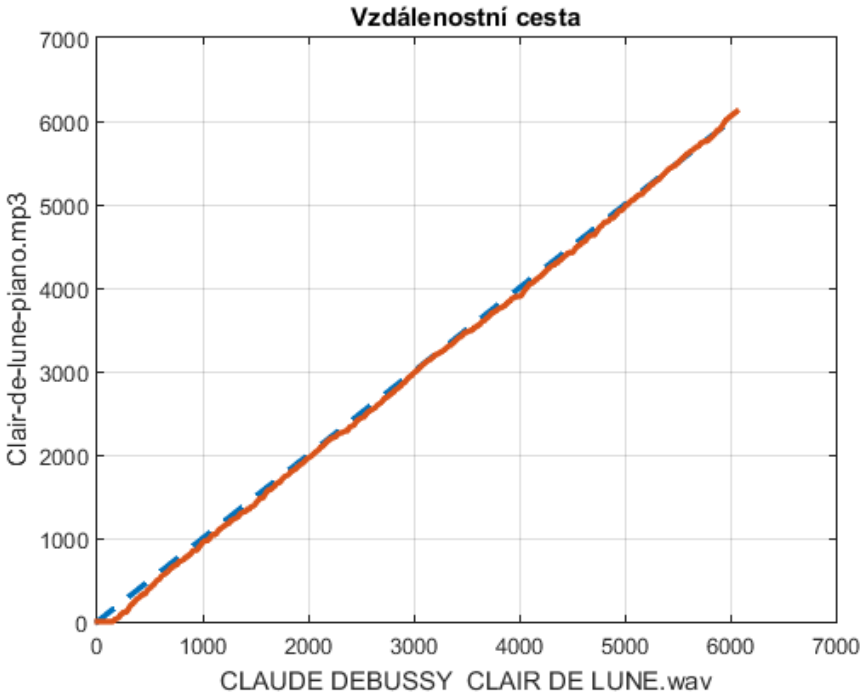
Třetí test bude poslechový a doplní dva předchozí testy. Bude proveden za účasti dvou osob, které nezávisle na sobě subjektivně vyhodnotí, o kolik se testovaná skladba liší v daném okamžiku od referenční. K porovnání nesrovnalostí bude použita stupnice, kde se žádné zpoždění (0 s) rovná 100 % shodě a poslední tolerovanou hodnotou bude zpoždění 9 sekund, která bude představovat shodu 10 %. Výslednou úspěšnost určí průměr poslechového testu.

První testování

Pro první testování byla vybrána klavírní skladba *Claire de Lune* od impresionistického skladatele Clauda Debussyho. Byly vybrány dva klavírní přednesy, které se drží notového zápisu i původního ztvárnění. Jejich podobnost určená programem nesla hodnotu 98 %. Graf *vzdálenostní* cesty těchto dvou skladeb najdeme na obr. 2.6. Celková úspěšnost tohoto testu je 100 %.

Tab. 2.1: První testování programu Audio-to-Audio.

Čas 1. písně [s]	60	120	240	138	201
Čas 2. písně [s]	57	117	239	135	196
Zpětné testování první písně [s]	60	120	241	138	200
Poslechový test (průměr) [%]	100	100	100	100	100



Obr. 2.6: První testování programu Audio-to-audio.

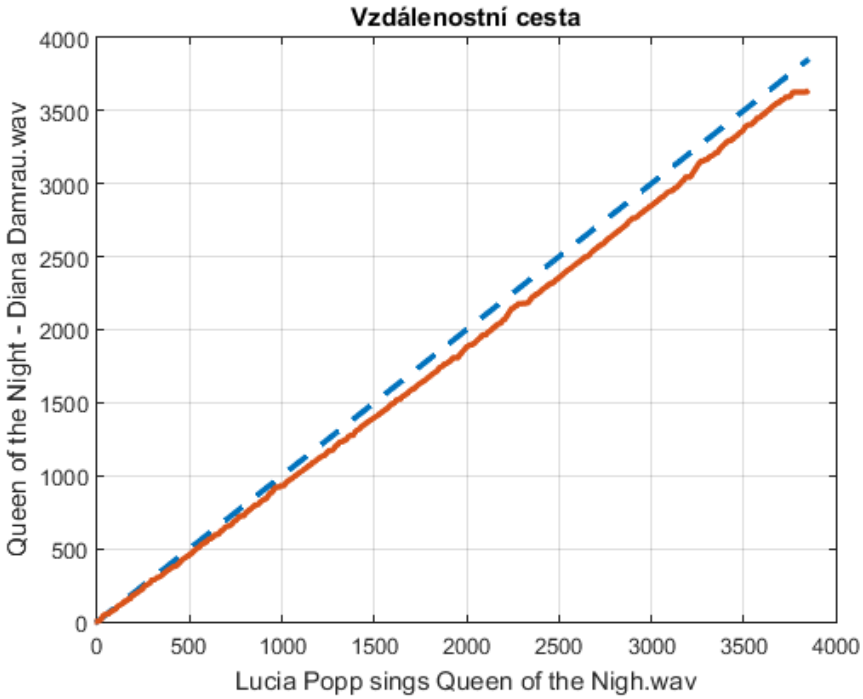
Druhé testování

Další testovací skladbou je árie *Královna noci* z opery *Kouzelná flétna* od skladatele Wolfganga Amadea Mozarta. Skladba byla vybrána především proto, aby byla demonstrována synchronizace nahrávky, ve které se nevyskytuje pouze jeden hudební nástroj. První verzí je provedení písně zpěvačkou Lucii Popp, druhou verzi nazpívala zpěvačka Diana Damrau.

Přesto, že jedna verze je nazpívaná jako studiová skladba a druhá verze je živá nahrávka pořízená během uvedení opery. Obě verze se pochopitelně striktně drží notového zápisu a klasického provedení. Jejich podobnost byla určena hodnotou 95 %, na obr. 2.7 je zobrazen graf *vzdálenostní* cesty. 94 % je celková úspěšnost pro druhé testování.

Tab. 2.2: Druhé testování programu Audio-to-Audio.

Čas 1. písně [s]	40	80	160	115	143
Čas 2. písně [s]	37	75	152	109	136
Zpětné testování první písně [s]	40	80	159	116	143
Poslechový test (průměr) [%]	95	100	100	90	90



Obr. 2.7: Druhé testování programu Audio-to-audio.

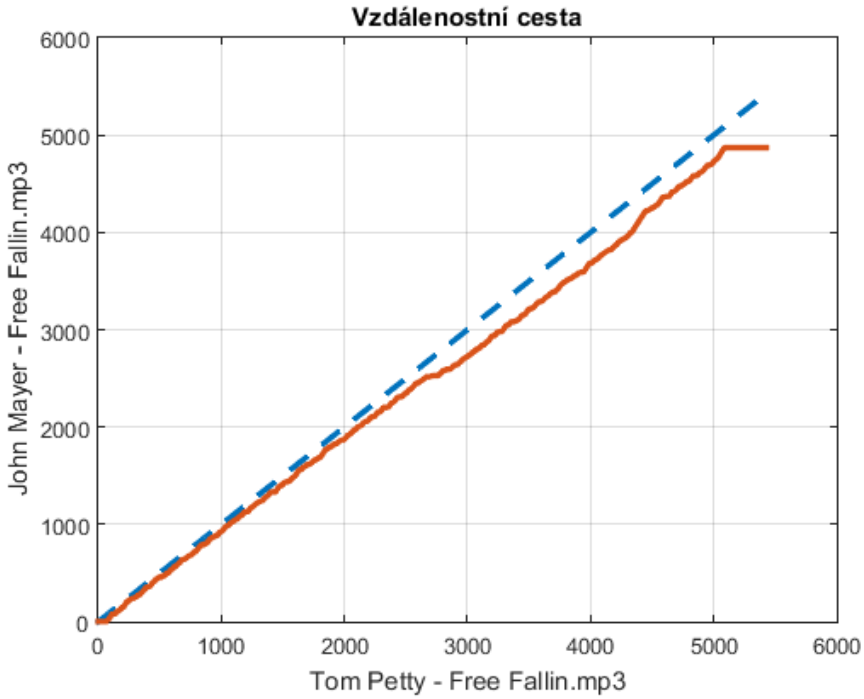
Třetí testování

Třetí testovací skladbou je rocková balada *Free Fallin'*. Původním interpretem této písně je Tom Petty, jehož verze zde bude referenční skladbou. Testovanou písní bude cover verze zpěváka Johna Mayera. Tato píseň má ukázat sjednocení moderní skladby, což s sebou nese určité znaky a problémy se sjednocením. Je to zejména to, že tyto skladby mají většinou předepsanou jen harmonickou složku, zatímco hlavní melodická linka se může libovolně měnit. Měnit se také může počet opakování refrénů, uvedení slok písně i nástrojové obsazení. Proto u těchto písní úspěšnost synchronizace výrazně klesá.

Podobnost těchto skladeb se rovná 92 % a *vzdálenostní* cestu vidíme na grafu 2.8. Úspěšnost programu v tomto případě dosahuje pouhých 29 %, a to zejména právě kvůli změnám melodické linky u testované nahrávky.

Tab. 2.3: Třetí testování programu Audio-to-Audio.

Čas 1. písně [s]	50	150	200	84	112
Čas 2. písně [s]	46	136	184	80	105
Zpětné testování první písně [s]	50	150	200	85	111
Poslechový test (průměr) [%]	0	20	35	70	20



Obr. 2.8: Třetí testování programu Audio-to-audio.

2.4.2 Audio-to-score

Testování tohoto programu bylo ztíženo nedostatkem správného testovacího materiálu. Zatímco midi formát je běžně a lehce dostupný, dostupnost formátu xml není dostačující. Jediný internetový portál, který bezplatně zprostředkovává formáty xml nebo mxl je *MuseScore*, tedy stejný portál, ze kterého lze stáhnout program MuseScore3, se kterým tento program Audio-to-score pracuje.

Další nevýhodou testování je skutečnost, že funkce *meter*, kterou program využívá pro počítání čitatele taktového předznamenání má svá velká omezení, a proto dokáže pracovat jen s předznamenáním $\frac{2}{4}$, $\frac{2}{8}$, $\frac{2}{2}$, $\frac{4}{4}$, $\frac{3}{4}$ a $\frac{3}{2}$. Zmíněná funkce navíc toto předznamenání umí určit pouze s úspěšností 80 % [17].

V neposlední řadě je nutné zmínit, že notový zápis může obsahovat několik prvků, se kterými program vytvořený v Matlabu neumí pracovat. Jsou to např. repetice nebo předtaktí. Program, který pracuje s notovým formátem xml, nedokáže přečíst, že se jedná o repetici, a proto počítá takty bez zaznamenání opakování. Tento problém odpadá s formátem midi, jelikož midi formát zapisuje notaci přesně tak, jak jde po sobě.

Problém předtaktí ale přetrvává pro oba formáty. Program začne počítat od první doby prvního taktu a neumí zaznamenat, že *zbývající část* taktu se nachází na samém konci skladby. Proto při testování takové skladby je program o takt napřed, než by správně měl být a v rámci taktů indikuje i nesprávnou dobu.

Proces testování proběhne tak, že midi nebo xml formát bude nahrán do programu MuseScore3, kde se zobrazí kompletní notový zápis. Do programu Audio-to-score bude nahrána skladba, notový zápis a následně bude provedena synchronizace. Skladba bude rozdělena podle své délky na 3 úseky, ve kterých proběhne aktualizace pro zobrazení taktu a doby. Tyto údaje budou porovnány s notovým zápisem v programu MuseScore3 (odpovídající takt a doba). Dále budou vybrána dvě náhodná místa během skladby, kde proběhne testování na stejném principu.

Po testování si z naměřených hodnot určíme celkovou dobu v celé písni. Pro toto určení budeme muset nejdříve zjistit čitatel taktového předznamenání a to buď z referenčního notového zápisu nebo pomocí funkce v Matlabu. Jestliže je v tab. 2.4.2 v prvním sloupci pro notový zápis hodnota taktu a doby 22/1 a víme, že čitatel taktového předznamenání je 3, spočítáme, že 63 dob se nachází ve 21 taktech a z 22. taktu připočteme další jednu dobu – dohromady tedy 64 dob. To samé provedeme s výsledkem programu a dostaneme hodnotu 61.

Úspěšnost testu pak bude spočítána trojčlenkou, kdy celkový počet dob notového zápisu je referenční – tedy 100 % a určujeme odchylky celkových dob stanovené programem. Celková úspěšnost programu je pak průměr z jednotlivých testů jednotlivé synchronizace.

První testování

Pro první testování byla vybrána věta *Presto* sonáty *Léto* ze známého díla Antonia Vivaldiho *Čtvero ročních dob* v úpravě pro klavír.

Tab. 2.4: Formáty porovnávané skladby a notového zápisu pro první testování.

	Forma	Formát
1	Píseň	.mp3
2	Noty	.mxl

Díky tomu, že se jedná o klasické dílo a bylo vybráno vzhledem k funkčnosti programu (tzn. taktové předznamenání, které lze rozpoznat, bez repetic, beze změn temp) je synchronizační program velice přesný. Podle testování se program liší nejvíce o jeden celý takt. Tato nepřesnost může být způsobena odchylkou, která vzniká při násobení dvaceti a zpětném dělení dvaceti času, který zobrazuje ukazatel písně nebo přesností DTW algoritmu. Další nepřesnost může vzniknout při samotném zastavení programu, ve vyhledávání a čtení z notového zápisu.

Tento test ukazuje 98% shodu.

Tab. 2.5: První testování programu Audio-to-score.

Čas písně [s]	30	90	120	45	68
Not. zápis (takt/doba)	22/1	74/2	99/1	35/1	54/3
Program (takt/doba)	21/1	73/1	99/1	34/2	54/3
Úspěšnost testu [%]	95	98	100	98	100

Druhé Testování

Druhou testovací skladbou byl světoznámý hit *Under Pressure* skupiny Queen ve spolupráci s Davidem Bowiem. Skladba byla převzata z alba *Hot Space*, zatímco notový zápis se skládá pouze z nástrojů, které ve skladbě hrají. Tato píseň byla vybrána proto, aby bylo ukázáno, že algoritmus dokáže pracovat i bez hlavní melodické linky zpěváka a dokáže synchronizovat pouze nástroje. Dalším kritériem pro výběr písně byl požadavek otestovat skladbu populární hudby.

Tab. 2.6: Formáty porovnávané skladby a notového zápisu pro druhé testování.

	Forma	Formát
1	Píseň	.mp3
2	Noty	.mid

Podle výsledků v 2.4.2 můžeme pozorovat, že odchylky mezi jednotlivými měřeními jsou výrazně větší. Je to způsobeno zejména tím, že popová hudba se většinou nedrží notového zápisu jako klasická hudba, ale umělci si často píseň uzpůsobují svým potřebám, ať už jde o noty, tempo či nástrojové obsazení. Úspěšnost tohoto testu odpovídá 88 %.

Tab. 2.7: Druhé testování programu Audio-to-score.

Čas písně [s]	50	150	200	82	125
Not. zápis (takt/doba)	25/2	76/3	96/1	40/2	60/3
Program (takt/doba)	22/2	63/3	85/1	40/2	53/2
Úspěšnost testu [%]	87	82	88	100	87

Třetí testování

Poslední testovaná skladba nese název *Pro Elišku* skladatele Ludwiga van Beethovena. V této skladbě se potkává několik vlastností, které vytvořený program nedokáže řešit. Jsou to zejména repetice, na kterých je skladba v podstatě založena. Jak už bylo zmíněno v úvodu testování, program není schopen vyhodnotit opakující se části, proto počítá takty dál místo toho, aby se vrátil na hodnotu počátku repetice. Další nevhodnou vlastností notového zápisu je předtaktí. Podobně jako u repetice, program nedokáže spočítat, že část taktu se nachází na začátku not a část na konci. Kvůli tomu můžeme pozorovat posunuté takty a doby o doby, které se nacházejí v předtaktí. Posledním kritériem pro úspěšnost je použití not, které mají stanovené taktové předznamenání, které bylo taktéž zmíněno v úvodu testování. Tato skladba je napsána v $\frac{3}{8}$ taktu. Navzdory tomu, že toto předznamenání nepatří do stanoveného výčtu, program jej správně určil. Toto správné určení však nemusí být pravidlem u jiných skladeb s nestanoveným předznamenáním.

Tab. 2.8: Formáty porovnávané skladby a notového zápisu pro třetí testování.

	Forma	Formát
1	Píseň	.mp3
2	Noty	.mid

Pokud bychom testovali originální notový zápis, testování bychom považovali za neúspěšné z důvodu popsaných výše, tedy velkými rozdíly v počítání taktů. Takty, které by byly zobrazeny ukazatelem synchronizace by musely být přepočítány vzhledem ke všem repeticím nacházejícími se v notách. Toto přepočítání je možné, však nežádoucí vzhledem ke skutečnosti, že program by měl sloužit pro rychlou orientaci v notách. Na druhou stranu program umožňuje testovat i notový zápis ve formátu midi, který zapisuje skladbu přesně tak, jak plyne v čase. To znamená, že pokud je ve skladbě repetice, formát midi zapíše tuto část stejně dvakrát za sebou. Takto upravený zápis pak můžeme lehce testovat v programu Audio-to-score a to dokonce i s vysokou úspěšností 97%.

Tab. 2.9: Třetí testování programu Audio-to-score.

Čas písně [s]	40	80	160	90	128
Not. zápis (takt/doba)	28/1	55/1	108/1	61/3	86/2
Program (takt/doba)	27/2	53/1	106/2	60/3	85/3
Úspěšnost testu [%]	97	96	98	98	99

3 ZÁVĚR

Bakalářská práce se zabývá problematikou dvou základních synchronizačních modelů a to nahrávky k nahrávce a nahrávky k jejímu notovému zápisu. Cílem práce bylo vytvořit a otestovat oba programy synchronizace.

V první části jsou popsány spektrální a *chroma* vlastnosti a jejich získávání z audio informace. Dále jsou uvedeny podstaty vybraných synchronizačních algoritmů *Dynamic Time Warping* a *Hidden Markov Model*, kde se uplatňují zmíněné vlastnosti audio nahrávky. V závěru této části jsou uvedeny aplikace dvou základních modelů synchronizace, kterými jsou Audio-to-audio a Audio-to-score.

Na základě teoretických znalostí byl za pomoci synchronizačního algoritmu *Dynamic Time Warping* vytvořen synchronizační program Audio-to-audio, který sjednocuje dvě uživatelem zadané skladby. Jeho výstupem je graf *vzdálenostní* cesty, hodnota podobnosti a synchronizace posuvníků pro přehrávání nahrávek. V druhé části byl program popsán pomocí blokového schématu, byly uvedeny jeho funkce a byl otestován na různých interpretacích skladby klasické hudby, stejně tak na interpretacích písně popové hudby.

Testování prokázalo, že tento program je vhodný pro skladby klasické hudby, kde očekáváme stejný nebo velmi podobný průběh nahrávek. Úspěšnost toho programu při testování dvou klasických skladeb dosahuje nad 95 %. Naopak úspěšnost testu skladby popového žánru dosáhla pouhých 29 %.

Během testování byla také zaznamenána neodladěná chyba programu, která spočívala v tom, že pokud se skladby často pozastavovali a znovu spouštěli, jezdce posuvníků nedosáhly své maximální hodnoty, přestože skladba už dohrála. Tato chyba může být způsobena velkým zaokrouhlováním, ke kterému v programu dochází především při rozšiřování a zpětném dělení odpočteného časového okamžiku a také při zaokrouhlení hodnoty, která je vstupním parametrem pro spuštění skladby od nějaké části.

Druhým vytvořeným programem byl program Audio-to-score. Tento program využívá pro konverzi notových formátů externí program MuseScore 3. Externí program umí notový zápis převést i do formátu, se kterým pracuje synchronizační algoritmus *Dynamic Time Warping*, a proto byl i zde tento algoritmus využit pro synchronizaci. V práci je uvedeno blokové schéma vytvořeného programu a popis jednotlivých funkčních částí programu. Dále byl program otestován opět na skladbách klasické i populární hudby.

I pro toto testování můžeme uvést, že skladby klasické hudby dosahují větší úspěšnosti testování (opět nad 95 %), je však nutné dodržet jisté náležitosti, které jsou uvedeny v kap. 2.4.1. Testování populární skladby v programu Audio-to-score proběhlo na 88 %, avšak i pro testování těchto skladeb platí stejné dodržení všech

náležitostí.

Po proběhlém testování bylo zvažováno reálné uplatnění těchto programů. Největším problémem je zde bezpochybně výpočetní zatížení a tedy rychlost samotné synchronizace, která se v moderní době nedá považovat za uspokojivou. Jedna synchronizace průměrně dlouhé písně (230 s) trvá na počítači běžného uživatele zhruba 10 minut. Je to způsobeno zejména nízkým výkonem procesoru a verzí Matlabu, kterou je uživatel schopen získat.

Program Audio-to-audio by mohl po odladění všech nepřesností najít uplatnění v oblasti posuzování autorských práv skladatelů. Program Audio-to-score je sice dobře realizovatelný, avšak existuje několik podobných programů, které ke svému spuštění nepotřebují rozhraní Matlab a dokáží k samotné synchronizaci promítnout i notový zápis.

LITERATURA

[1] KRUMHANSL, C. The psychological representation of musical pitch in a tonal context. In *Cognitive Psychology* [online]. Volume 11, Issue 3, 1979, Pages 346-374 [cit. 24. 5. 2019]. Dostupné z URL: <https://www.sciencedirect.com/science/article/pii/0010028579900161> ISSN 0010-0285.

[2] MÜLLER, M., EWERT, S. Chroma Toolbox: MATLAB implementations for extracting variants of chroma-based audio features. In *Proceedings of the 12th International Conference of Music Information Retrieval (ISMIR)* [online]. 2011, Pages 215-220 [cit. 24. 5. 2019]. Dostupné z URL: <https://pdfs.semanticscholar.org/31cf/b91158031038ad16218997cc9724a9ab5c66.pdf>. ISBN 0615548652.

[3] WALKER, J., DON G. *Mathematics and Music: Composition, Perception, and Performance.* CRC Press, 2013. ISBN 978-1482208504.

[4] TENOUDJI, F. *Analog and Digital Signal Analysis: From Basics to Applications.* Springer, 2016. ISBN 978-3-319-42382-1.

[5] MÜLLER, M. *Fundamentals of music processing.* New York, NY: Springer Berlin Heidelberg, 2015. ISBN 978-3-319-21944-8.

[6] MIOTTO, R., ORIO N. A music identification system based on chroma indexing and statistical modeling. In *Proceedings of the 9th International Conference of Music Information Retrieval (ISMIR)* [online]. 2008, Pages 301-306 [cit. 24. 5. 2019]. Dostupné z URL: <https://ismir2008.ismir.net/papers/ISMIR2008_234.pdf>. ISBN 0615248497.

[7] RABINER, L.R. A tutorial on hidden Markov models and selected applications in speech recognition. In *Proceedings of the IEEE* [online]. 1989, 77(2), 257-286 [cit. 24. 5. 2019]. Dostupné z URL: <http://ieeexplore.ieee.org/document/18626/>. ISSN 00189219.

[8] GIANNAKOPOULOS, T., PIKRAKIS A. *Introduction to audio analysis: a MATLAB approach.* Oxford: Academic Press, 2014. ISBN 978-0-08-099388-1.

[9] LERCH, A. *Audio content analysis: an introduction.* 2012, Wiley-IEEE Press. ISBN 978-1118266823.

[10] STAMP, M. *A Revealing Introduction to Hidden Markov Models* [online]. 2004 [cit. 24. 5. 2019]Dostupné z URL: <https://www.cs.sjsu.edu/~stamp/RUA/HMM.pdf>.

[11] MAEZAWA, A., OKUNO, H. Bayesian Audio-to-Score Alignment Based on Joint Inference of Timbre, Volume, Tempo, and Note Onset Timings. *Computer Music Journal* [online]. 2015, 39(1), Pages 74-87 [cit. 24. 5. 2019]. Dostupné z URL: <https://ieeexplore.ieee.org/document/7067563>. ISSN 0148-9267.

[12] CANDY, J. *Bayesian signal processing: classical, modern, and particle filtering methods* Hoboken: Wiley, 2009. ISBN 978-0-470-18094-5.

[13] Signal Processing Toolbox – MATLAB. *MathWorks - Makers of MATLAB and Simulink* [online]. [cit. 24. 5. 2019]. Dostupné z URL: <https://www.mathworks.com/products/signal.html>.

[14] MÜLLER, M., EWERT, S. Chroma Toolbox 2.0 [online]. 2011 [cit. 24. 5. 2019]. Dostupné z URL: <http://resources.mpi-inf.mpg.de/MIR/chromatoolbox/>.

[15] LARTILLOT, O., TOIVIAINEN, P., SAARI P., EEROLA, T. MIRtoolbox 1.7 [online]. 2017 [cit. 24. 5. 2019]. Dostupné z URL: <https://www.jyu.fi/hytk/fi/laitokset/mutku/en/research/materials/mirtoolbox>.

[16] TOIVIAINEN, P., EEROLA, T. MIDI Toolbox 1.1 [online]. 2016 [cit. 24. 5. 2019]. Dostupné z URL: <https://github.com/miditoolbox/1.1>.

[17] TOIVIAINEN, P., EEROLA, T. *MIDI toolbox: MATLAB tools for music research* [online]. 2004 [cit. 24. 5. 2019]. Dostupné z URL: <https://www.researchgate.net/publication/228092590_MIDI_toolbox_MATLAB_tools_for_music_research>.

[18] Matlab Audio Analysis Library – MATLAB Central. *MathWorks - Makers of MATLAB and Simulink* [online]. [cit. 24. 5. 2019]. Dostupné z URL: <https://www.mathworks.com/matlabcentral/fileexchange/45831-matlab-audio-analysis-library?focused=3812592&tab=function>.

[19] MuseScore 3 [online]. [cit. 24. 5. 2019]. Dostupné z URL: <https://musescore.org/cs>.

[20] ZAPLATÍLEK, K., DOŇAR, B. *MATLAB: tvorba uživatelských aplikací.* Praha: BEN - technická literatura, 2004. ISBN 80-7300-133-0.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

a, b	Obecné posloupnosti
BPM	Údery za minutu – Beats per minutes
$\boldsymbol{C}$	Cenová matice
CENS	Chroma Energy Normalized Statistics
CQT	Constant Q transform
d	Stav
$\boldsymbol{D}$	Vzdálenostní matice
d_t	Pravděpodobnost stavu
DTW	Borcení časové osy – Dynamic Time Warping
f_{vz}	Vzorkovací frekvence
GUI	Grafické uživatelské rozhraní – Graphical User Interface
GUIDE	Graphical User Interface Development Environment
HMM	Skrytý Markovův model – Hidden Markov Model
HSMM	Poloskrytý Markovův model – Hidden Semi-Markov Model
Hz	Hertz
K	Velikost posloupností a, b
L	Délka vzdálenostní cesty
L_d	Délka stavu
LDS	Lineární dynamický systém – Linear dynamic system
LHA	Latentní harmonická alokace – Latent harmonic allocation
n, m	Prvek posloupnosti
N, M	Délka posloupnosti
MIDI	Musical Instrument Digital Interface
ms	Milisekunda
p	Prvek posloupnosti vzdálenostní cesty
P	Posloupnost vzdálenostní cesty
PDF	Hustota pravděpodobnostní funkce – Probability density function
s	Sekunda
STFT	Krátkodobá Fourierova transformace – Short Time Fourier Transform
T	Perioda
T_d	Tempová křivka
ω	Délka okna

SEZNAM PŘÍLOH

A UŽIVATELSKÝ MANUÁL PROGRAMU AUDIO-TO-AUDIO

Spustit program lze dvěma způsoby a to buď zadáním názvu *m-scriptu* do příkazového řádku Matlabu nebo otevřením *m-scriptu* a tlačítkem *Run* nacházejícím se v Editoru (klávesová zkratka F5). Pro správný chod programu, je třeba prvně načíst dvě skladby, tlačítko v horní polovině uživatelského rozhraní pro referenční skladbu, tlačítko v dolní polovině pro skladbu testovanou. Obě skladby je možné nahrát ve formátu wav nebo mp3. O průběhu načítání je uživatel informován ukazatelem.

Pokud byly skladby správně načteny, jejich název se ukáže nad posuvnými ukazateli. Dále je třeba sledovat indikaci synchronizace, která je umístěna pod druhým ukazatelem. Indikace se mění v závislosti na provedených příkazech uživatelem. Červený nápis *Nesynchronizováno* informuje uživatele, že písně byly právě načteny nebo změněny.

K synchronizace skladeb slouží tlačítko *Synchronizace*. Průběh synchronizace je zobrazena na ukazateli progresu. Až tento ukazatel nabude maximální hodnoty, proces je ukončen. Skladby jsou nyní synchronní, zobrazí se podobnost skladeb a indikace se změní na *Synchronizováno*.

V této chvíli je možno skladby přehrávat, stopovat nebo zastavovat přes příslušná tlačítka. Přehrávat druhou skladbu je možné vždy až po pozastavení první skladby, tlačítka pro pozastavení a zastavení jsou vždy aktivní. Uživatel nyní sleduje jezdce posuvníků, které zobrazují synchronizaci stejně jako časové ukazatele nad jednotlivými posuvníky. Dalším výstupem programu může být graf *vzdálenostní* cesty, který se zobrazí stlačením tlačítka *Graf*. Tlačítko *Konec* slouží pro ukončení uživatelského rozhraní.

B UŽIVATELSKÝ MANUÁL PROGRAMU AUDIO-TO-SCORE

Spustit program lze dvěma způsoby a to buď zadáním názvu *m-scriptu* do příkazového řádku Matlabu nebo otevřením *m-scriptu* a tlačítkem *Run* nacházejícím se v Editoru (klávesová zkratka F5). Po spuštění programu jsou uživateli nabídnuta dvě načítací tlačítka a to načítání skladby, kde uživatel může vybrat ze dvou formátů (wav a mp3) a tlačítko pro načtení not, kde jsou povoleny tři formáty (xml, mxl a midi). Aby mohly být soubory nahrány do programu, musí se nacházet ve složce disku počítače, ze které je program spouštěn. Název skladby nebo not se zobrazí pod příslušným tlačítkem. Pokud načtení neproběhlo správně, nebude možné kliknout na tlačítko Synchronizace a uživatel musí znovu načíst skladbu nebo hledat příčinu neúspěchu.[1]

Po správném načtení následuje tlačítko Synchronizace, které provede DTW algoritmus. O průběhu funkce je uživatel informován *načítacím* ukazatelem. Provedení algoritmu indikuje textové pole vedle tlačítka, které zobrazuje dvě varianty. *Synchronizováno* značí úspěšnost, *nesychronizováno* svědčí o tom, že se algoritmus provedl nesprávně nebo jsou písně natolik odlišné, že je nebylo možno sjednotit.

V tomto okamžiku se odblokuje tlačítko *Play* a uživatel může začít přehrávat skladbu. Zobrazení synchronizace má za úkol tlačítko *Aktualizovat*. Nad tímto tlačítkem jsou dvě textová pole, jedno pro ukázání taktu a druhé pro dobu. Tyto pole se aktualizují vždy po zmáčknutí příslušného tlačítka. Během přehrávání je zablokováno tlačítko *Play*, ostatní tlačítka pro proces přehrávání jsou vždy aktivní.

Pokud uživatel načte novou píseň nebo noty, textové pole u synchronizace upozorní, že nové soubory nejsou synchronizovány, uživatel tedy pro správné zobrazení veličin sjednocení musí znovu synchronizovat. Program ukončuje tlačítko *Konec*, která zavře uživatelské rozhraní a smaže všechny pomocné soubory, které se vytvořili během používání programu.

[1] Častou příčinou neúspěchu načítání do Matlabu může být název souboru, který nesmí obsahovat žádné mezery a nepovolené znaky.

C OBSAH PŘILOŽENÉHO DVD

Obsahem přiloženého DVD jsou vytvořené programy Audio-to-audio a Audio-to-score – jejich m-scripty a uživatelská rozhraní. Dále obsahuje uživatelské příručky pro oba programy a elektronickou verzi této bakalářské práce.

)
/ ..... kořenový adresář přiloženého DVD
- `audio1.m` ..... M-script
- `audio1.fig` ..... Uživatelské rozhraní
- `score.m` ..... M-script
- `score.fig` ..... Uživatelské rozhraní
- `Uživatelský manuál programu Audio-to-audio.txt` ..... Uživatelská příručka
- `Uživatelský manuál programu Audio-to-score.txt` ..... Uživatelská příručka
- `Bakalářská práce - Tereza Búliková` ..... Elektronická verze BP