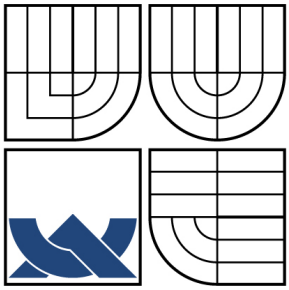
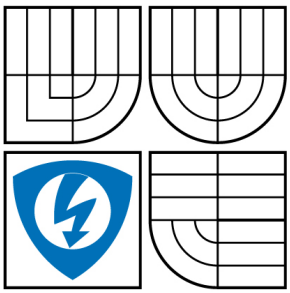


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

PŘEVOD NOT JEDNOHLASÉ MELODIE ZE ZVUKOVÉHO SIGNÁLU DO PROTOKOLU MIDI

CONVERSION OF MONOPHONIC MELODY FROM THE AUDIO SIGNAL INTO THE MIDI PROTOCOL STREAM

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

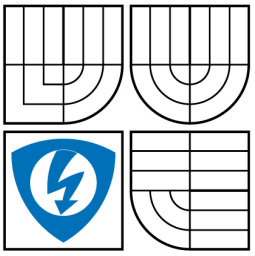
AUTOR PRÁCE — Bc. JAN KRUPIČKA

AUTHOR

VEDOUCÍ PRÁCE — Ing. JAROMÍR MAČÁK

SUPERVISOR

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student: Bc. Jan Krupička ***ID:*** 83576
Ročník: 2 ***Akademický rok:*** 2008/2009

NÁZEV TÉMATU:

Převod not jednohlasé melodie ze zvukového signálu do protokolu MIDI

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte metody detekce kmitočtu dominantní složky spektra zvukového signálu, realizujte je v prostředí Matlab a vyberte nejpřesnější metodu. Detekovanému kmitočtu najděte nejbližší výšku tónu v temperované soustavě ladění a přiřaďte číslo noty dle protokolu MIDI a celou melodii zapište do standardního MIDI souboru. Odladěný algoritmus implementujte v programovacím jazyce C++.

DOPORUČENÁ LITERATURA:

[1] PSUTKA, J, et al. Mluvíme s počítačem česky. 1. vyd. Praha : ACADEMIA, 2006. 752 s. ISBN 80-200-1309-1.
[2] ATASSI, H. Metody detekce základního tónu řeči. Elektrorevue [online], 2008. Dostupný z http://www.elektrorevue.cz/cz/clanky/zpracovani-signalu/5/metody-detekce-zakladniho-tonu-reci/. ISSN 1213-1539.
[3] FORRÓ, D. Svět MIDI. GRADA publishing, 1997. ISBN 80-7169-415-6.

Termín zadání: 9.2.2009 ***Termín odevzdání:*** 26.5.2009

Vedoucí práce: Ing. Jaromír Mačák

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práve třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

Anotace

Úkolem diplomové práce je porovnat možnosti detekce frekvence základního tónu v jednohlasé melodii. V práci je uveden přehled detekčních metod vycházejících z metod pro detekci základního tónu řeči. Tyto metody využívají časové, spektrální a kepstrální oblasti. Jsou porovnávány z hlediska úspěšnosti detekce na různých testovacích signálech. Součástí práce je implementace těchto metod v programovacím prostředí Matlabu.

V úvodu práce jsou popsány základní vlastnosti hudebního signálu. Uveden je přehled různých soustav ladění a popsána problematika určení výšky tónu z detekované frekvence.

V další části se práce zabývá problematikou MIDI. Uvedena je stručná historie a základní popis MIDI protokolu. Větší část je věnována struktuře a způsobu zápisu do takzvaného standardního MIDI souboru (SMF) a způsobu převodu detekovaných frekvencí na čísla not dle MIDI protokolu.

Posledním úkolem práce bylo vytvořit program v jazyce C. Jeho úkolem je analyzovat jednohlasou melodii ve formě zvukového signálu a přiřadit detekovaným tónů čísla not dle MIDI, které jsou následně zapsány do SMF. Pro tento program byla vybrána detekce základního tónu v časové oblasti, která vykazovala nejlepší výsledky v porovnání s ostatními. Pro urychlení výpočtu korelace byl použit algoritmus takzvané rychlé korelace.

Program byl vytvořen ve formě MEX souboru, který je možno využít v programovacím prostředí MATLAB prostřednictvím jeho externího rozhraní pro jazyky C a Fortran. Byl také připojen také popis knihovny FFTW, použité pro výpočty Fourierovy transformace.

Klíčová slova

MIDI, detekce základního tónu, MEX funkce, knihovna FFTW, hudební signál, Matlab

KRUPIČKA, J. *Převod not jednohlasé melodie ze zvukového signálu do protokolu MIDI*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 52 s. Vedoucí diplomové práce Ing. Jaromír Mačák.

Abstract

The aim of this thesis is to compare possibilities of the pitch extraction methods in the monophonic melody. There is presented the overview of the methods based on the speech pitch extraction techniques in the thesis. These methods uses frequency, time and „cepstral“ domain. They are compared in the term of success of the detection of various test signals. The part of the thesis specification is the implementation of these methods in Matlab.

There are described basics of sound features at the beginning of this work. The overview of the musical tuning systems is mentioned and there is described a problem of the determination of the pitch from the detected frequency.

There is considered an issue of MIDI protocol in the next part of the work. There are described the brief history and the essential structure of MIDI protocol.

The last task of the work was the creation of the program in C language. The purpose of the program is to analyze the monophonic melody in audio signal form and assign note numbers to the detected sounds according to MIDI specification. After that the numbers are written into the standard MIDI file (SMF).

There was implemented a correlation pitch detection algorithm in this program. It had the best results as compared to the others. There was used the fast correlation based on Fast Fourier transformation to accelerate computing of the correlation. The program was created in the form of MEX function, which provides various possibilities to be used in Matlab. There was also attached the description of the FFTW library, which was used to compute FFT.

Keywords

MIDI, basic pitch extraction, MEX function, FFTW library, music signal, Matlab

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma "**Převod not jednohlasé melodie ze zvukového signálu do protokolu MIDI**" jsem vypracoval samostatně pod vedením vedoucího diplomové 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 uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.“

V Brně dne …… ……..

(podpis autora)

PODĚKOVÁNÍ

Děkuji vedoucímu diplomové práce Ing. Jaromíru Mačákovi za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

V Brně dne …… ……..

(podpis autora)

Obsah

Seznam obrázků

Seznam tabulek

Úvod

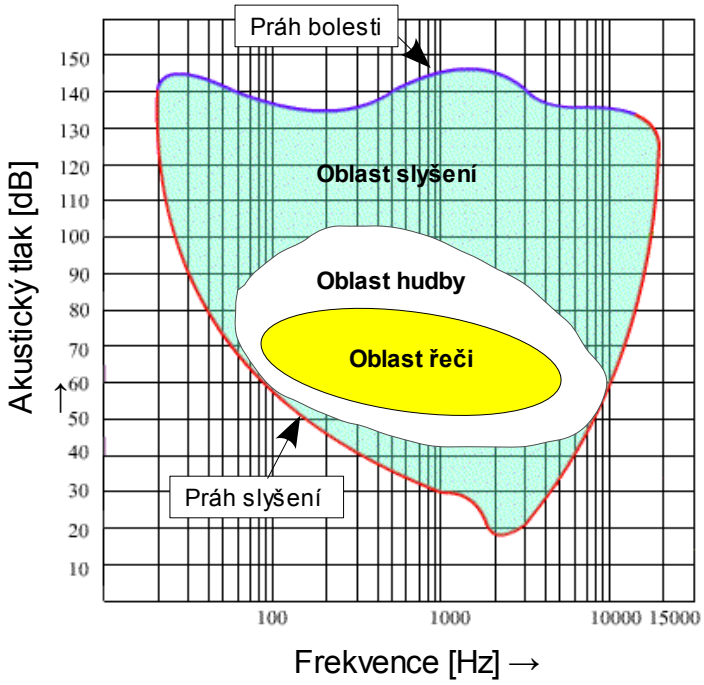
Úkolem této diplomové práce je porovnat metody detekce kmitočtu dominantní složky spektra zvukového signálu a vybrat nejpřesnější metodu. Poté detekovanému kmitočtu přiřadit číslo noty dle protokolu MIDI. Jednotlivé noty následně zapsat do standardního MIDI souboru (SMF). Dalším úkolem bylo vybrat jednu metodu detekce základního tónu. Vytvořit v programovacím jazyce C program, který bude převádět zvukový soubor wav na MIDI soubor SMF.

Pro detekci dominantního kmitočtu v jednoduchých melodiích lze použít některé metody detekce základního tónu řeči. Vybral jsem tedy pro porovnání metody, které využívají časové oblasti, spektrální metodu a kepstrální metodu. Práce se věnuje ověření, zda jsou tyto metody vhodné, po změně jejich parametrů, i pro zpracování hudebního signálu. Dále jsou porovnávány a je vyhodnocena nejpřesnější metoda. Oproti detekci základního tónu řeči jsou zde specifické rozdíly a problémy. Nejvýznamnější je asi rozdílný frekvenční rozsah. Řečový signál využívá pouze omezené kmitočtové spektrum, oproti tomu hudební signál většinou využívá téměř celé slyšitelné spektrum. S tím souvisí i různá nastavení délky segmentů při segmentaci signálu. Dalším význačným problémem je neznalost nástroje, na který je melodie hrána. Každý nástroj má ve spektru a v čase velmi specifický průběh a jeho znalost by velmi zjednodušila detekci základního tónu. V této práci se budu zabývat porovnáním několika metod při zpracování signálu různých hudebních nástrojů a pokoušet se vybrat univerzální metodu.

Praktická část práce popisuje vytvořený program v jazyce C. Program umožňuje ze zadaného zvukového signálu detekovat základní tón a určit noty dle MIDI protokolu. Poté tyto zapsat do standardního MIDI souboru. Zvolil jsem formu funkce využitelné v programu Matlab – takzvaný MEX soubor. Program se skládá z několika částí, které je možné využít samostatně. Forma MEX souboru umožňuje využít širokých možností programovacího prostředí Matlab.

1 Hudební signál

Ve slyšitelném zvukovém spektru lze vyznačit oblasti, ve kterých se vyskytuje hudební a řečový signál. Jak je zřejmé z obrázku 1.1 [12] oblast řeči je podskupinou oblasti hudby. Rozdíly jsou především v kmitočtovém rozsahu oblastí, kdy řeč má tento rozsah omezenější. Frekvence základního tónu řeči se pohybuje v rozsahu 60-400 Hz [9]. Také dynamika řečového signálu bývá obvykle menší. Při práci s hudebním signálem je třeba uvažovat téměř celou oblast vnímání lidského sluchu. Frekvence základních tónu v hudbě pokrývá téměř celý rozsah lidského sluchu. Specifikace MIDI umožňuje zápis not se základními frekvencemi od 8,18 Hz do 12,54 kHz. V praxi samozřejmě není využitá úplně, kmitočty základních tónů používaných v klasické hudbě se obvykle pohybují mezi 16 Hz a 8400Hz. V hudbě moderní, zejména elektronické se však mohou vyskytovat i tóny s vyšší základní frekvencí. Dynamika je v hudbě mnohem větší, než u řečového signálu. U vážné hudby se rozsah intenzity může pohybovat od nízkých hodnot blízkých prahu slyšení po hodnoty okolo 110dB.



Obr. 1.1: Charakteristické oblasti vnímání akustického signálu lidským sluchem [12]

1.1 Tón

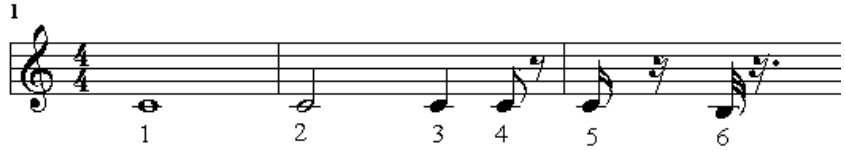
Tón je základní stavebním kamenem v hudbě a je za něj považován každý zvuk se stálou frekvencí. Toto označení se však také používá pro pojmenování intervalu mezi jednotlivými celými tóny například C a D. Značka pro tón je tedy v hudbě nota, která se zapisuje do notové osnovy. Právě přiřazením noty respektive tónu určité detekované frekvenci se budu zabývat.

Základní vlastnosti tónu jsou:

- výška – je dána frekvencí
- délka – doba po kterou tón zní
- síla – výchylka signálu
- barva – závisí na počtu vyšších harmonických, jejich poměru apod.

1.2 Délka tónu

Délka tónu je tedy definována jako délka časového úseku po kterou tento tón zní. Pro identifikaci se v hudbě používají různé typy not. Ty jsou v grafickém vyjádření rozlišeny tvarem noty.



Obr. 1.2: Grafická reprezentace not

Na obrázku 1.2 jsou znázorněny následující noty: 1-celá, 2-půlová, 3-čtvrťová, 4-osminová, 5-šestnáctinová, 6-dvaatřicetinová. Nejdelší průběh má nota celá (4 doby), poloviční nota půlová (2 doby), dále nota čtvrťová (1 dobu), osminová (1/8 doby) atd. Délka doby je pak určena určena tempem skladby, to se udává v BPM – počet úderů za minutu a má buď přesné číselné vyjádření nebo slovní vyjádření přibližné. Tempo je možné jinými slovy vyjádřit jako počet čtvrťových not za minutu.

1.3 Síla tónu

Je matematicky vyjádřena výchylkou signálu, ta udává intenzitu hraného zvuku. Pro označení síly tónu se v hudbě používá subjektivní dynamická stupnice, ta však není nijak matematicky závislá na intenzitě signálu. Nejslabší a nejsilnější tón je vymezen možnostmi hudebního nástroje. Pro označení dynamiky skladby se používají italské názvy. Příklad označení s významem uvádím v Tab. 1.1:

Tab. 1.1: Názvy a označení hudební dynamiky

Název	Označení	Význam
Pianissimo piano	*ppp*	co nejslaběji
Pianissimo	*pp*	velmi slabě
piano	*pp*	slabě
mezzo-forte	*mf*	středně silně
forte	*ff*	silně
fortissimo	*ff*	velmi silně
Fortissimo forte	*fff*	co nejsilněji

1.4 Výška tónu

Výška tónu vyjádřená absolutně je definována kmitočtem daného tónu. V hudbě se frekvence základních tónů pohybují od 16 Hz do 8400 Hz [11].

Častěji se však v hudbě používá relativní výška tónu neboli interval. Ten udává poměr kmitočtů. Nejdůležitější jsou intervaly, které se dají vyjádřit poměrem celých čísel. Základním intervalem dvou tónu je interval 2:1 nazývaný oktáva. Tón o oktávu vyšší má tedy dvojnásobný kmitočet než základní tón. Spektrum hudebních signálů se tímto způsobem dá rozdělit na 10 oktáv. Jejich značení a názvy jsou uvedeny v následující tabulce 1.2.

Tab. 1.2: Názvy a značení hudebních oktáv

Název oktávy	Označení		Anglické číselné označení / MIDI
subkontra	C_2	,,C	0
kontra	C_1	,C	1
Velká	C	C	2
malá	cc	cc	3
jednočárkovaná	c^1	c'	4
dvoučárkovaná	c^2	c''	5
tříčárkovaná	c^3	c'''	6
čtyřčárkovaná	c^4	c''''	7
pětičárkovaná	c^5	c'''''	8
šestičárkovaná	c^6	c''''''	9

V jedné oktávě existuje tónová řada nazývaná stupnice. Tóny ve stupnici navzájem tvoří řadu intervalů. Nejmenší vzdálenost mezi dvěma tóny je většině zemí půltón, přičemž oktáva je rozdělena na dvanáct půltónů. Výjimku tvoří některé historické stupnice nebo východní hudba. Uspořádání vzájemných výškových poměrů jednotlivých tónů stupnice hudebního nástroje se nazývá ladění. Soustavy ladění budou popsány v další kapitole. Pro vzájemnou polohu tónů v řadě není rozhodující rozdíl kmitočtů, ale jejich podíl [11].

Pro další usnadnění výpočtů při porovnávání intervalů a různých ladění se používá logaritmická stupnice, která vznikne rozdělením intervalu temperovaného ladění (bude popsáno dále) na sto stejných dílků. Nazývá se cent a lze ho definovat následujícím vztahem:

$$1\text{cent} = \sqrt[100]{q} = \sqrt[100]{\sqrt[(12)]{2}} = \sqrt[1200]{2} \quad , \qquad (1.1)$$

kde q je velikost intervalu. Pro rovnoměrně temperované ladění je $q = \sqrt[12]{2}$.

1.5 Barva tónu

Je dána spektrálním složením tónu. Zejména zastoupením alikvotních tónů – tedy vyšších harmonických složek. Jejich intenzitou, nebo poměrem intenzit jednotlivých harmonických složek a jejich počtem [14]. Barva tónu je velmi komplexní pojem, není možné jej přímo navázat na určitou veličinu, jako je tomu u výšky (frekvence), síly (intenzita). Tyto dva parametry nepřímo také ovlivňují vnímání barvy zvuku.

Barva zvuku je velmi špatně definovatelná. Jedna z definic je, že dva různé hudební nástroje mohou vydávat stejně vysoké, hlasité a dlouhé tóny. Přesto pro posluchače nebudou znít stejně. Odlišují se tedy právě v barvě zvuku. Další definicí je Helmholtzova teorie barvy zvuku: U tónů hudebních nástrojů jsou poměry intenzit jednotlivých složek spektra na absolutní výšce těchto složek nezávislé. To je takzvaná relativní teorie týkající se hudebních nástrojů. Další je absolutní teorie týkající se vokálů: Barva zvuku je určena existencí jedné nebo dvou charakteristických harmonických složek, které jsou ve zvuku zdůrazněny. Poloha těchto složek je pevně spojena s jejich absolutní výškou. V praxi to znamená, že se stoupající výškou tónu hudebního nástroje se tvar spektra tohoto tónu nemění, zatímco u vokálů se zase nemění frekvenční poloha zdůrazněných složek (označovaných jako formanty), které se stoupající výškou vokálu postupně přecházejí na harmonické složky nižších pořadových čísel [14].

Barva zvuku se popisuje velmi subjektivně, ale obecně lze říci, že čím více jsou zastoupeny alikvotní (vyšší harmonické) tóny a čím vyšší je jejich intenzita, tím zní nástroj ostřeji. S klesajícím počtem vyšších harmonických nabývají tóny měkké zabarvení. Na zabarvení tónu mají vliv takzvané formanty, které jsou vzhledem k výšce základního tónu umístěny v určitých rezonančních oblastech. Některé formanty nebývají harmonické, tj. jejich kmitočet není celočíselným násobkem kmitočtu základního tónu. Formanty hudebních nástrojů mohou být dvou typů [11]:

- pevné formanty - poloha formantů se změnou výšky tónu nemění,
- pohyblivé formanty - poloha formantů se změnou výšky tónu mění.

1.6 Soustavy ladění

Ladění je uspořádání vzájemného výškového poměru tónů v tónové soustavě. Je možné ladění rozdělit, když pomineme historická ladění, na dvě základní skupiny:

- ladění čisté (pythagorejské, harmonické, přirozené),
- ladění temperované.

1.6.1 Čistá ladění

Jako čistá nebo také přirozená ladění se označují ladění využívající pouze tóny, jejichž frekvence jsou ve vzájemných poměrech vyjádřitelných celými čísly. V případě, že jsou tato celá čísla dostatečně malá, znějí souzvuky čistě. Tato ladění mají však nevýhody, které znemožňují jejich větší využití. Při důsledném použití čistého ladění by se v oktávě dal vytvořit nekonečný počet tónů. To samozřejmě není možné a tak při použití omezeného počtu tónů vznikají nelibozvučné intervaly. Další problémem je využití při ladění nástrojů s pevnými výškami tónů. Není možná takzvaná enharmonická záměna – takto se označuje fakt, kdy zvýšený tón (například C#) zní stejně jako snížený tón následující (tedy například D*b*). U klavíru je his totožné s c, cis s des, ces s h apod. [8]

Mezi čistá ladění patří pythagorejské ladění nazývané také jako kvintové ladění. Tento systém se tvořil na základě melodického principu ze dvou základních intervalů, kvinty a oktávy. Všechny tóny oktávy jsou získány postupnými kvintovými kroky, sedmitónová stupnice je tvořena šesti kroky – interval 3/2. Rozdělení pythagorejské stupnice je pak znázorněno v tabulce 1.3:

Tab. 1.3: Rozdělení pythagorejské stupnice

stupeň	**c**	**d**	**e**	**f**	**g**	**a**	**h**	**c**
	prima	velká sekunda	velká tercie	čistá kvarta	čistá kvinta	velká sexta	velká septima	oktáva
interval	1	$\frac{3^2}{2^3}$	$\frac{3^4}{2^6}$	$\frac{2^2}{3}$	$\frac{3}{2}$	$\frac{3^3}{2^4}$	$\frac{3^5}{2^7}$	2

V tomto ladění bývají laděny soubory smyčcových nástrojů - kvartet, sextet atd.

Nejběžnějším z čistých ladění je didymické (také nazývané terciové nebo přirozené). Soustava vznikla zavedením dalšího intervalu 5/4 tercie. Dále sexta byla nahrazena novým intervalem 5/3 vzniklým vynásobením intervalu tercie a kvarty (tzv. velká sexta). Septima byla nahrazena intervalem 15/8, což je interval tercie od kvinty (tzv. velká septima). Průběh přirozené oktávy pak ilustruje následující tabulka 1.4:

Tab. 1.4: Rozdělení pythagorejské stupnice

stupeň	**c**	**d**	**e**	**f**	**g**	**a**	**h**	**c**
	prima	velká sekunda	velká tercie	čistá kvarta	čistá kvinta	velká sexta	velká septima	oktáva
interval	1	$\frac{3^2}{2^3}$	$\frac{5}{2^2}$	$\frac{2^2}{3}$	$\frac{3}{2}$	$\frac{5}{3}$	$\frac{3\cdot 5}{2^3}$	2

Toto ladění zní absolutně čistě v tónině odvozené od základního tónu. Bohužel již ve velmi příbuzných tóninách se objevují velice disonantní – takzvané vlčí intervaly. V důsledku toho je toto ladění pro složitější hudbu prakticky nepoužitelné. Řešením, které se využívá v dnešní době nejvíce je zavedení takzvaného temperovaného ladění.

1.6.2 Temperované ladění

Pro zmírnění problémů, které se vyskytují u čistých ladění bylo vytvořeno mnoho druhů temperovaných ladění. U mnohých došlo záměrně k rozladění některých čistých intervalů (nerovnoměrně temperovaná ladění), aby se dalo dosáhnout přesnějšího naladění u jiných. Velkou předností temperovaných ladění je možnost modulace do jiných tónin bez vzniku disharmonických intervalů. U většiny temperovaných ladění existuje enharmonická záměna.

Temperovaná ladění mohou být rozdělena na rovnoměrná a nerovnoměrná. Mezi nerovnoměrně temperovaná ladění například patří: Parejovo ladění (15. století), středotónové ladění (16.-17. století), Kirnbergerovo ladění (18. Století). V současnosti nejpoužívanějším laděním, nejen v evropské hudbě, je rovnoměrně temperované ladění. Všechny intervaly stejného druhu (tercie, kvinty, kvarty) jsou stejně velké, ale žádný interval kromě oktáv není úplně „čistý“. Také všechny tóniny jsou rovnocenné, modulace je možná do libovolně vzdálených tónin bez vlivu na zvukovou kvalitu intervalů [8, 11].

Čisté soustavy ladění nemají žádný interval, jehož násobek by tvořil oktávu, k vyplnění oktávy se používají kombinace různých tónů. Temperovaná soustava tento nedostatek odstraňuje. Interval oktávy je rozdělen na 12 stejných půltónů. Pro frekvence každých dvou sousedních půltónů platí:

$$f_{n+1} = q \cdot f_n \quad , \tag{1.2}$$

kde f_{n+1} je nejbližší vyšší půltón k půltónu f_n a q je kmitočtový interval temperovaného půltónu.

Za předpokladu, že v jedné oktávě je dvanáct půltónů a jedna oktáva představuje zdvojnásobení kmitočtu jednoho tónu. Můžeme tedy odvodit vztah pro q:

$$q = \sqrt[12]{2} \quad . \tag{1.3}$$

Jako základ slouží standardní sedmitónová stupnice. Základní frekvencí, od které se odvozují všechny tóny a jejich frekvence je frekvence komorního a[1]. Je určena mezinárodní normou ISO R16 na 440 Hz s maximální povolenou odchylkou ±0,5 Hz. V následující tabulce 1.5 jsou uvedeny kmitočty jednotlivých půltónů.

Tab. 1.5: Kmitočty tónů temperovaného ladění

tón	Oktáva – označení české a anglické číselné							
	f [Hz]							
	C_2	C_1	C	c	c^1	c^2	c^3	c^4
	0	1	2	3	4	5	6	7
c	16,35	32,7	65,41	130,81	261,63	523,25	1046,5	2093
cis	17,32	34,65	69,3	138,59	277,18	554,37	1108,73	2217,46
d	18,35	36,71	73,42	146,83	293,66	587,33	1174,66	2349,32
dis	19,45	38,89	77,78	155,56	311,13	622,25	1244,51	2489,02
e	20,6	41,2	82,41	164,81	329,63	659,26	1318,51	2637,02
f	21,83	43,65	87,31	174,61	349,23	698,46	1396,91	2793,83
fis	23,12	46,25	92,5	185	369,99	739,99	1479,98	2959,96
g	24,5	49	98	196	392	783,99	1567,98	3135,96
gis	25,96	51,91	103,83	207,65	415,3	830,61	1661,22	3322,44
a	27,5	55	110	220	440	880	1760	3520
ais	29,14	58,27	116,54	233,08	466,16	932,33	1864,66	3729,31
h	30,87	61,74	123,47	246,94	493,88	987,77	1975,53	3951,07

Pro potřeby této práce bude dále použita pouze rovnoměrně temperovaná soustava ladění. Zejména kvůli snadnému a přesnému výpočtu frekvencí tónů.

1.7 Tempo

K popisu rychlosti pohybu skladby v čase se používá pojem tempo. Tempo se v notovém záznamu značí italským názvem, nebo příslušným číslem. Při vyjádření konkretním číslem toto číslo představuje počet čtvrťových not za minutu. Při zápisu tempa číslem toto číslo reprezentuje BPM – to jest počet úderů za minutu, je to počet tiků pomyslného metronomu podle kterého by se hudebník řídil. Při slovním zápisu se pak nejedná o přesné určení tempa, ale jeho přibližného rozsahu. Základem je střední tempo, které se pohybuje mezi 60-90 bpm. Tato hodnota odpovídá přibližně rychlosti lidského tepu. Pro účely této práce je však důležité přesné určení tempa pomocí údaje v bpm.

2 MIDI

Musical Instrument Digital Interface, je rozhraní určené pro digitální nástroje. Přesněji se jedná o komunikační protokol navržený pro komunikaci mezi hudebními nástroji, počítači a dalšími zařízeními v reálném čase prostřednictvím sériového rozhraní. Tento standard spravuje MMA - MIDI Manufacturers Associatoin a JMSC - Japanese Midi Standard Committee.

Součástí tohoto standardu je jak definice způsobu přenosu dat mezi zařízeními, tak způsob uložení dat v tzv. Standardním MIDI souboru – Standard MIDI file (zkráceně SMF).

2.1 Historie

Za počátek standardu MIDI se považuje rok 1981, kdy na výstavě NAMM (National Association of Music Merchants) v USA vznikly první návrhy na univerzální rozhraní určené pro hudební nástroje. Tento návrh nejprve projednávali zástupci amerických firem SCI, OBERHEIM a japonské firmy ROLAND. S průběhem času se do projektu zapojili také firmy YAMAHA, KORG a KAWAI. Tyto firmy pak představili první ucelený návrh pod názvem USI (Universal Syntethize Interface). O tento systém však nebyl velký zájem.

Během dalšího roku byla k tomuto systému dodána řada vylepšení a připojeny další návrhy. Nový koncept firma ROLAND nazvala MIDI - Musical Instruments Digital Interface. V říjnu roku 1983 byla výrobcům představena první verze tohoto standardu MIDI 1.0. Ani tento standard však nebyl dokonalý a mezi výrobci docházelo k problémům s kompatibilitou. Aby se předcházelo dalším problémům vznikly o rok později organizace které měli za úkol nést zodpovědnost za dodržování normy MIDI a podílet se na dalším vývoji. Těmi byli MMA (MIDI Manufacturers Associatoin) a JMSC (Japanese Midi Standard Committee) [7].

V září roku 1985 byla předvedena podrobná MIDI norma. Ta řešila problémy s kompatibilitou. S dalším vývojem zařízení si tato specifikace vyžádala doplňky a dodatky, které pokrývali nejrůznější oblasti. Postupně se zvyšovala rychlost komunikace na rozhraní mezi zařízeními, měnil se způsob časování na tomto rozhraní. S rozvojem počítačů se v roce 1988 přidal také standard pro ukládané MIDI dat, takzvaný SMF (Standard MIDI File). V roce 1991 byl zaveden standard General MIDI (GM). Ten na novější výrobky klade vyšší nároky na zařízení, například zavádí minimální sadu nástrojů, které musí nástroj podporovat, nebo také musí podporovat být schopny hrát nejméně 24 tónů zároveň (tzv. polyfonie).

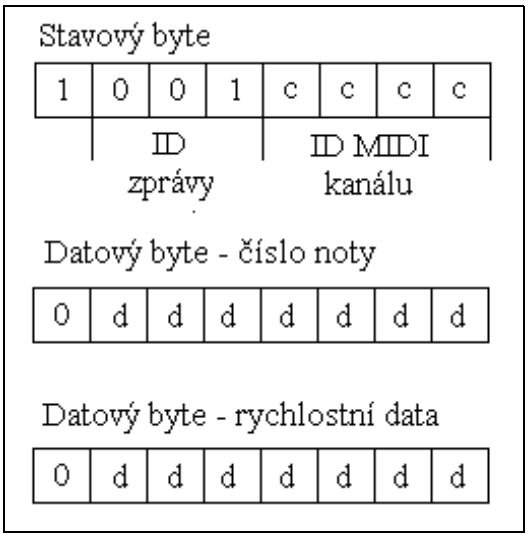
V současnosti je standard MIDI také stále vyvíjen, kvůli stále vyšším nárokům na množství přenesených informací. MIDI zaznamenalo rozvoj také po stránce dodatků týkajících se ukládání souborů. Tento rozvoj se udál zejména se rozšířením mobilních telefonů s podporou MIDI jako vyzváněcích tónů, zpočátku jednohlasé později i vícehlasé.

Zároveň se začínají na trhu objevovat vysokorychlostní přenosová rozhraní určená pro hudební nástroje. Příkladem může být projekt mLAN (Music Local Area Network) firmy YAMAHA, který je založen na standardu IEEE1394 známým jako FireWire [16].

2.2 Princip MIDI

Protokol MIDI nepracuje se zvukem jako s klasickým zvukovým signálem, ale využívá komunikace prostřednictvím zpráv s různými parametry zvukového signálu. Těmi jsou identifikace nástrojů, not, doby trvání a podobně. Koncové zařízení tyto informace zpracuje na jejich základě vytvoří konkrétní audiosignál. Zpráva může mít jeden a více bajtů přičemž první bajt je vždy stavový a za ním mohou být bajty datové. Například zpráva reprezentující stisk klávesy syntetizátoru obsahuje tři bajty. První bajt je stavový a nese informaci, že byla stisknuta klávesa (Note On). Druhý je datový a udává číslo noty a třetí datový bajt určuje s jakou razancí byla klávesa stisknuta [4, 5].

Nejdůležitějšími zprávami jsou tedy Note On a Note Off. Obě mají shodný počet datových bytů a to 2. Struktura zprávy Note On je znázorněna na obrázku 2.1.



Obr. 2.1: Znázornění bytů zprávy Note On

První bit zprávy je identifikátor stavového bytu, další tři bity identifikují typ zprávy, v tomto případě Note On, poslední čtyři bity jsou vyhrazeny pro identifikaci virtuálního MIDI kanálu. V prvním datovém bytu je první bit identifikátor typu zprávy (datový byte) a dalších sedm bitů určuje číslo noty v rozsahu 0-127. Čísla not (decimálně) jsou uvedeny v tabulce 2.2. V posledním datovém bytu je sedmibitové vyjádření dynamiky [10].

Zpráva Note Off má stejnou strukturu, která je znázorněna na obrázku 2.1. Jediný rozdíl je v ID zprávy, což jsou tři bity za MSB stavového bytu. Ten se v případě zprávy Note Off změní na 000.

2.3 Noty a jejich interpretace

Nota je v MIDI zprávě zakódována číslem. Toto číslo jednoznačně určuje oktávu kam nota patří a její frekvenci. Frekvenci lze z čísla noty vypočítat podle následujícího vztahu [10]:

$$f = 440 \cdot 2^{\frac{n-69}{12}} \quad , \tag{2.1}$$

kde n je číslo noty a f je frekvence v Hz.

Po úpravě lze snadno získat vztah pro odvození čísla noty při známe frekvenci:

$$n = (12 \cdot \log_2(\frac{f}{440}) + 69) \quad . \tag{2.2}$$

Číslo noty je v rozsahu 0-127, což je dáno tím, že číslo noty je kódováno sedmi bity. V tabulce 2.2 jsou přehledně vyznačeny názvy not s jejich číselným označením.

Tab. 2.2: Noty a jejich číselné vyjádření dle MIDI standardu

Číslo, označení oktávy		**Čísla not v MIDI zápisu, frekvence**											
		C	C#	D	D#	E	F	F#	G	G#	A	A#	H
-1	C_3	0	1	2	3	4	5	6	7	8	9	10	11
0	C_2	12	13	14	15	16	17	18	19	20	21	22	23
1	C_1	24	25	26	27	28	29	30	31	32	33	34	35
2	C	36	37	38	39	40	41	42	43	44	45	46	47
3	c	48	49	50	51	52	53	54	55	56	57	58	59
4	c^1	0	61	62	63	64	65	66	67	68	69	70	71
5	c^2	72	73	74	75	76	77	78	79	80	81	82	83
6	c^3	84	85	86	87	88	89	90	91	92	93	94	95
7	c^4	96	97	98	99	100	101	102	103	104	105	106	107
8	c^5	0	109	110	111	112	113	114	115	116	117	118	119
9	c^6	120	121	122	123	124	125	126	127				

2.4 Zápis melodie do SMF

SMF – Standard MIDI File je standardizovaný formát pro zápis MIDI dat do souboru. Do MIDI normy byl oficiálně začleněn v roce 1988 a následně i do General MIDI standardu. Struktura zápisu dat do souboru umožňuje kódování hudebních i jiných dat způsobem, který umožňuje následný přenos dat mezi různými přístroji.

Existují různé způsoby zápisu dat do souboru. Podle počtu zapisovaných stop se rozděluje SMF na 3 formáty:

- **Formát 0** – zapisuje se pouze jedna stopa (*Track*), která může obsahovat všechny MIDI kanály.
- **Formát 1** – ukládá jednu i více současně hraných stop (*Pattern*). Může obsahovat i multikanálové stopy (tzv. nečistý formát 1).
- **Formát 2** – uloží celou skladbu (*Song*). Je určen pro patternově orientované sekvencery – každý pattern obsahuje jednu multikanálovou stopu.

Celý soubor je pak možné uložit v binární nebo Hex ASCII podobě. Po převodu na sedmibitový formát lze soubor přímo přenášet po MIDI rozhraní. Výsledný soubor se skládá z hlavičky SMF následované hlavičkou stopy. Po těchto hlavičkách jsou zapsána samotná data [4].

Hlavička SMF:

Tab. 2.3: Popis hlavičky SMF

Formát(hex):	[4D 54 68 64]	[ll ll ll ll]	[ff ff]	[nn nn]	[rr rr]
Význam pole:	„M T h d“ Označení hlavičky v ASCII kódu	Délka hlavičky v bajtech	Formát SMF (0,1,2)	Počet stop	Časové rozlišení
Délka (bit):	32	32	16	16	16

Hlavička SMF musí být uvedena na úplném začátku souboru a je přítomna vždy, délka hlavičky je standardně 6 bytů. Počet stop se odvíjí od zvoleného formátu SMF (popsán výše), pro formát 0 musí být pouze jedna stopa. Časové rozlišení je velmi důležitý údaj od kterého se odvíjí jakékoliv značení času v celém souboru. Může být uvedeno ve dvou formátech. Prvním je relativní popis času jako počet tiků ve čtvrťové notě, v tomto případě je první bit nastaven na hodnotu 0. Druhým způsobem je popis ve formátu SMPTE/EBU (SMPTE/AES3). Tento formát byl vyvinut jako univerzální formát záznamu času pro zvukové záznamy, přenositelný mezi různými zařízeními. Tento formát je zvolen, pokud první bit (MSB) je nastaven na hodnotu 1. Další bity pak reprezentují počet tiků na okénko. Okénko může být různě dlouhé, nejčastěji však používá 25 nebo 30 okének za vteřinu.

Hlavička stopy:

Tab. 2.4: Popis hlavičky stopy

Formát(hex):	[4D 54 72 6B]	[ll ll ll ll]	<mm>	<dd>
Význam pole:	„M T r k“ Označení hlavičky v ASCII kódu	Délka stopy v bajtech	Označení *meta-eventu*	Hudební data stopy
Délka (bit):	32	32	proměnná	proměnná

Součástí hlavičky stopy jsou takzvané *meta-eventy,* které nesou různé informace jako je tempo, metrum skladby, textové popisy, písňové texty v čase synchronizované s hudbou (např. pro karaoke). V případě, že chybí *meta-eventy* tempo a metrum, jsou použity standardní hodnoty pro tempo - 120bpm a metrum - 4/4.

Každý *meta-event* má svoji strukturu podobnou hlavičkám. Rozdílem je, že *meta-event* začíná hodnotou $FF. Struktura je popsána v tabulce 2.5:

Tab. 2.5: Struktura *meta-eventu*

Formát(hex):	[FF]	[tt]	<ll>	<dd>
Význam pole:	Konstantní hodnota označující, že se jedná o *meta-event*	Identifikátor typu *meta-eventu*	Délka *meta-eventu*	Data *meta-eventu*
Délka (bit):	8	8	proměnná	proměnná

Identifikátor typu rozlišuje jednotlivé *meta-eventy,* v MSB tohoto bytu má nastaven na hodnotu 0, je tedy možné rozlišit 128 různých eventů.

Identifikátor délky má obecně různou velikost, nejčastěji však 8 nebo 16 bitů. Délka eventu označuje délku dat v konkretní události. Může označovat například délku textu ve jméně skladby nebo stopy.

Jako příklad uvádím několik nejdůležitějších *meta-eventů*:

- Tempo: $FF $51 $03 [tt tt tt]

  V tomto eventu definuje 24bitový parametr *t* tempo skladby, které je definováno jako „čas za jednu dobu“. Udává se v mikrosekundách za čtvrťovou notu.

- Konec stopy: $FF $2F $00

  Velmi důležitá značka, která určuje konec aktuální stopy. Velký význam má zejména když jedna stopa běží ve smyčce.

- Metrum: $FF $04 [nn] [dd] [cc] [bb]

  Metrum je definováno pomocí 4 hodnot *n* udává čitatele, *d* jmenovatele metrického údaje. Ve jmenovateli reprezentuje hodnota 0 celou notu, 1 půlovou, 2 čtvrťovou notu atd. Hodnota *c* značí počet MIDI synchrotiků v jednom úderu metronomu. Hodnota *b* vyjadřuje počet dvaatřicetinových not v jedné čtvrťové.

Poslední částí jsou samotná data. Data se opět skládají z různých po sobě jdoucích událostí. Před každou událostí je zapsána informace o jejím relativním časovém umístění, takzvaný deltačas. Tato informace definuje vzdálenost od předcházející události a je uvedena jako počet *tiků*. Po této informaci následuje stavový byte, který definuje o kterou událost jde a číslo příslušného kanálu. Stavový byte je také možné uvést pouze na začátku série stejných událostí a dále uvádět pouze časové značky a „datovou“ část eventu (například číslo noty a rychlost stisknutí), je tak možné zmenšit velikost záznamu. Jedná se o takzvaný *trvající stavový byte*. Jeho průběh lze přerušit záznamem dalšího *meta-eventu*. Podle typu události se odvíjí význam dalších dvou bytů. Například pro událost Note On následuje identifikace noty a rychlost zapnutí (popsáno v části 2.2).

Deltačas může být obecně různě dlouhý, od jeho délky se odvíjí počet bytů jež ho popisují. Pro zápis do souboru se využívá kódování, kdy se převádí osmibitový zápis skupin na sedmibitové skupiny. Do MS bitu se ve všech bytech, kromě posledního, doplní 1. Pro demonstraci uvádím ukázku přepočtu hodnoty 16383 tiků:

Tab. 2.6: Kódování údaje deltačasu

Původní čas (Hex)	3	F	F	F
Binárně (Bin)	0 0 1 1	1 1 1 1	1 1 1 1	1 1 1 1
7bit skupiny (Bin)	~~0 0~~	1 1 1 1 1 1 1	1 1 1 1 1 1 1	
Úprava MSB	**1** 1 1 1 1 1 1 1		**0** 1 1 1 1 1 1 1	
Nibblizace	1 1 1 1	1 1 1 1	0 1 1 1	1 1 1 1
Zakódovaný deltačas (Hex)	F	F	7	F

3 Metody detekce výšky tónu

V této kapitole popisuji zvolené metody, jejich vlastnosti, výhody a nevýhody při použití pro detekci základního kmitočtu.

Všechny metody nevyužívají na svém vstupu přímo celý zvukový signál, ale jeho segmenty. Délka segmentu se odvíjí od zvoleného tempa skladby a nejkratší zvolené noty. Segmenty pak svojí délkou přímo reprezentují dobu trvání jedné noty.

Aby při dalším zpracování mohla být použita FFT, je třeba aby délka segmentu byla 2^N. Toho může být dosaženo přidáním potřebného počtu nulových vzorků k segmentu.

Metody se dají rozdělit na skupiny podle oblasti, kterou využívají:

1. Detekce v časové oblasti – autokorelační metoda
2. Detekce ve spektrální oblasti – jednoduché prahování ve spektrální oblasti
3. Detekce v kepstrální oblasti

U řečových signálů se jako první krok zpracování používá frekvenční omezení signálu pouze na kmitočtové pásmo v němž se vyskytuje řečový signál. Tento krok je však u hudebních signálů nemožný, protože hudební signál zpravidla pokrývá celé spektrum signálu. Při detekci klasických nástrojů by bylo možné omezit maximální frekvenci signálu na 8400 Hz, protože to je maximální základní kmitočet používaný u varhan. Vyšší se ani u jiných nástrojů nepoužívají. Toto však nemusí platit u elektronické hudby.

3.1 Segmentace signálu

Prvním krokem všech metod je segmentace vstupního signálu. V Matlabu jsem implementoval algoritmus pro segmentaci. Jeho vstupní proměnná je vektor a výstupní proměnná je matice, kde počet řádků reprezentuje délku segmentu a počet sloupců odpovídá počtu segmentů.

Prvním krokem je stanovit délku segmentu. Ta se vypočte z tempa signálu a nejmenší délky noty. Tyto hodnoty si volí uživatel. V závislosti na zvolených hodnotách pak dochází k segmentaci. Vytvářené segmenty odpovídají svoji délkou délce zvolené nejkratší noty. Pro výpočet délky noty je možné použít vztahu:

$$t_n = \frac{4 \cdot n \cdot 60}{tempo} \quad , \tag{3.1}$$

kde t_n je délka trvání zvolené nejkratší noty *n, n* je nota (například celá - 1, šestnáctinová - 1/16) a *tempo* je zvolené tempo v bpm.

Při známé délce noty již není problém vypočítat podle následujícího vztahu délku segmentu ve vzorcích.

$$n_s = t_n \cdot f_{vz} \quad , \tag{3.2}$$

kde n_s je počet vzorků segmentu, t_n délka noty v sekundách, f_{vz} je vzorkovací kmitočet signálu.

Známe-li délku segmentu ve vzorcích a délku vstupního signálu, můžeme vypočítat počet segmentů, na které signál rozčleníme. Počet segmentů se dá obecně vypočítat podle vztahu:

$$N_s = \frac{1 + (l - l_s)}{l_s - p} \quad , \tag{3.3}$$

kde N_s je počet segmentů, l je délka vstupního signálu ve vzorcích, l_s je délka segmentu ve vzorcích a p je možné překrytí ve vzorcích. V našem případě při segmentaci uvažujeme pravoúhlé okno a žádné překrytí segmentů. Vycházím z předpokladu, že se segmenty budou v čase překrývat s hranými notami. Změna noty tak bude korespondovat se změnou segmentu. Kdybychom v tomto případě uvažovali překrytí, vnášela by se do části segmentu nežádoucí frekvence. Její vliv by se dal potlačit použitím jiných typů oken (např. Hammingovo). V této práci budeme tedy dále uvažovat nulové překrytí segmentů a obdélníkové okno. Poté se jednotlivé úseky skládají do sloupců a v případě, že poslední úsek bude mít délku kratší než je délka segmentu, je doplněn nulami.

Pro ilustraci uvádím tabulku (viz Tab. 3.1) s vybranými hodnotami minimální délky noty, vzorkovacími frekvencemi a odpovídajícími délkami segmentu ve vzorcích.

Tab. 3.1: Délky segmentů v závislosti na tempu, vzorkovací frekvenci a délce noty

vzorkovací frekvence	tempo	nejkratší délka noty	délka segmentu
Hz	bpm	-	vzorky
32000	90	1/32	2666,67
	120		2000
	160		1500
32000	90	1/64	1333,33
	120		1000
	160		750
44100	90	1/32	3675
	120		2756,25
	160		2067,19
44100	90	1/64	1837,5
	120		1378,13
	160		1033,59
48000	90	1/32	4000
	120		3000
	160		2250
48000	90	1/64	2000
	120		1500
	160		1125

Abychom při dalším zpracování měli dostatečné frekvenční rozlišení, pro detekci nízkých kmitočtů, je třeba dostatečně dlouhý segment. Pro frekvenční rozlišení platí vztah [6]:

$$\Delta f = \frac{f_{\mathrm{VZ}}}{N} \quad , \tag{3.4}$$

kde f_{VZ} je vzorkovací kmitočet v Hz a N je délka segmentu ve vzorcích. Frekvenční rozlišení zároveň udává nejnižší možný detekovatelný kmitočet. Pro testovací signály použité v této práci, které mají f_{VZ} = 48000 Hz, při použití nejkratší délky noty 1/32, je tedy při standardním tempu 120 úderů za minutu nejnižší detekovatelný kmitočet Δf = 16 Hz. V testovacích signálech se vyskytují pouze tóny se základní frekvencí vyšší než 65Hz proto je tato hodnota dostatečná. Z pohledu frekvenčního rozlišení však tato hodnota dostatečná není, protože půltóny se při nízkých frekvencích od sebe liší řádově v jednotkách Hz. Proto je třeba použít další korekční mechanizmy, jako je třeba využití průměrování při detekci v časové oblasti, jak je popsáno dále.

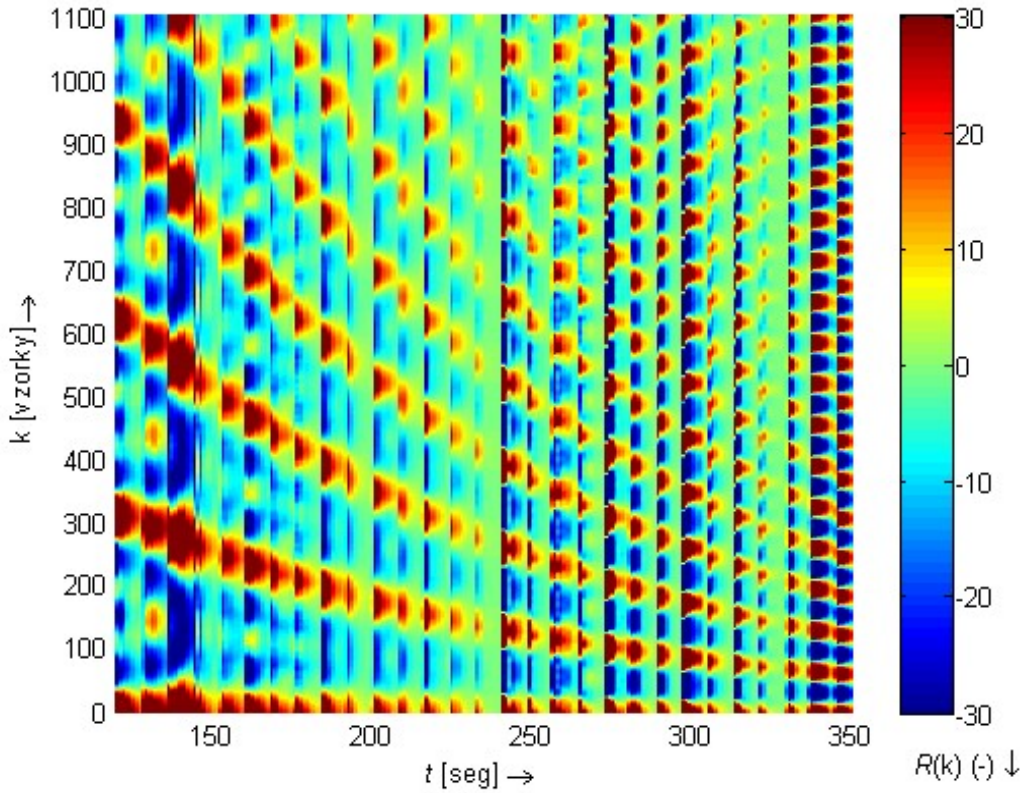
3.2 Detekce v časové oblasti

Pro detekci v časové oblasti se využívá takzvané autokorelace, ta vyjadřuje míru podobnosti jediné vzájemně posunuté posloupnosti. Jednostranná autokorelace je definována [13]:

$$R(k) = \frac{1}{N} \sum_{i=0}^{N-k-1} s(i)s(i+k) \quad , \qquad k = 0,1,2,\ldots,N-1 \tag{3.5}$$

kde $R(k)$ je hodnota autokorelační funkce, $s(i)$ je hodnota vzorku signálu, N je délka signálu.

Každý segment signálu je podroben autokorelaci. Vynesením hodnot autokorelačních funkcí v čase do grafu vzniká korelogram, kde na vodorovné ose jsou zobrazeny jednotlivé segmenty tak jak následují po sobě v čase a na svislé ose jsou vyneseny průběhy autokorelačních funkcí jednotlivých segmentů signálu. Barvou jsou vyjádřeny hodnoty autokorelační funkce.



Obr. 3.1: Ukázka korelogramu

Na obrázku 3.1 je zachycen výřez korelogramu testovacího signálu piana s rostoucím kmitočtem v čase. V jednotlivých segmentech jsou v průběhu autokorelační funkce dobře patrné špičkové hodnoty. Poloha těchto maximálních hodnot přímo souvisí s hledanou základní frekvencí tónu. A to podle vztahu [1]:

$$L = \frac{T_0}{T_{vz}} = \frac{f_{vz}}{f_0} \quad , \tag{3.6}$$

L - je index špičky nebo vzdálenost jednotlivých špiček autokorelační funkce, také označuje periodu základního tónu ve vzorcích,

T_{VZ} - je vzorkovací perioda, T_0 je perioda základního tónu,

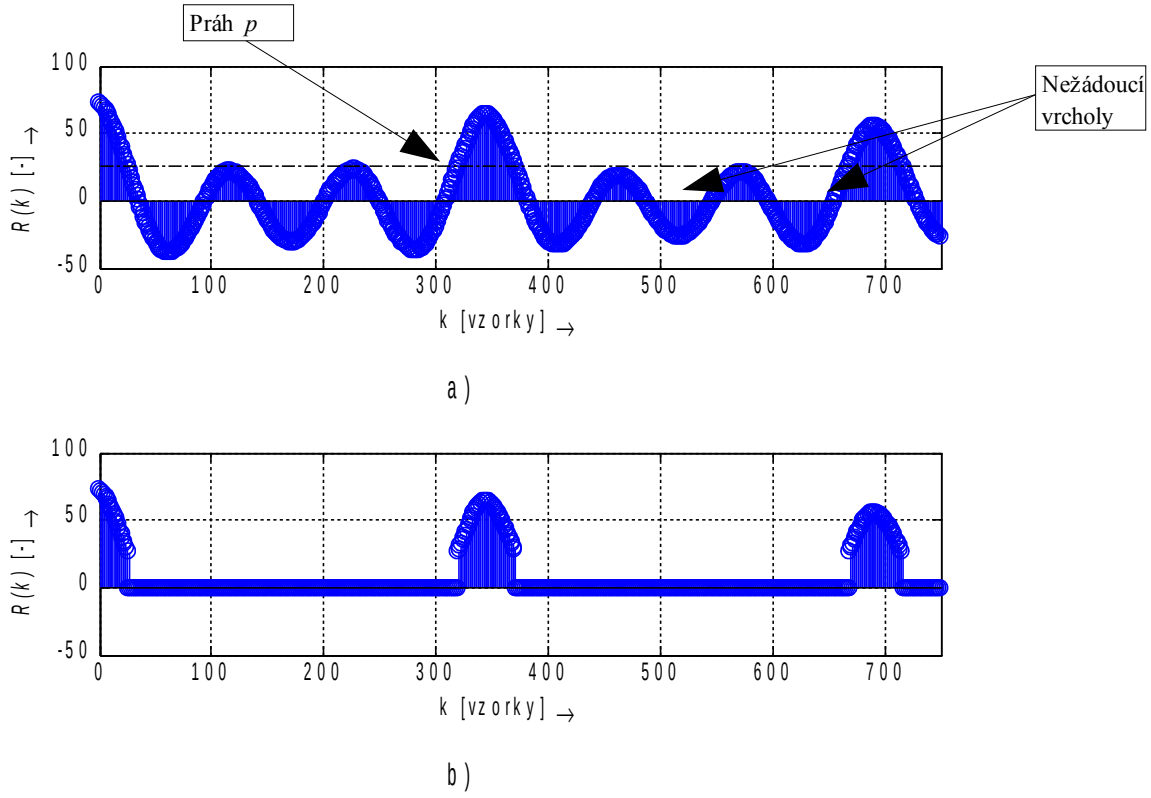
f_{VZ} - je vzorkovací frekvence, f_0 je frekvence základního tónu.

Implementoval jsem tuto metodu v Matlabu. Signál je po segmentaci načítán po sloupcích, v rámci každého segmentu je provedena autokorelace pomocí funkce `xcorr`. Po provedení autokorelace bylo nutné pro další zpracování použít pouze polovinu vzniklého korelogramu, protože funkce `xcorr` provádí cyklickou korelaci pro k = -N+1, .. , 0, .. , N-1. Pro naše potřeby postačí pouze jednostranná autokorelace.

Po vytvoření korelogramu se provádí detekce jednotlivých vrcholů a jejich vzdáleností. Jak je patrné z obrázku 3.2 a), v autokorelační funkci se objevují vrcholky, které je třeba odfiltrovat, přičemž je třeba zachovat významné vrcholy. To je možné vyřešit pomocí prahování, kdy hodnoty pod určitým prahem p jsou nulovány. Hodnota tohoto prahu bývá nastavitelná a je vyjádřena jako procentuální hodnota nejvyšší hodnoty autokorelační funkce v segmentu.

$$p_r = \frac{p}{100} \cdot max(s) \quad , \tag{3.7}$$

kde p_r je hodnota prahu, p je procentuální vyjádření hodnoty prahu, $max(s)$ je maximální hodnota v segmentu.

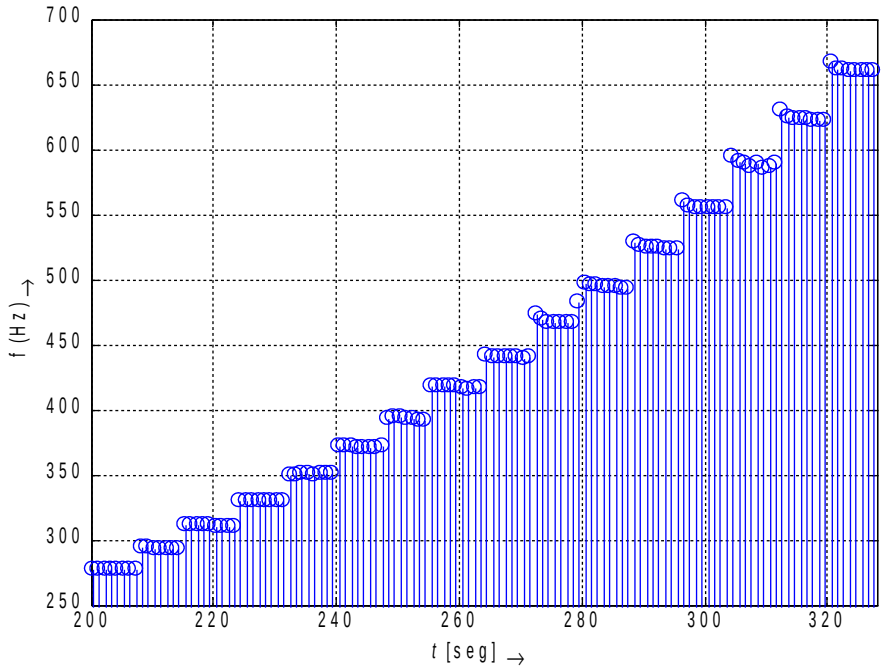


Obr. 3.2: Porovnání výřezů průběhu autokorelační funkce

Na obrázku 3.2 a) je průběh autokorelační funkce testovacího signálu piana s vyznačeným prahem $p = 35\%$. Dále jsou vyznačeny vrcholy které reprezentují vyšší harmonické složky. Na obrázku 3.2 b) je průběh téže funkce po prahování, nežádoucí vrcholy byly odstraněny.

V další fázi dochází k detekci jednotlivých lokálních maxim. Signál je nejprve rozdělen na části a to tak, že se detekují průchody nulovou hodnotou. V těchto částech jsou pak vyhledány maximální hodnoty.

Pokud bychom při výpočtu základní frekvence dle vzorce (3.6) uvažovali pouze vzdálenost prvního detekovaného vrcholu od počátku, měli bychom při detekci poměrně malé frekvenční rozlišení. Tento jev je možno omezit při použití průměrování vzdáleností volitelného počtu vrcholů. Pokud není tento počet vrcholů k dispozici (detekce nízkých frekvencí), průměrují se vzdálenosti všech vrcholů. Tím je dosaženo na vyšších frekvencích přijatelného rozlišení. Na výstupu této metody je tedy spektrogram, kde se vyskytují pouze detekované frekvence. Na obrázku 3.3 je výřez výstupu korelační metody pro signál piana. Jak je vidět, detekce obsahuje velmi málo chybně určených hodnot. Při změně frekvence se v prvních dvou až třech segmentech vyskytuje mírně odlišná hodnota, to je dáno nepřesným segmentováním signálu, kdy začátek segmentu přesně nekoresponduje s okamžikem změny kmitočtu. Změna frekvence tak proběhne uvnitř segmentu a může způsobit zkreslení. Další možnou příčinou je přechodový děj na začátku tónu.



Obr. 3.3: Detekované frekvence korelační metodou

Tato metoda přináší dobré výsledky, její realizace je však poměrně časově náročná, vzhledem k velké náročnosti výpočtu korelace. Výpočet lze urychlit využitím takzvané rychlé korelace, která využívá rychlé Fourierovy transformace. Segment je převeden do frekvenční oblasti, kde je modul hodnot umocněn na druhou a výsledek je následně převeden zpět do časové oblasti. Je využit takzvaný Wienerův-Chinčinův (autokorelační) teorém [6]:

$$r_{xx}(\tau) = x(n) * x(-n) \overset{DTFT}{\rightarrow} S_{xx}(\omega) = X(\omega) \cdot X^{*}(\omega) = |X(\omega)|^2 \quad , \qquad (3.8)$$

kde $S_{xx}(\omega)$ je výkonové spektrum signálu, které je Fourierovým obrazem autokorelační posloupnosti $r_{xx}(\omega)$.

Pro správnou detekci frekvence je klíčová fáze detekce špiček v autokorelační funkci. Bez použití prahování je detekce velmi nepřesná, kvůli vyšším harmonickým frekvencím v signálu. Pokud algoritmus prahování využijeme situace se velmi zlepší, avšak je problematická volba hodnoty prahu. Pro tóny hrané na kytaru a na piano se osvědčila hodnota prahu 30% z maximální hodnoty autokorelační funkce v segmentu. Jako univerzální hodnota, použitelná i pro jiné nástroje se osvědčila hodnota prahu 50%. Obecně je však tento parametr nutno přizpůsobit konkretní aplikaci.

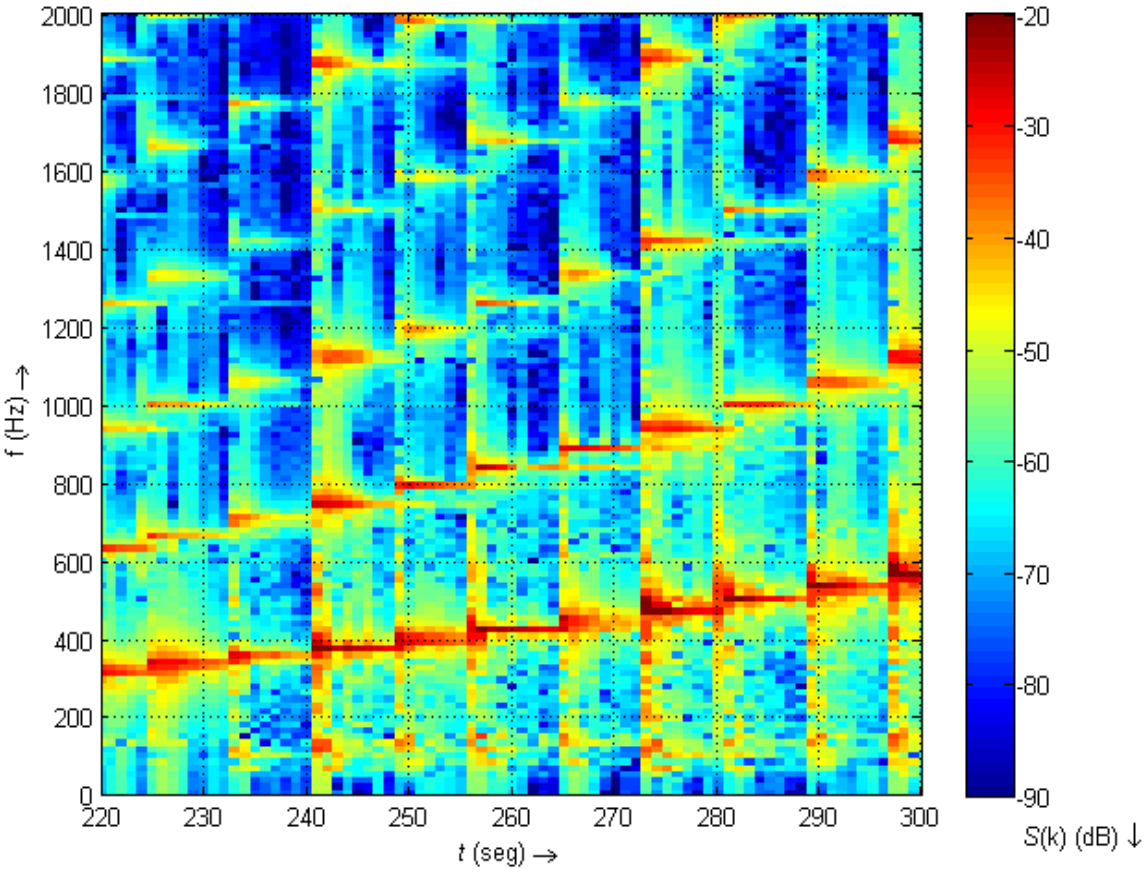
3.3 Detekce ve spektrální oblasti

Tato metoda využívá diskrétní Fourierovy transformace, která je definována [13]:

$$S(k)=\sum_{n=0}^{N-1} s(n)\cdot e^{-jk\frac{2\pi}{N}n} \quad , \qquad k=0,1,\ldots,N-1 \quad , \tag{3.9}$$

kde $S(k)$ je hodnota k. spektrální složky, s(n) je hodnota vzorku vstupního signálu a N délka signálu ve vzorcích.

Po segmentaci signálu je každý úsek podroben DFT. Vzniká tak spektrogram.



Obr. 3.4: Spektrogram melodie - výřez

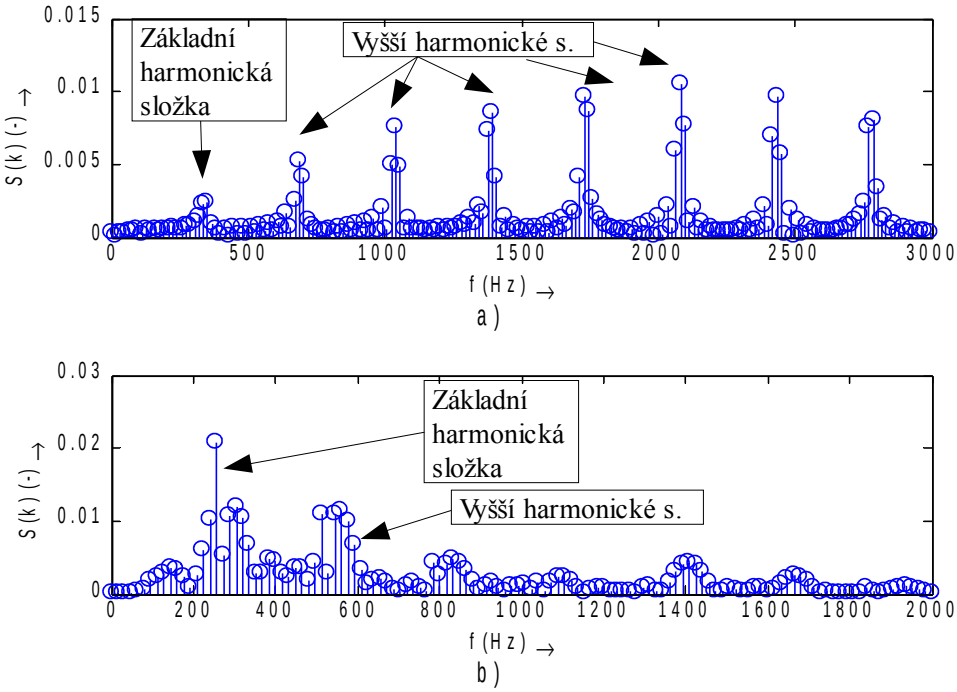
Ve výřezu spektrogramu signálu piana (Obr. 3.4) jsou dobře viditelné významné frekvenční složky. Problém této metody spočívá v rozlišení základního kmitočtu od vyšších harmonických složek - alikvotních tónů.

V případě některých hudebních nástrojů má základní harmonická složka ve spektru signálu nejvyšší hodnotu. To však nejde říci obecně o všech nástrojích. Například u tónu flétny a podobných dechových nástrojů má základní harmonická složka nejvyšší hodnotu, ale v případě strunných nástrojů, jako je například kytara, toto neplatí [2].

Bez znalosti hudebního nástroje tedy metoda detekce maximální složky ve spektru často selhává. Pro zpřesnění by bylo možné využít prahování ve spektru, kdy by se po prahování zachovaly významné špičky a první z nich (i když ne nutně nejvyšší) by se označila jako základní harmonická. Další možnost vychází z předpokladu, že nejvyšší hodnotu má druhá nebo třetí harmonická složka. Pak by bylo možné detekovat maximální hodnotu a ověřovat, zda se na polovičním (třetinovém) kmitočtu nenachází významnější špičková hodnota. Ta by pak byla označena jako základní harmonická složka.

Při znalost hudebního nástroje by se způsob detekce dal přizpůsobit typickému spektrálnímu složení tónu nástroje a výsledky by se velmi zlepšily.

Na obrázku 3.5 je ukázka porovnání spektra kytarového signálu a signálu piana s vyznačením základní a vyšších harmonických složek. Výška spektrálních složek je kvůli přehlednosti vyjádřena v absolutní míře. Na obrázku 3.5 a) je kytarový signál, je zřejmé, že základní harmonická složka nemá maximální hodnotu. Naproti tomu na obrázku 3.5 b) v této konkretní části spektra signálu piana má nejvyšší hodnotu základní harmonická a další vyšší harmonické se tedy mohou odfiltrovat prahováním, nebo pouze detekovat maximální hodnotu.



Obr. 3.5: Porovnání spekter kytary a piana

Ve skriptu pro Matlab jsem implementoval i tuto metodu. Prvním krokem bylo vytvoření spektrogramu. Spektrum je následně omezeno pouze na frekvence, kde je možné očekávat výskyt první harmonické složky (0 – 8400 Hz). Další krok je prahování spektrogramu. V prahovaném spektrogramu následuje vyhledání první harmonické složky – tento problém jsem řešil tak, že spektrum každého segmentu jsem seřadil podle velikosti a z původních indexů prvních několika hodnot jsem vybral nejnižší. Je nutná kontrola, zda hodnota na tomto indexu není nulová, ale je to opravdu špička ve spektru. Tato špička je následně považována za první harmonickou složku a v původním spektru je okolo této pozice ponechán určitý počet hodnot, zbylé jsou vynulovány. Na takto ořezaný spektrogram je aplikován dvourozměrný mediánový filtr, který má za úkol vyhladit jeho průběh. Posledním krokem je v každém segmentu vyhledat maximální hodnotu a z její pozice vypočítat odpovídající frekvenci ve spektru.

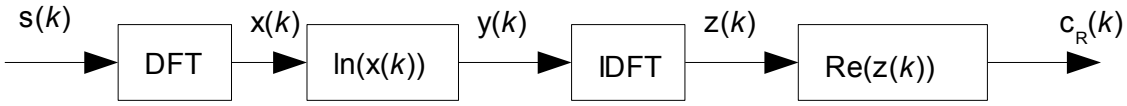
Výpočetní náročnost je přijatelná díky dobré implementaci Fourierovy transformace. To ovšem platí pouze při použití FFT, kdy je podmínkou aby délka zpracovávaného segmentu byla 2^N. Vyhledání maximální hodnoty je velmi rychlé a výpočetně nenáročné. Největší problém této metody spočívá v identifikaci správné spektrální složky.

3.4 Kepstrální metoda

Tato metoda využívá takzvaného reálného kepstra, to je definováno jako reálná část inverzní Fourierovy transformace z přirozeného logaritmu modulu Fourierovy transformace vstupního signálu [1, 13]

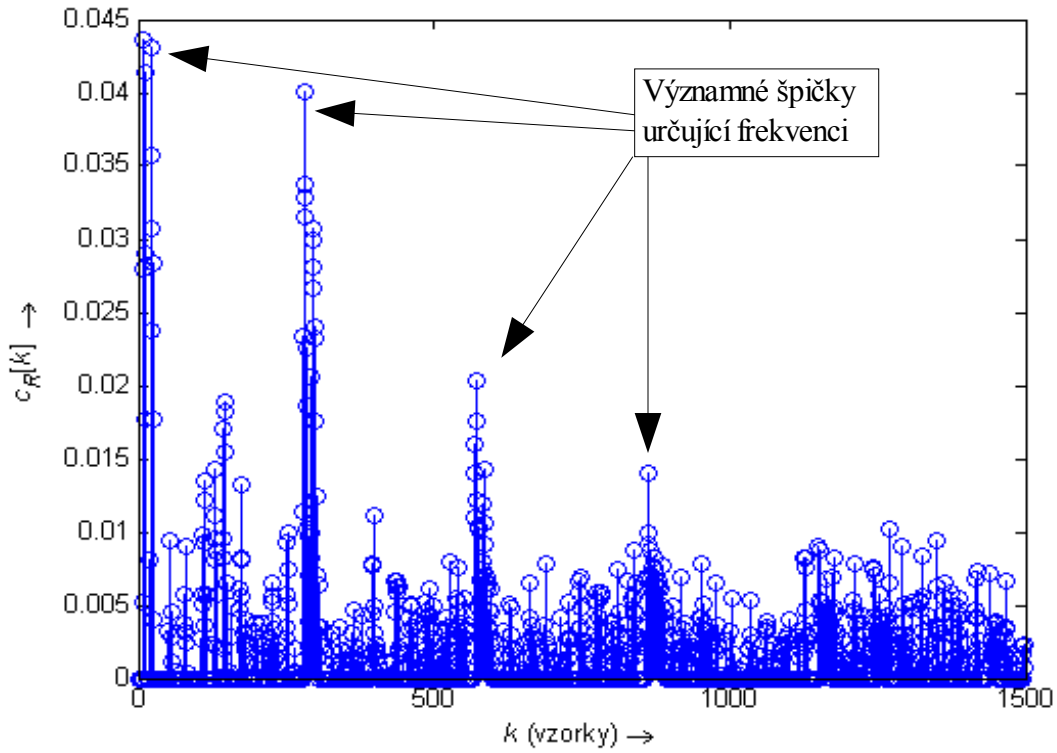
$$c_R[k] = \Re\{IFFT(\ln|FFT(s[k])|)\} \quad . \tag{3.10}$$

Při realizaci se provede FFT a získá se spektrogram, modul spektra se zlogaritmuje přirozeným logaritmem, následně se provede IFFT. Reálná část pak tvoří „kepstrogram“. Blokové schema zpracování je uvedeno níže.



Obr. 3.6: Blokové schéma výpočtu kepstra

Složky kepstrogramu se pak analyzují podobně jako v případě korelogramu. Jak je vyznačeno na obrázku 3.7, kde je zachycen detail kepstra signálu piana, v každém segmentu se najdou špičky a z jejich vzdáleností se pak vypočítá frekvence podle vztahu (3.6). Rozdílem oproti korelační metodě je fakt, že první složky kepstra neurčují frekvenci a musí být tedy při hledání lokálních špiček vynechány. Tyto složky na prvních indexech se využívají například při analýze řeči a nesou informace o parametrech hlasového ústrojí.



Obr. 3.7: Detail kepstra jednoho segmentu signálu s vyznačenými špičkami

Při implementaci jsem postupoval výše uvedeným způsobem, kdy jsem vytvořil ze segmentů kepstrum podle vztahu (3.10). Ve vytvořeném kepstrogramu jsem vyzkoušel použití mediánové filtrace, s tímto krokem se výsledky detekce částečně zlepšily. Jako další krok je prahování, ve kterém je zahrnuto vynulování několika prvních vzorků. Po prahování následuje hledání špičky v kepstru. Oproti charakteru spektra signálu, kdy základní harmonická nemusí být vždy nejvyšší, je v kepsrtu situace jednodušší. Pro úspěšnou detekci frekvence postačuje najít první nejvyšší špičku (tedy maximální hodnotu v kepstru jednoho segmentu) a jako v případě detekce v časové oblasti využít vztahu (3.6) pro výpočet frekvence.

Pro přesnější výpočet by bylo možné využít detekci špičkových hodnot jako v případě korelační metody, tedy detekci několika špiček a průměrování jejich vzdáleností. Při použití této metody jsem však paradoxně dosahoval horších výsledků , než při jednoduché detekci jedné maximální hodnoty

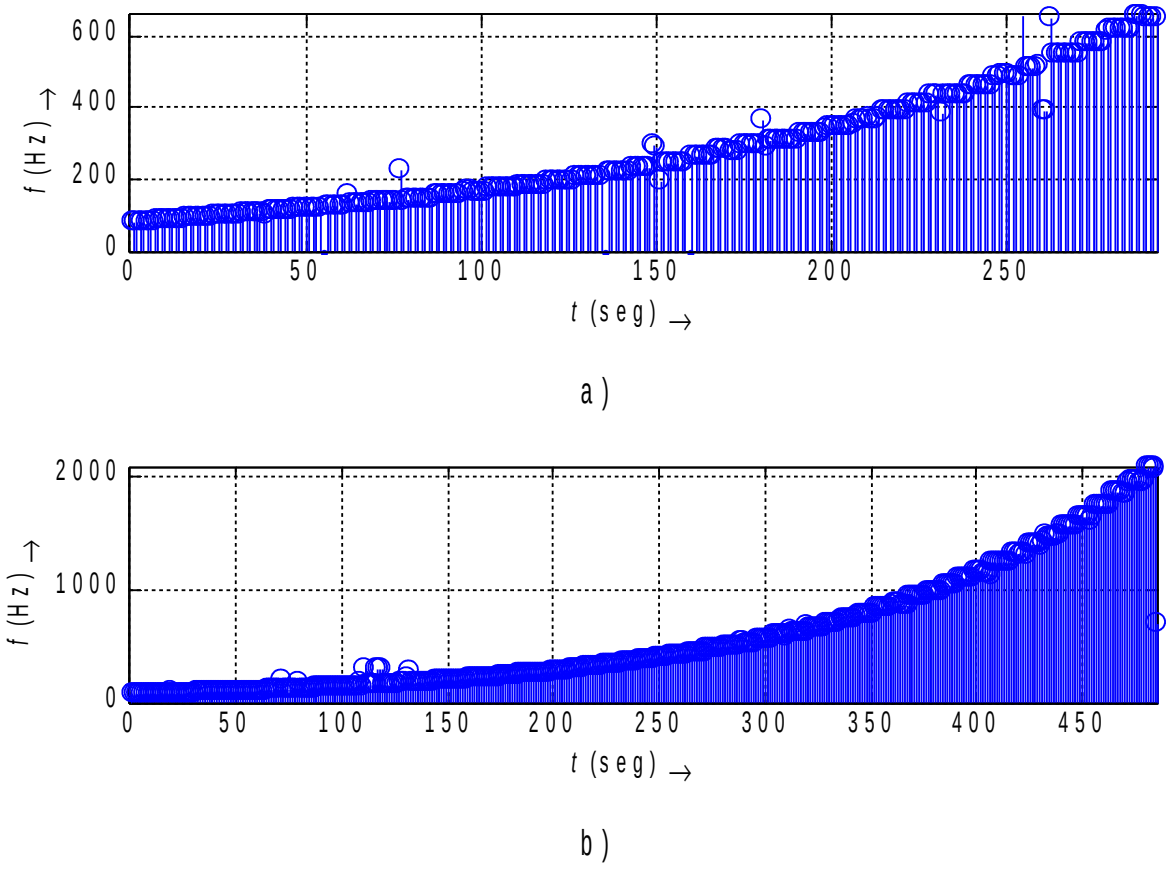
Tato metoda je výpočetně náročnější než spektrální, zejména kvůli dalším matematickým operacím se spektrem. Výsledky této metody jsou lepší než spektrální.

4 Porovnání metod

Pro porovnání metod jsem využil nahrávek kytary a piana. Na obou je zachycena vzestupná sekvence půltónů. U kytary je to posloupnost tónů od E (odpovídající notě č. 40) do E^2 (odpovídající notě č. 76), u piana potom od C (odpovídající notě č. 36) do C^4 (odpovídající notě č. 96). Porovnávány byly všechny výše zmíněné implementované metody a to jak z hlediska úspěšnosti detekce u obou nahrávek, tak metody vzájemně mezi sebou.

4.1 Porovnání korelační metody

Na obrázku 4.1 a) jsou detekované frekvence pro kytarový signál a na obrázku 4.1 b) signál piana. Pro oba signály je tato metoda velmi úspěšná. Při zpracování signálu piana dochází na kmitočtech okolo 120Hz k nepřesnostem. Děje se tak z důvodu silnějšího zastoupení vyšších harmonických složek než u ostatních tónů. Nepřesnost detekce by se dala odstranit změnou prahu na přibližně 45% maximální hodnoty autokorelační funkce. Jelikož jsem porovnával oba vstupní signály se stejným nastavením prahu, volil jsem práh přijatelný pro oba signály a to 30%. Nepřesnosti detekce v kytarovém signálu jsou většinou způsobeny změnou frekvence v průběhu jednoho segmentu.



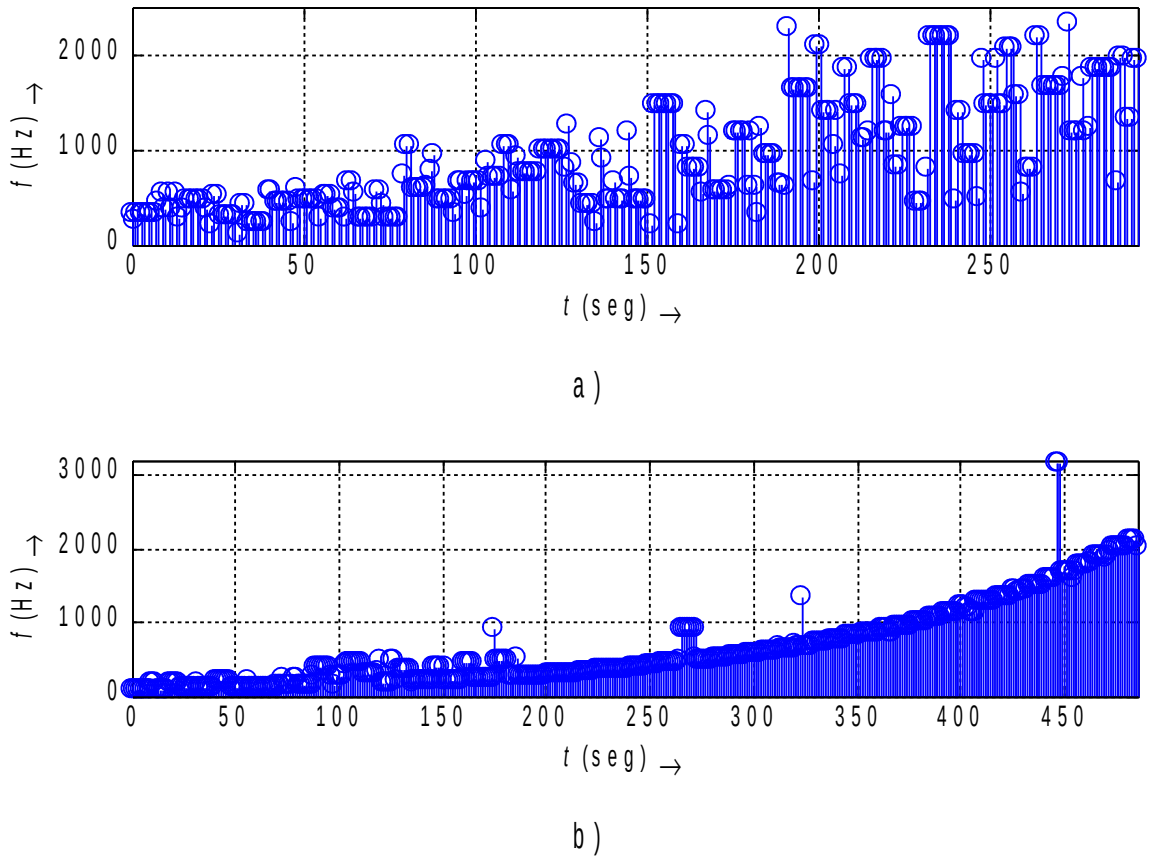
Obr. 4.1: Porovnání detekovaných frekvencí korelační metodou u kytary (a) a piana (b)

4.2 Porovnání spektrální metody

Na obrázku 4.2 a) jsou detekované frekvence z testovacího signálu kytary. Je patrné, že detekce ve spektrální oblasti při použití metody popsané v části 3.3 není příliš úspěšná. Důvodem je zejména struktura spektra kytarového signálu, vyšší harmonické složky mají vysoké hodnoty a základní harmonická je často „utopena v šumu“. Z toho důvodu dochází k nepřesnému určení jejího umístění a je chybně detekována některá z vyšších harmonických složek.

Situace se zlepší při použití testovacího signálu piana (Obr. 4.2 b)). Při nižších tónech je zastoupení alikvotních tónů ještě poměrně výrazné a chybovost je v této části vyšší, naproti tomu na vyšších frekvencích je tato metoda poměrně přesná. Avšak ani spektrální průběh signálu piana není pro tuto metodu detekce příliš příznivý.

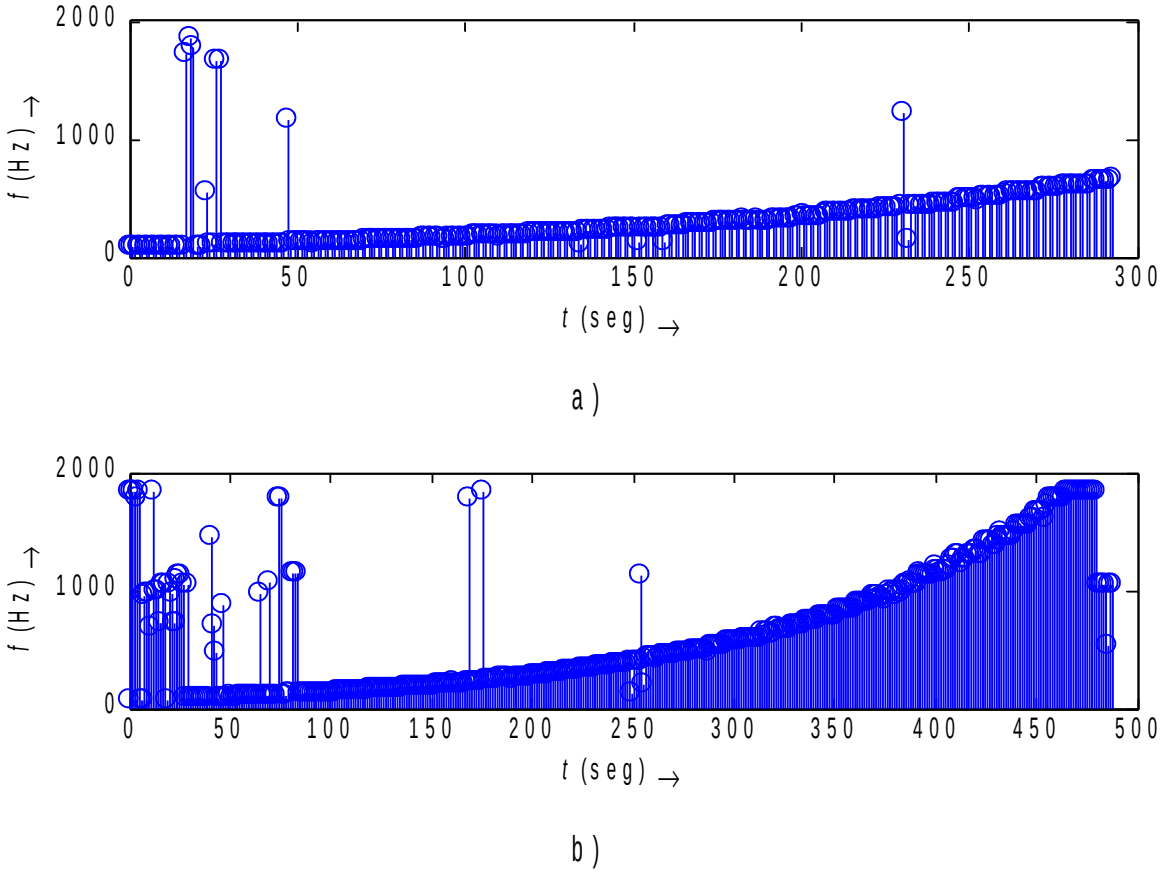
Chybovost metody by se dala ovlivnit zejména použitým algoritmem pro určení a odlišení první harmonické složky od ostatních alikvotních tónu.



Obr. 4.2: Porovnání detekovaných frekvencí spektrální metodou u kytary (a) a piana (b)

4.3 Porovnání kepstrální metody

V případě kepstrální metody je postup podobný jako u korelační metody. Určení prahu je v tomto případě poněkud problematičtější, špičky nejsou tak výrazné. Ovšem detekce pouze maximální hodnoty v kepstru se osvědčila jako velmi úspěšná.

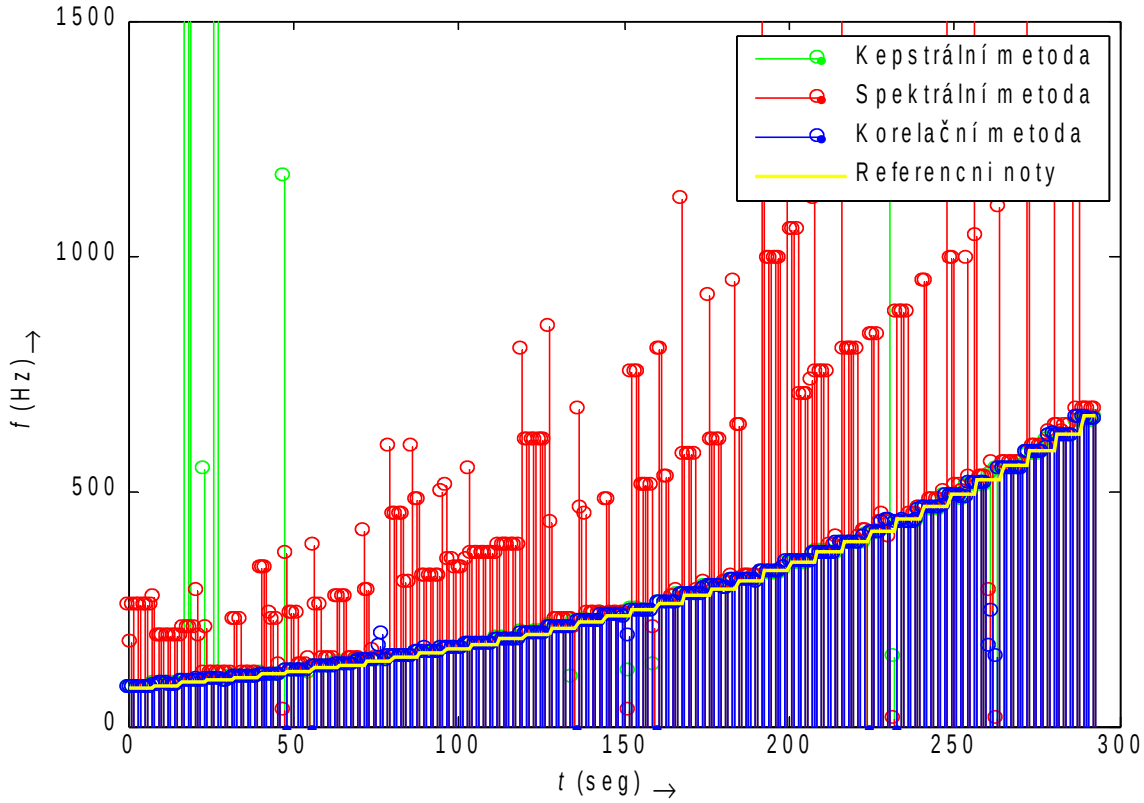


Obr. 4.3: Porovnání detekovaných frekvencí kepstrální metodou u kytary (a) a piana (b)

Jak je vidět na obrázku, pro kytarový signál (Obr. 4.3 a)) je detekce poměrně přesná. Pro signál piana (Obr. 4.3 b)) je situace horší. Celkově je tento způsob detekce úspěšnější než spektrální metoda. V porovnání s korelační metodou je tato náročnější na výpočet, kvůli výpočtu logaritmu.

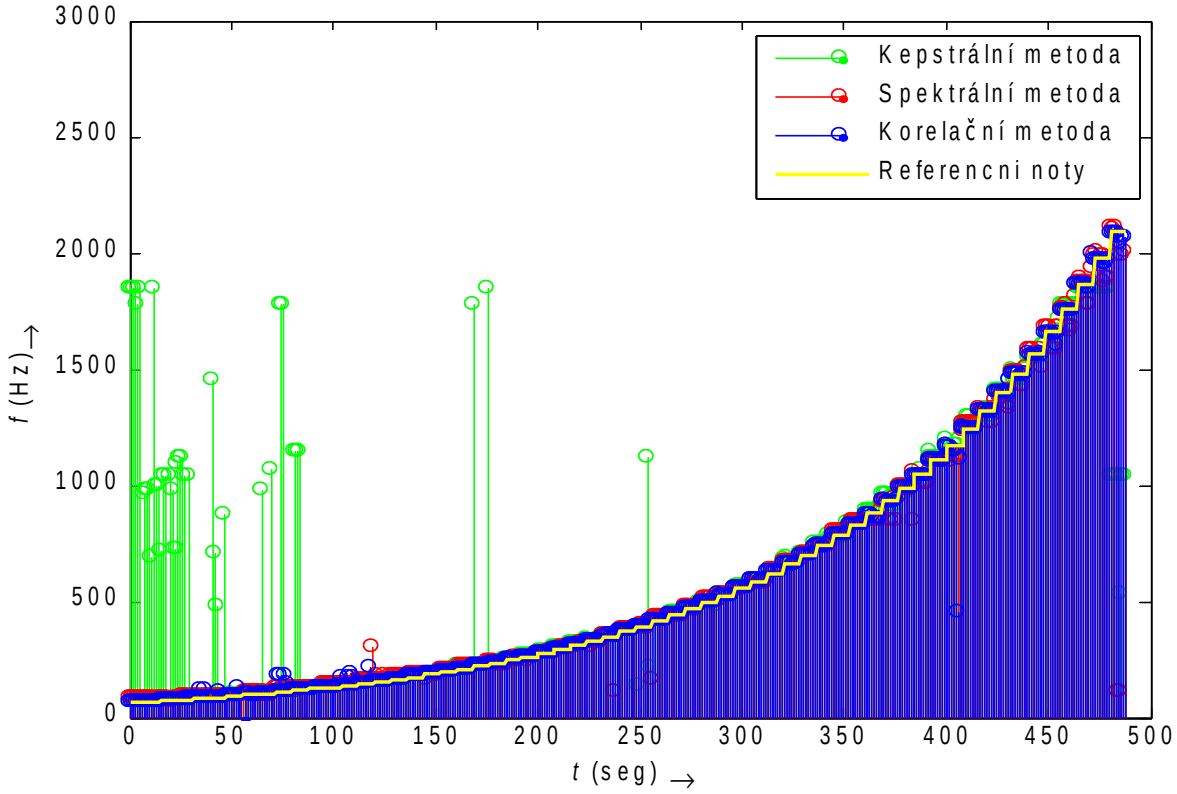
4.4 Vzájemné porovnání metod

Porovnával jsem vzájemnou úspěšnost jednotlivých metod na jednom signálu. Pro vstupní testovací signál (kytaru) jsem vynesl do jednoho obrázku frekvence detekované všemi metodami. Jak je vidět, u obou signálů je nejméně přesná spektrální metoda. Je to právě z důvodu poměru zastoupení vyšších harmonických složek ve spektru signálu kytary. Metoda detekce v kepstru je úspěšnější. Nejpřesnější je korelační metoda.



Obr. 4.4: Porovnání metod detekce základní frekvence tónu, testovací signál kytara

U testovacího signálu piana jsou všechny metody úspěšnější. Přesnost spektrální metody se blíží korelační, je to dáno spektrálním složením signálu piana, kde je základní složka nejsilnější a tak vyhovuje použitá detekce maximální složky. Nejpřesnější je opět korelační metoda.



Obr. 4.5: Porovnání metod detekce základní frekvence tónu, testovací signál piano

V testovacím signálu piana jsou zastoupeny půltóny od C do c^4, každá nota trvá 1 dobu. Vytvořil jsem tedy referenční vektor s odpovídajícím čísly not dle MIDI (interval not 36-96), přičemž jsou použity stejné parametry jako u detekovaných not. Zejména nejkratší délka noty, ta je zvolena 1/32. Tím je dáno, že jednu dobu reprezentuje 8 not. Pro určení úspěšnosti metod je použit následující postup:

- určení základní frekvence tónu různými metodami z testovací melodie piana
- přiřazení čísla noty každé detekované frekvenci dle vztahu (5)
- zaokrouhlení čísla noty na nejbližší celé číslo
- porovnání detekovaných čísel not s referenční hodnotou

Z poměru počtu správně určených hodnot vůči celkovému počtu not se následně vypočítá úspěšnost detekce.

Tab. 4.1: Úspěšnost detekce

Metoda detekce	Autokorelační	Spektrální	Kepstrální
Úspěšnost	93,6 %	45,3 %	81,1 %

5 Program pro převod melodie do MIDI

Úkolem této práce bylo také vytvořit funkční program pro převod melodie do MIDI souboru v programovacím jazyce C++. Program jsem vytvořil ve formě MEX funkce použitelné v prostředí MATLAB. Výhodou použití MEX funkcí v MATLABu je jejich rychlost a možnost využití všech možností jazyka C++. Program je vytvořen tak, že lze využít vestavěných funkcí matlabu pro načítání zvukových souborů. Program se skládá z několika samostatně použitelných částí. Pro tyto části lze doprogramovat jakékoliv rozhraní jako náhradu za stávající MEX rozhraní. První částí je funkce `WAV2MIDI` provádějící detekci základního tónu s následným převodem na čísla not, dalším celkem je funkce `MIDIwrite`, která z vektoru čísel not vytváří MIDI SMF soubor. Tyto funkce jsou zahrnuty v MEX souboru, který zprostředkovává načtení a předání všech potřebných proměnných a interakci s Matlabem.

5.1 Externí rozhraní Matlabu

Program Matlab lze propojit s dalšími programovacími jazyky a využít tak jejich možnosti. Kromě rozhraní COM (Component Object Model), DDE (Dynamic Data Exchange), Java a dalších obsahuje možnost využití programů napsaných v jazyce C nebo Fortran – takzvaných MEX souborů.

Jedná se o dynamicky připojitelnou knihovnu, kterou interpreter Matlabu automaticky načte a provede. Formát tohoto souboru je závislý na použité platformě a podle toho se liší i přípona souboru. Pro operační systém Windows je přípona .mexw32, nebo .mexw64, podle použité platformy. Spolu s Matlabem je dodáván překladač jazyka C (Lcc), ale pro překlad je možné použít jakýkoliv překladač.

MEX soubor se obecně může skládat ze dvou částí. První je procedura, která zajišťuje interakci s Matlabem, jedná se zejména o načítání a předávání proměnných mezi programy. Druhou částí je samotná výpočetní procedura, která může však může být zakomponována do vstupní procedury.

Deklarace vstupní procedury je následující:

```
void mexFunction (int nlhs, mxArray *plhs[], int nrhs,
                  const mxArray *prhs[])
```

Argumenty této funkce jsou: `nlhs` – počet výstupních parametrů, `plhs` – pole s ukazateli na výstupní parametry, `nrhs` – počet vstupních parametrů, `prhs` – pole s ukazateli na vstupní parametry.

Programovací jazyk Matlab pracuje pouze s jedním datovým typem, kterým je *MATLAB Array* („pole“). Všechny v tomto programu používané proměnné (skaláry, vektory, matice, struktury, objekty) jsou ukládány jako tento datový typ. V jazyce C je tento datový typ reprezentován strukturou `mxArray`. Tato struktura kromě jiného nese tyto informace:

- svůj typ
- svoje rozměry
- data spojená s polem
- v případě čísel, zda jsou reálná nebo komplexní
- v případě struktur a objektů, počet polí a jejich názvy

Tato struktura se používá i pro předání zpracovaných hodnot zpět do Matlabu. Pro vytvoření tohoto datového typu se používají funkce v závislosti na typu ukládaných dat. Pro čísla se používá funkce `mxCreateNumericArray`, pro textové hodnoty `mxCreateCharArray`, atd. Pro práci s datovým typem `mxArray`, obecně slouží funkce s předponou `mx*`. Pro přístup k hodnotám se používá funkce `mxGetPr` (reálné prvky), `mxGetPi` (imaginární prvky). Ke zjištění parametrů matice slouží funkce `mxGetM` (počet řádků) a `mxGetN` (počet sloupců). Pro nastavení velikosti `mxSetM`, `mxSetN`. Pro alokaci a uvolnění paměti slouží funkce `mxMalloc` a uvolnění `mxFree`. [15]

V prostředí Matlabu se tyto mex soubory volají jako klasické funkce Matlabu vytvořené v m-file, tedy svým názvem a parametry.

5.2 Knihovna FFTW

Jedná se o knihovnu, která byla vyvinuta na Massachusetts Institute of Technology a jejími autory jsou Matteo Frigo and Steven G. Johnson. Oficiální název zní „Fastest Fourier Transform in the West“. Již podle názvu tato knihovna poskytuje rozhraní pro výpočty Fourierovy transfomace. Hlavní předností této knihovny je zejména rychlost a efektivita výpočtu, využívá rozšíření instrukčních sad různých procesorů jako jsou *SSE*, *SSE2*, *3DNow!*. Další výhodou je GPL licence neomezující využití této knihovny. Knihovna je napsána v jazyce C pro operační systém Linux, je však možné je využít i v dalších operačních systémech jak například Windows.

Knihovna umožňuje transformovat různé druhy vstupních dat od jednorozměrných reálných hodnot, přes komplexní hodnoty, až po vícerozměrná pole dat. Délka vstupních dat není omezena. Taktéž je možno využít vícevláknového paralelního zpracování [3]. Knihovna disponuje množstvím rozhraní, které lze využít pro různé druhy zpracování a požadovaných výsledků. Podporuje formát jak čistě reálných čísel, tak komplexních dat. Další možností je využití formátu takzvaných smíšených dat, ten je popsán níže. Dále bych se zaměřil pouze na

popis základní funkce, která je využity v tomto programu. Tou je transformace reálného 1D signálu.

Nejprve je třeba vytvořit takzvaný `fftw_plan`, který definuje parametry transformace. Podle názvu `fftw_plan_X_Y` lze určit, jaký způsob transformace bude zvolen. X reprezentuje, zda se jedná o reálnou nebo komplexní transformaci. Y určuje rozměr vstupních dat – zda se jedná o vektor (1d) nebo matici (2d). Pro případ jednorozměrného reálného signálu jsem zvolil tento plán, který zajišťuje transformaci je jednorozměrných reálných dat:

```
fftw_plan_r2r_1d(N, in, out, FFTW_R2HC, FFTW_ESTIMATE) ,
```

kde `N` je velikost zpracovávaného pole, `in` je pole vstupních hodnot, `out` výstupních hodnot. Další parametr určuje typ výstupních dat. Typ `FFTW_R2HC` určuje, že výstupní data budou mít takzvaný *halfcomplex* formát, tedy polovina výstupního pole bude obsahovat reálné hodnoty a polovina jejich komplexní část – je tedy využito symetričnosti spektra kolem poloviny vzorkovacího kmitočtu. V tomto případě poslední parametr (`FFTW_ESTIMATE`) určuje jakým způsobem bude funkce počítat – tato volba zajišťuje co nejrychlejší výpočet. Je možno také zvolit parametr `FFTW_MEASURE`, kdy funkce přepočítá transformaci několika algoritmy a vybere ten, který trvá nejkratší dobu.

Alokace vstupních a výstupních proměnných pro použití s FFTW se provádí použitím funkce `fftw_malloc(size_t n)`. Tato funkce zajišťuje zarovnání, aby bylo možno využít SIMD instrukcí. Takto alokovaná paměť musí být uvolněna pomocí funkce `fftw_free()`.

Provedení výpočtu zajišťuje funkce `fftw_execute(const fftw_plan p)`, parametrem je předem vytvořený fftw plán.

Když už definovaný plán není třeba, je vhodné ho uvolnit pomocí funkce `fftw_destroy_plan(const fftw_plan p)`, parametrem je opět pouze název plánu.

Při použití této knihovny pod operačním systémem Windows je nutné k výslednému mex souboru přiložit ještě knihovny zkompilované pro tento OS (libfftw3-3.dll, libfftw3f-3.dll, libfftw3l-3.dll). [3]

5.3 Funkce WAV2MIDI

V této funkci se provádí segmentace signálu, detekce základního tónu melodie pomocí autokorelační funkce a převod detekované frekvence na číslo noty. Deklarace funkce má následující podobu:

```
void det_not(double *x, int M, int FS, int tempo, int minnote, int
             npeak, int prah, double *noty_out )
```

Potřebné vstupní parametry jsou:

- `x` – vektor s načteným signálem,
- `FS` – vzorkovací frekvence signálu,
- `M` - délka vstupního signálu,
- `tempo` – tempo skladby v bpm, spolu s nejkratší detekovatelnou notou slouží k výpočtu délky segmentu,
- `minnote` – číselně vyjádřená nejkratší detekovatelná nota (1-celá, 2-půlová, 4-čtvrtinová, 8-osminová, atd.),
- `npeak` – počet maxim autokorelační funkce segmentu které se detekují a následně průměr jejich vzdáleností určuje frekvenci tónu,
- `prah` – hodnota prahu vyjádřená v procentech.

Výstupním parametrem je proměnná `noty_out`, kde jsou uloženy detekovaná čísla not.

V této funkci je tedy prvním krokem segmentace vstupního signálu. Segmentace je provedena na základě zvoleného tempa a nejkratší možné noty, aby každý segment odpovídal jedné notě této délky. Jednotlivé segmenty jsou pak podrobeny rychlé autokorelaci, zde je využita knihovna FFTW, která je popsána výše. Vzniklý korelogram je prahován, práh je zadáván relativně a konkretní hodnota je určena pro každý segment zvlášť jako procentuální část z maximální hodnoty v segmentu. V prahovaných segmentech se vyhledá definovaný počet špiček a jejich vzdálenosti se průměrují. Následně je vypočtena dle vztahu (3.6) frekvence. Vektor s detekovanými frekvencemi je předán funkci zajišťující přepočet frekvence na číslo noty dle vztahu (2.2). Detekované frekvence jsou ještě před zápisem do souboru „průměrovány“. Jedná se o jednoduchý kontrolní mechanizmus. V této funkci se kontrolují změny not. Pokud nastane změna noty mezi současnou a následující, jsou kontrolovány 2 okolní noty a pokud jde o notu, která má ojedinělou hodnotu odlišnou od okolních not je považována za chybnou a je nahrazena správnou notou. Výsledný vektor obsahující čísla not je dále použit v další funkci pro zápis do souboru. Vývojový diagram je znázorněn v příloze B.

5.4 Funkce detnot

Tato funkce má za úkol zapsat detekované frekvence do midi SMF souboru. Pro účely jednohlasé melodie jsem zvolil formát 0, tedy s jednou stopou (viz kapitola 2.4). Nejprve je třeba správně zapsat hlavičku souboru. Zejména je třeba vypočítat z velikosti zapisovaných dat správnou velikost stopy. Dále z údajů o tempu a počtu tiků ve čtvrťové notě nastavit správné časování. Funkce je deklarována následovně:

```
void zapisdosouboru(int N, double *in_char, double tempo, double
                    tpq, double minnote, double rychlostz )
```

kde `N` je délka vstupního vektoru s notami, `in_char` je vektor s notami k zápisu, `tempo` je tempo v bpm, `tpq` je počet tiků do čtvrťové noty, `minnote` je nejkratší detekovaná nota, jinými slovy délka zapisované noty. `Rychlostz` je rychlost stisku a vypnutí klávesy.

Ze vstupních údajů je tedy sestavena hlavička a následuje zápis jednotlivých not. U not je ještě před jejich zápisem kontrolováno, zda se nevyskytují v řadě za sebou stejné noty. Pokud ano, je zapsána pouze jedna nota s dobou trvání odpovídající počtu po sobě jdoucích stejných not. Následuje ukončení stopy. Vývojový diagram je znázorněn v příloze C.

6 Závěr

V práci jsou stručně popsány rozdíly mezi hudebním a řečovým signálem. Historie a základní informace o standardu MIDI.

Dále je popsán princip vybraných metod detekce základního tónu hudebního signálu. Snažil jsem se ověřit, zda tyto metody, které vycházejí z metod pro detekci základního tónu řečového signálu, jsou vhodné i pro signál hudební.

Nejvhodnější je autokorelační metoda, kde je pro detekci špičkových hodnot v korelogramu použito pouze prahování s následnou detekcí maxim v jednotlivých částech segmentu a průměrováním jejich vzdáleností.

Druhé nejlepší výsledky vykazovala kepstrální metoda, s úspěšností přes 80%. V kepstru bylo použito pouze prahování a detekce nejsilnější kepstrální složky.

Nejhorší výsledky měla spektrální metoda, která naráží zejména na problém neznalosti nástroje a spektrálních vlastností jeho signálu. Pokud by byl nástroj znám, lze tuto metodu přizpůsobit a poskytovala by lepší výsledky.

Testování probíhalo na signálu piana, který byl rostoucí po půltónech. Signál byl segmentován způsobem popsaným v kapitole 3.1. Nejkratší detekovatelná nota byla 1/32, tempo 120 bpm, bez překrytí, s obdélníkovým oknem.

Součástí zadání práce bylo vytvořit program v jazyce C, který bude zajišťovat převod melodie ze souboru .wav do standardního MIDI souboru. Program jsem vytvořil formou MEX souboru pro program Matlab. Jako vývojové prostředí jsem použil MS Visual Studio 2008. Segmentace je prováděna způsobem popsaným v části 3.1. Vzhledem k úspěšnosti v předchozím testování jsem zvolil pro detekci základního tónu autokorelační metodu. Autokorelace je implementována s použitím FFT. Pro výpočet FFT jsem využil volně šiřitelnou knihovnu FFTW, zejména kvůli rychlosti výpočtu. Po přepočtu na číslo noty a základní korekci ojedinělých chybných hodnot jsou tyto zapisovány do souboru SMF formátu 0. Více stejných not je transformováno do jedné déle trvající noty. Vzhledem ke struktuře zdrojových C funkcí je tyto možné také použít samostatně s jiným rozhraním než pro Matlab.

Seznam použité literatury

[1] ATASSI, H. *Metody detekce základního tónu řeči. Elektrorevue* [online], 2008. Dostupný z http://www.elektrorevue.cz/cz/clanky/zpracovani-signalu/5/metody-detekce-zakladniho-tonu-reci/. ISSN 1213-1539.

[2] BOŘIL, H. *Kytarový MIDI převodník*. [s.l.], 2003. 72 s. Diplomová práce. ISBN 80-200-1309-1.

[3] FFTW. *FFTW Home Page* [online]. 2009 [cit. 2009-04-10]. Dostupný z WWW: <http://www.fftw.org/>.

[4] FORRÓ, D. *Svět MIDI*. GRADA publishing, 1997. ISBN 80-7169-415-6.

[5] HOIDEKER, J. *MIDI.* Studentské sborníky ZČU v elektronické podobě [online]. 2000 [cit.2008-12-01]. Dostupný z WWW: <http://www.kiv.zcu.cz/~herout/html_sbo/midi/toc.html>.

[6] JÁN, J. *Číslicové zpracování a analýza signálů.* Elektronická skripta k předmětu BCZA. [online]. 2003. Dostupný z WWW: <www.feec.vutbr.cz/et>

[7] MIDI Manufacturers Association Inc. *MIDI Manufacturers Association* [online]. c2008 [cit. 2008-12-05]. Dostupný z WWW: <http://www.midi.org/>.

[8] OBDRŽÁLEK, J. *Přehled a výklad hudebních ladění*. *Laboratoř biokybernetiky a počítačové podpory výuky* [online]. 2004 [cit. 2008-12-02]. Dostupný z WWW: <http://patf-biokyb.lf1.cuni.cz/projects/ladeni/prehled_a_vyklad_hudebniho_ladeni.doc>.

[9] PSUTKA, J, et al. *Mluvíme s počítačem česky.* 1. vyd. Praha : ACADEMIA, 2006. 752 s.

[10] SCHIMMEL, J. *Komunikační rozhraní MIDI*. Elektrorevue : Časopis pro elektrotechniku [online]. 2002, č. 69 [cit. 2002-12-20]. Dostupný z WWW: <http://www.elektrorevue.cz/clanky/02069/index.html>

[11] SCHIMMEL, J. *Studiová a hudební elektronika.* Elektronická skripta, VUT v Brně [online]. 2004 [cit. 2008-11-27]. Dostupný z WWW: <http://www.feec.vutbr.cz/et/skripta/utko/Studiova_a_hudebni_elektronika_S.pdf>.

[12] SKARNITZL, R. *Psychoakustika*. Doplňkové materiály k výuce - Fonetický ústav Filozofické fakulty Univerzity Karlovy [online]. 2008 [cit. 2008-12-01]. Dostupný z WWW: <http://fu.ff.cuni.cz/vyuka/akustika/3_psychoakustika.pdf>.

[13] SMÉKAL, Z. *Číslicové zpracování signálů.* Elektronická skripta k předmětu MCSI. [online]. 2007,s. 1-149. Dostupný z WWW: <www.feec.vutbr.cz/et>

[14] SYROVÝ, V. *Výchozí teorie barvy zvuku a jejich současná akustická interpretace*. Zvukové studio HAMU [online]. 2002 [cit. 2008-11-25]. Dostupný z WWW: <www.h.amu.cz/zvuk/studio/dokumenty/Lit28z.pdf >.

[15] The MathWorks, Inc.. *MEX-files Guide* [online]. c1994-2009 [cit. 2009-04-18]. Dostupný z WWW: <http://www.mathworks.com/support/tech-notes/1600/1605.html>

[16] Yamaha Corporation. *What´s mLAN* [online]. [2005] , 2007 [cit. 2008-12-01]. Dostupný z WWW: <http://www.yamahasynth.com/products/mlan/index.html>.

Seznam zkratek, veličin a symbolů

AES3 Audio Engineering Society (SMTPTE/AES3 – způsob kódování času)
BPM Beats per minute (úderů za minutu)
COM Component Object Model
DDE Dynamic Data Exchange
EBU European Broadcasting Union (SMTPTE/EBU – způsob kódování času)
FFT Fast Fourier Transform (rychlá Fourierova trnsformace)
FFTW Fastest Fourier Transform in the West
GM General Midi standard
GPL General Public License (všeobecná veřejná licence)
IEEE The Institute of Electrical and Electronics Engineers
IFFT Inverse Fast Fourier Transform (inverzní rychlá Fourierova transformace)
ISO International Standard Organization
JMSC Japanese Midi Standard Committee
MIDI Musical Instrument Digital Interface (digitální rozhraní pro hudební nástroje)
mLAN Music Local Area Network
MMA MIDI Manufacturers Associatoin
MS Microsoft
MSB Most Significant Bit (bit s nejvyšší hodnotou)
NAMM National Association of Music Merchants
SIMD Single Instruction Multiple Data
SMF Standard Midi File (standardní MIDI soubor)
SMPTE Society of Motion Picture and Television Engineers
SSE Streaming SIMD Extensions
SSE2 Streaming SIMD Extensions2
USI Universal Syntethize Interface - předchůdce MIDI

Δf frekvenční rozlišení
$c_R[k]$ reálné kepstrum
f_{VZ} vzorkovací frekvence
T_{VZ} vzorkovací perioda
$r_{XX}(\tau)$ hodnota autokorelační funkce
$R(k)$ jednostranná autokorelace
$S(k)$ obraz diskrétní Fourierovy transformace
$S_{XX}(\omega)$ výkonové spektrum signálu
$\Re(x)$ reálná část komplexního čísla

Seznam příloh

A Obsah CD

Adresáře

Text - obsahuje elektronickou verzi práce ve formátu .pdf a .odt

Matlab - obsahuje skripty pro matlab, ve kterých jsou implementovány všechny metody detekce základního tónu a jejich porovnání.

MSVS - obsahuje projekt psaný ve MS Visual Studiu 2008, zdrojové soubory .c, výsledný MEX soubor.

MEX - obsahuje výsledný MEX soubor včetně nutných knihoven, testovacích signálů a ovládacího skriptu pro Matlab.

Soubory

Matlab

frq2not.m - funkce využívaná ve skriptech pro převod detekované frekvence na číslo noty

kepstrogram.m - obsahuje implementaci algoritmu detekce základního tónu v kepstrální oblasti - pro oba testovací signály

korekce1.m - funkce využívaná ve skriptech pro porovnávání, mediánová korekce chybných not

korelace.m - obsahuje implementaci algoritmu detekce základního tónu v časové oblasti - pro oba testovací signály

kytara_cut_start_end.wav - testovací signál kytara

metody_porovnani_kytara - skript porovnávající všechny metody detekce na testovacím signálu kytary

metody_porovnani_kytara - skript porovnávající všechny metody detekce na testovacím signálu kytary

piano_cut_start_end.wav - testovací signál piana

kytara_cut_start_end.wav - testovací signál kytara

spektrogram.m - obsahuje implementaci algoritmu detekce základního tónu v časové oblasti - pro oba testovací signály

MSVS - Obsahuje všechny zdrojové soubory vytvořené v rámci projektu ve MS Visual Studio 2008.

MEX

WAV2MIDI.mexw32 - samotná MEX funkce pro Matlab

ovladani.m - ovládací skript pro Matlab, který využívá vytvořenou MEX funkci

libfftw3l-3.dll, libfftw3f-3.dll, libfftw3-3.dll - dynamické knihovny FFTW

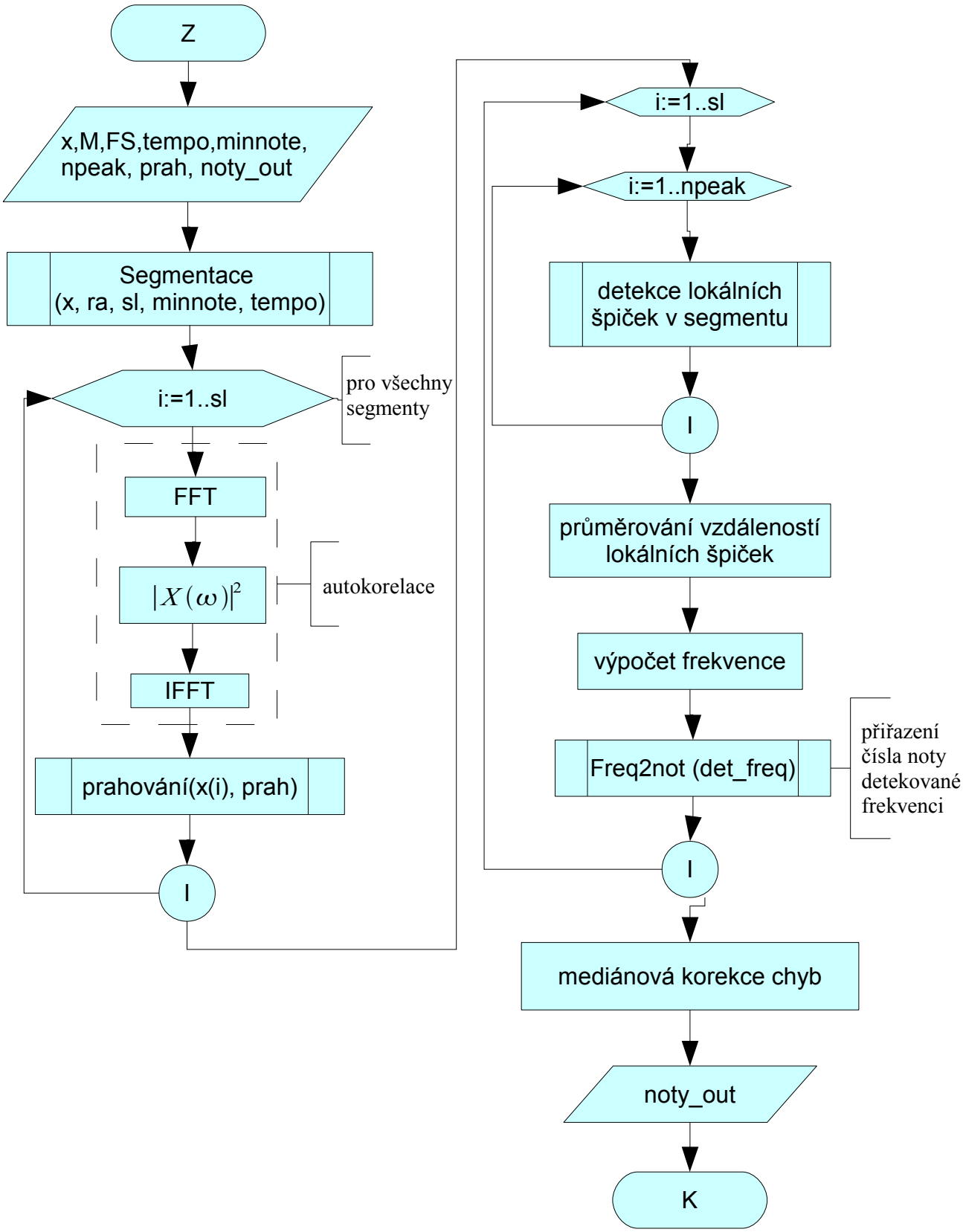
noty.mid - výstupní SMF soubor obsahující výslednou melodii

Text

diplomova_prace.pdf - elektronická verze práce formát PDF

diplomova_prace.odt - elektronická verze práce formát OpenOffice

B Vývojový diagram funkce detnot



C Vývojový diagram funkce MIDIWrite