

Počítačový syntetizér spevu

DIPLOMOVÁ PRÁCA

Michal Šukola

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY
KATEDRA APLIKOVANEJ INFORMATIKY

9.2.9 Aplikovaná informatika

Vedúci: RnDr. Marek Nagy

BRATISLAVA 2010



KATEDRA APLIKOVANEJ INFORMATIKY
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY
UNIVERZITA KOMENSKÉHO, BRATISLAVA

Počítačový syntetizér spevu

(Diplomová práca)

MICHAL ŠUKOLA

Čestne prehlasujem, že som túto diplomovú prácu
vypracoval samostatne s použitím citovaných zdrojov a
odbornou pomocou školiteľa.

.....

Abstrakt

V práci sa zaoberáme návrhom a implementáciou počítačového syntetizéra spevu. Syntetizér používa ako vstup notový zápis piesne, jej text a nahrávku narozeného textu. Pri návrhu vychádzame z kapitoly o syntéze spevu, kde popisujeme jednotlivé techniky a metódy syntézy. Popisujeme technológie použité pri návrhu a implementácií. Hlavnou motiváciou je navrhnúť a vytvoriť taký počítačový syntetizér spevu, ktorý využitím notového zápisu a narozeného textu vygeneruje nahrávku danej piesne s charakteristickou farbou hlasu rečníka.

Obsah

1	Úvod	1
1.1	Cieľ práce	2
1.2	Štruktúra práce	2
2	Techniky syntézy spevu	3
2.1	Syntéza spevu na základe modelovania sinusoid	3
2.1.1	Sínusový model	4
2.1.2	Priebeh syntézy	5
2.1.3	Zhrnutie metódy	7
2.2	Syntéza spevu použitím neuronových sietí pre modelovanie vibráta	8
2.2.1	Vibráto	8
2.2.2	Zistovanie parametrov vibráta	9
2.2.3	Neurónova sieť	10
2.2.4	Priebeh syntézy	12
2.2.5	Zhrnutie metódy	12
2.3	Syntéza spevu pomocou metódy samplingu	13
2.3.1	Zvukový priestor	13
2.3.2	Komponenty syntézy	14
2.3.3	Priebeh syntézy	16
2.3.4	Zhrnutie metódy	16
2.4	Syntéza prevodom reči na spev pomocou jedinečných akustických znakov spevu	16
2.4.1	Komponenty syntézy	17
2.4.2	Priebeh syntézy	19
2.4.3	Zhrnutie metódy	20

3 Návrh a implementácia	21
3.1 Reprezentácia piesne	21
3.1.1 Súbor s piesňou	22
3.2 Určenie melódie	23
3.3 Tvarovanie krvky hlasivkového tónu F_0	27
3.3.1 F_0 Efektor	27
3.3.2 Vibráto	28
3.3.3 Jemné kolísanie	30
3.3.4 Digitálne filtre	31
3.3.5 Trieda filtra	34
3.3.6 Butterworthov dolnopriepustný filter	35
3.3.7 Jemné kolísanie pomocou filtra	36
3.3.8 Príprava	38
3.4 Syntéza	38
3.4.1 Syntéza prostredníctvom LPC koeficientov	39
3.4.2 Syntéza prostredníctvom algoritmu PSOLA	43
3.4.3 Celkový priebeh syntézy	47
4 Záver	50
4.1 Zhrnutie	50
4.2 Anketa	50
4.3 Výsledky	51
4.4 Budúca práca	51
Prílohy	54

Kapitola 1

Úvod

Spev nás sprevádza v celom našom živote a je neodmysliteľnou súčasťou snáď každej kultúry. Je ale len málo tých, ktorí sa spevu venujú profesionálne. Keďže spev je časťou takmer každej multimedialnej produkcie je teda potreba zohnať speváka, ktorý poskytne potrebnú nahrávku. Nie je žiadnym prekvapením, že každé vystúpenie speváka je finančnou záťažou pre náš multimedialny projekt. Tak ako sa postupne prestali používať nahrávky skutočných nástrojov či orchestrálnych produkcií, môže tento osud postihnúť aj živý spev¹. Nejde ale len o finančnú stránku, ktorá nás motivuje sa zaoberať syntézou spevu.

V dnešnej dobe sa výskum v oblasti spracovania a syntézy ľudského hlasu zaoberá skôr rečou ako jednej z možnej formy prezentácie hlasu. Je to pochopiteľné, keďže aplikácie pre spracovanie alebo generovanie reči sú v praxi použiteľné v mnohých odvetviach. Spev je ale neoddeliteľnou súčasťou ľudského hlasu a pre správne pochopenie a popis vokálneho traktu je bezosporu potrebné sa zaoberať budovaním modelov schopných parametricky popísať aj priebeh spevu. Keďže modelov popisujúcich vokálny trakt pre potreby reči je mnoho, je rozumné sa zaoberať tým či nie je možné daný model použiť aj pre potreby spevu alebo či by sa nedal pre ne rozšíriť.

Študovanie spevu je zaujímavé aj pre rôzne formy, ktoré môže spev nadobúdať. Zaujímavé sú iste faktory, ktoré určujú typ spevu, jeho kvalitu, aký pocit vyvolajú v poslucháčovi či metódu prednesu. Len veľmi málo výskumu je však venované parametrizácii týchto faktorov.

Osobnou motiváciou je pohľad na spev s dvoch strán. Keďže študujem spev na konzervatóriu, mám možnosť vnímať spev empiricky, avšak v tejto práci sa venujem jeho štúdiu

¹Ukážkou tohto môže byť film Titanic, v ktorom bola doprovodná hudba nasyntetizovaná

a syntéze z fyzikálneho a informatického pohľadu.

1.1 Cieľ práce

Cieľom tejto práce je automatická syntéza spevu využitím notového zápisu a naročného textu. Budeme sa snažiť navrhnúť a implementovať taký počítačový syntetizér spevu, ktorý využitím notového zápisu a naročného textu vygeneruje nahrávku danej piesne s charakteristickou farbou hlasu rečníka.

1.2 Štruktúra práce

V druhej kapitole sa venujeme prehľadu a popisu techník syntézy spevu. Popisujeme vlastnosti týchto techník, výhody a nevýhody ich používania.

V tretej kapitole hovoríme o návrhu a implementácii syntetizéra. Popisujeme jeho komponenty, ich vlastnosti a popis funkčnosti. Na záver kapitoly popisujeme výslednú aplikáciu.

Štvrtá kapitola – záver zhodnocuje dosiahnuté výsledky, porovnáva ich s cieľom práce. Hovorí o ľažkostiach a objavoch pri tvorbe počítačového syntetizéra spevu.

Kapitola 2

Techniky syntézy spevu

Táto kapitola prezentuje rôzne techniky a prístupy, ktoré sa používajú pri syntéze spevu.

V syntéze spevu podobne ako v syntéze reči existujú dva hlavné prístupy k tejto otázke.

Prvý vychádza z faktu, že vokálny trakt dokážeme popísať pomocou parametrov. Syntéza potom spočíva v poskytnutí parametrov do modelu a v postupnom dodávaní vstupov podľa definície modelu. Príkladom je Fujisakiho model (kapitola 3 v [11]). Výsledky syntetického spevu podľa týchto modelov sú v porovnaní s originálnym spevom neuspokojivé a to hlavne kvôli množstvu parametrov, ktoré obsahujú a ich pracnému zisťovaniu a nastavovaniu pre model. Aj keď v tomto smere už bol učinený určitý pokrok [14].

Druhým prístupom k syntéze spevu je použitie vzorkovania (samplingu). V skratke sa jedná o generovanie spevu pospájaním vzoriek hlasu z databázy. Tento prístup sa s úspechom používa pri syntéze zvuku hudobných nástrojov, no použitie v speve sa ukazuje ako oveľa ťažšia úloha vzhľadom k rozmanitosti hlasového prednesu spevu. Väčšina prístupov k syntéze spevu, ktorých kvalita je blízka originálному spevu používa v istej miere sampling [3]. Hlavnou nevýhodou je pamäťová náročnosť a nutnosť mať rozsiahlu databázu vzoriek.

2.1 Syntéza spevu na základe modelovania sinusoid

Aby sme mohli prezentovať túto metódu syntézy, musíme predstaviť sínusový model.

2.1.1 Sínusový model

V tomto modeli je vzruchový signál reprezentovaný ako suma konečného počtu sínusových parametrov pri hlasivkovej frekvencii (hlasivkovom tóne) a jej harmonických frekvenciach (alikvótach).

V tomto modeli je vstupný hlasový signál daný

$$s(n) = \sum_{l=1}^L A_l \cos(\omega_l n + \phi_l), \quad (2.1)$$

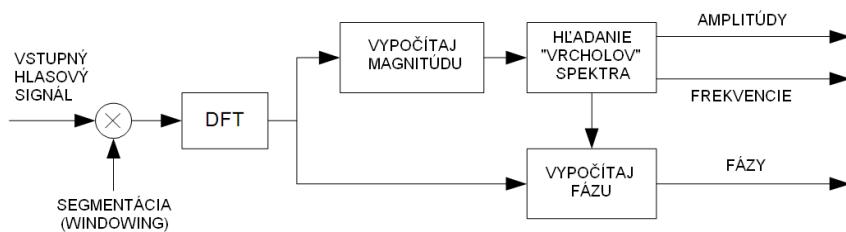
kde L je počet sínusových parametrov a A_l , ω_l a ϕ_l sú (s časom sa meniace) parametre reprezentujúce amplitúdu, frekvenciu a fázu každej sínusovky. Ak sa signál rozsegmentuje a pozrie sa na k -ty segment potom syntetický hlasový signál pre tento segment $\tilde{s}^k(n)$ môže byť reprezentovaný ako

$$\tilde{s}^k(n) = \sum_{l=1}^{L^k} A_l^k \cos(\omega_l^k n + \phi_l^k), \quad (2.2)$$

kde L^k je počet parametrov v k -tom segmente, A_l^k , ω_l^k a ϕ_l^k sú amplitúda, frekvencia resp. fáza v k -tom segmente.

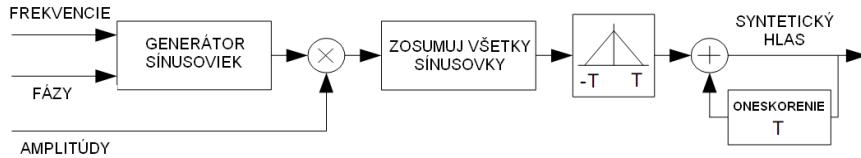
Priebeh analýzy vstupného hlasového signálu (zisťovania parametrov) je na obr. 2.1 a priebeh syntézy je ukázaný na obr. 2.2. V prípade analýzy sa prejde vstupný hlasový signál rozdelený na okienka (časti signálu). Následne sa vypočíta diskrétna Furiérova transformácia (DFT) a na spektre DFT sa nájdú vrcholy spektra. Takto možno získať zoznam frekvencií a korenšpondujúcich amplitúd týchto frekvencií.

Počas syntézy sa hlasový signál zrekonštruuje pomocou týchto zistených parametrov[6].



Obrázok 2.1: Princíp analýzy signálu v sínusovom modeli

Teraz, keď sme povedali čo je sínusový model, môžeme sa vrátiť k metóde syntézy pomocou modelovania sínusoviek.



Obrázok 2.2: Princíp syntézy signálu pri sínusovom modeli

2.1.2 Priebeh syntézy

Priebeh syntézy tejto metódy syntézy je ukázaný na obr. 2.3. Najprv užívateľ popíše vlastnosti spevu (napr. vibrato, crescendo¹) a zadá fonetický prepis slov piesne. Tieto vlastnosti sa uložia do vstupného súboru vo formáte MIDI. Následne sa z databázy hlasových dát vyberú také parametre, ktoré zodpovedajú zoskupeniu foném, ktoré chceme nasynthetizovať. Samozrejme sa prihliada aj na kontext, v ktorom sa tieto fonémy vyskytujú. Tieto zoskupenia sa považujú za jednotky. Potom sa jednotlivé jednotky spájajú pomocou overlap-add algoritmu. Aby sa dali do syntézy zahrnúť vlastnosti spevu - dĺžka nôt, výška tónu a spektrálna charakteristika hlasu, používajú sa na tento účel práve parametre sínusového modelu.

Zo spomenutého priebehu syntézy zatiaľ nie je jasné ako prísť k databáze hlasových dát. Ďalej sú zaujímavé najmä časti, ktoré popisujú, ako sa v priebehu syntézy do výstupu zakomponujú vlastnosti spevu.[10]

Zbieranie hlasových dát

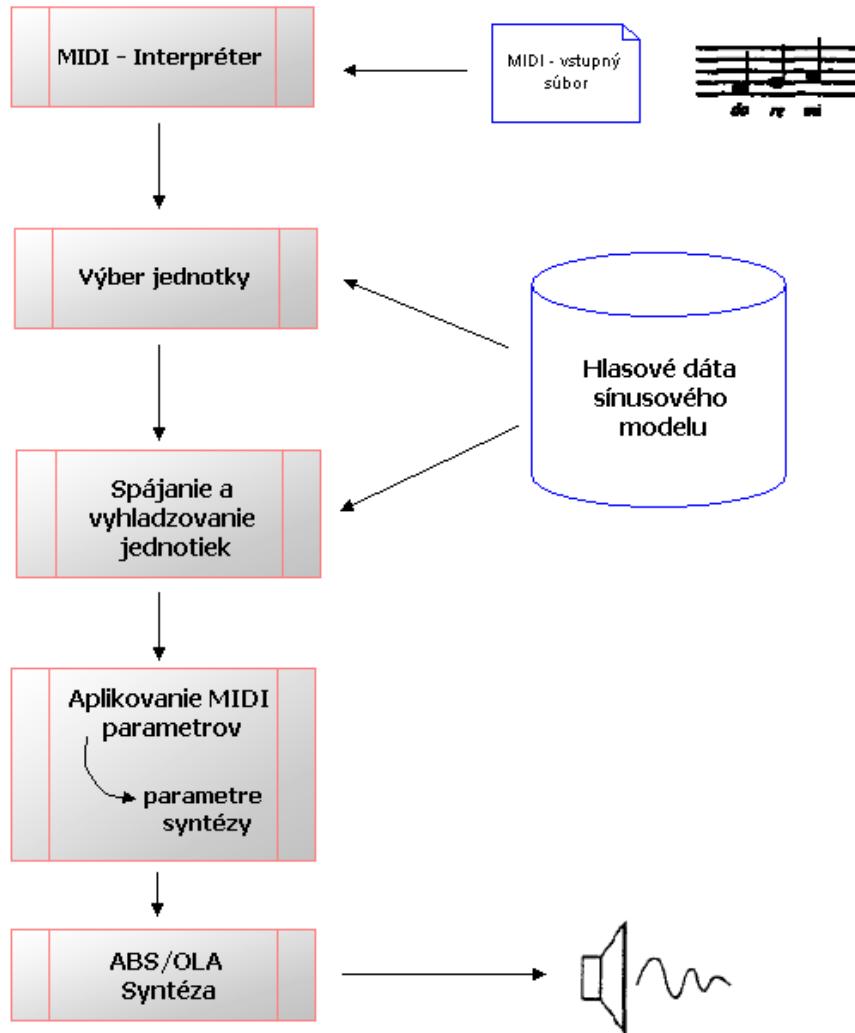
Pri zbieraní hlasových dát je snaha vytvoriť takú databázu, ktorá by adekvátnie pokryla čo najviac alofonických variácií² foném v rozličných kontextoch. Zrejme, čím viac týchto variácií sa pokryje, tým lepší výsledok možno dostať.

Ďalším dôležitým faktom pri budovaní databázy je, že na rozdiel od reči sa v spevove prikladá oveľa väčšia váha samohláskam ako spoluohláskam, a že pre priblženie sa k naturálnosti spevu je dôležitá skôr kvalita spevu ako zrozumiteľnosť slov.

Aby sa predišlo problémom pri spájaní jednotiek, rozdelia sa jednotky (variácie foném) do tried na tie v tvare $C_L V$, kde C_L je taká spoluohláska, ktorá bola vyslovená ako prvá

¹hudobný termín - postupné zosilňovanie

²také zoskupenia zvukov, ktoré zodpovedajú rovnakým fonémam



Obrázok 2.3: Blokový diagram sinusového systému syntézy

(pred ňou nebola žiadna iná fonéma) a V je samohláska a do triedy v tvare VC_R , kde C_R je spoluľáska, ktorá je vyslovená ako posledná. Je potrebné teda naplniť databázu hlasových dát, aby sa pokryli triedy C_LV a VC_R tak, že všetky kombinácie spoluľások a samohlások budú obsiahnuté. Aby sa predišlo artefaktom pri spájaní jednotiek zostaví sa aj trieda C_LVC_R a naplní sa vzorkami.

V [10] takto dostali okolo 500 vzoriek, ktoré boli nahraté mužským spevákom, orezané o sprievodné ticho, označené fonémami a poslané na analýzu do sínusovému modelu.

Modelovanie vlastností spevu

Vibráto – V tomto modeli je možné vibráto (viď časť 2.2.1) modelovať veľmi ľahko. Modulácie pri vibráte môžu byť vykonané pomocou modifikovania hlasivkového tónu F_0 . Tieto môžu byť vykonané až po tom, čo sa odstráni spektrálna obálka, ktorá predstavuje vokálny trakt (odstránenie násobných, alikvótnych frekvencií).

Sila hlasu – Ide o implementáciu hudobných efektov ako je crescendo/decrescendo alebo efektov na ovplyvňovanie sily hlasu (f , mf , p^3 , prízvuky a i.). Keďže sínusový model je reprezentovaný vo frekvenčnej doméne možno dosiahnuť úpravu sily hlasu zmenou sklonu sínusoviek pomocou úpravy amplitúd.

Škálovanie dĺžky vokálneho traktu – Tento efekt bol objavený pri testovaní modelu, kedy sa zistilo, že pri prílišnom znížení hlasivkového tónu dochádza k výraznému zníženiu kvality syntetického spevu. Ukázalo sa ale, že túto degradáciu kvality možno zredukovať zdeformovaním spektrálnej obálky (zmenou amplitúd v každom okne) tak, že

$$\hat{H}(\omega) = H(\omega/\mu), \quad (2.3)$$

kde $H(\omega)$ je spektrálna obálka a μ je frekvenčný škálovací faktor, ktorý závisí od toho ako veľmi sa menila výška tónu. Typicky $0.75 < \mu < 1.0$. Takýto faktor pomáha odstrániť degradáciu kvality, ktorá bol spôsobená zlým rozložením formantov. Faktor $\mu > 1.0$ vie simulať detsky znejúci hlas.

2.1.3 Zhrnutie metódy

V tejto metóde sme zistili, že pre kvalitu spevu je podstatná aplikácia efektov, ktoré sme spomenuli v predchádzajúcej časti, a že ich aplikácia pri zvolení sínusového modelu je jednoduchá. Predstavili sme takisto spôsob výberu hlasových dát, ktorý teoreticky pokrýva všetky možné kombinácie, ktoré sa môžu vyskytnúť vo vstupnej piesni. Ak sa pozrieme na rozdelenie zo začiatku tejto kapitoly, môžme túto metódu zaradiť tak do jednej, ako aj do druhej skupiny, keďže sa sice neskladajú zvukové vzorky ale skladá sa ich ich parametrická reprezentácia.

³forte, mezzoforte, piano - hudobné termíny pre silno, stredne silno a slabo

Výhody

- Jednoduchý sínusový model
- Ľahká implementácia hudobných efektov
- Uspokojivá kvalita nasynetizovaného hlasu

Nevýhody

- Dostupnosť a získavanie vzoriek do databáze
- Obmedzené iba na charakteristiku hlasov v databáze

2.2 Syntéza spevu použitím neuronových sietí pre modelovanie vibráta

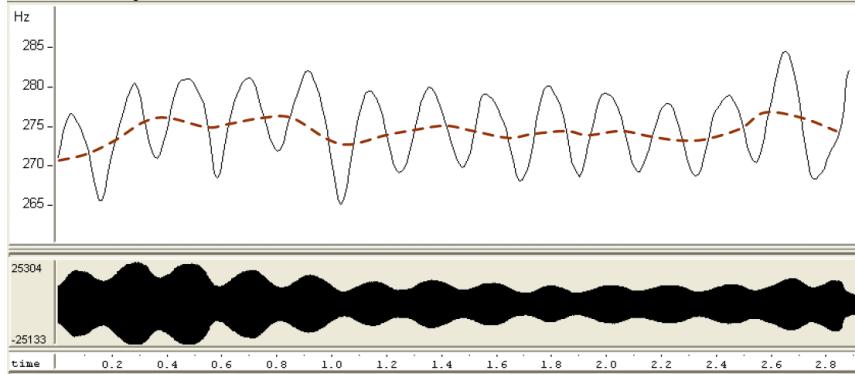
Jednou z hlavných motivácií tejto metódy je fakt, že vibráto je jedným z hlavných faktorom, ktorý robí spev dôveryhodným a podobným reálnemu spevu. Nevýhodou iných metód, ktoré v sebe zahrňovali aplikáciu efektu vibráta je, že jeho parametre boli zafixované, napriek tomu, že pri reálnom speve sa tieto parametre môžu v čase a v kontexte meniť. V tejto metóde sa upravuje vibráto pomocou parametrov, ktoré generuje neuronová sieť. Vysvetlime si podrobnejšie, čo je to vibráto a ako pracuje v tejto metóde neurónova sieť.[4]

2.2.1 Vibráto

Podľa hudobnej terminológie – vibráto – chvejivo; prirodzené kolísanie frekvencie a amplitúdy tónu pri speve alebo nástrojovej hre. Fyzikálne sa dá vibráto popísať ako kvázi pravidelná modulácia frekvencie. Ak chceme parametricky popísať vibráto pozrieme sa na tieto jeho vlastnosti: frekvencia opakovania a rozsah modulácie. Podľa uskutočnených výskumov sa frekvencia opakovania u školených spevákov pohybuje v rozmedzí 5-7 Hz a rozsah modulácie je medzi ± 50 až ± 150 centov (kde 1200 centov = 1 oktáva)[9].

Ukážka vibráta je na obr. 2.4, kde prerusovanou čiarou je označená výška tónu a plnou čiarou je aktuálna frekvencia (tón) spevu v danom čase. Na ukážke sa rozsah modulácie pohybuje medzi 265 až 285 Hz a frekvencia opakovania je okolo 5 Hz.

Jasne teda vieme vibráto popísať pomocou troch parametrov: **výška tónu** ($V_d(t)$), **frekvencia opakovania** ($V_e(t)$) a **rozsah vibráta** ($V_r(t)$).



Obrázok 2.4: Ukážka efektu vibráta

2.2.2 Zistovanie parametrov vibráta

Pri pohľade na obrázok 2.4 sa zamyslime ako je možné popísat frekvenciu f v čase t .

$$f(t) = V_d(t) + V_e(t) \cdot \cos(\phi(t)) \quad (2.4)$$

$$V_r(t) = \frac{1}{2\pi} \cdot \frac{d\phi(t)}{dt} \quad (2.5)$$

Vibráto sa teda prejavuje kmitaním okolo výšky tónu V_d s amplitúdou V_e a vo fáze ϕ . Frekvenciu opakovania V_r , ktorá sa získa podľa 2.5. Ako ale získať tieto parametre?

Pre získanie **výšky tónu** V_d sa použije jednoduchý priemerovací filter, kde výška tónu v čase t sa získa spriemerovaním frekvencií v časoch $t - 128, t - 127, \dots, t + 128$ (pre okienko veľkosti 512 a posune o 32 vzoriek).

Rozsah vibráta a frekvenciu opakovania sa získa tak, že sa najprv vyjadrí signál $s(t)$ ako $s(t) = V_e(t) \cdot \cos(\phi(t))$, čo sa získa z 2.4 ako $f(t) - V_d(t)$. Ďalej podľa definície Gabora analytický signál $z(t)$ prislúchajúci k $s(t)$ sa skladá z reálnej zložky $s(t)$ a imaginárnej zložky $\hat{s}(t)$ tak, že

$$z(t) = s(t) + i \cdot \hat{s}(t) \quad (2.6)$$

$$\hat{s}(t) = H[s(t)] \quad (2.7)$$

kde $H[s(t)]$ je Hilbertova transformácia (otáča vstupný signál o $\pi/2$ alebo $-\pi/2$), teda keďže $s(t) = V_e(t) \cdot \cos(\phi(t))$ bude $\hat{s}(t) = H[s(t)] = V_e(t) \cdot \sin(\phi(t))$ a $z(t) = V_e(t) \cdot \exp(i \cdot \phi(t))$.

Potom sa dá odvodiť $V_e(t)$ a $\phi(t)$ ako

$$V_e(t) = \sqrt{s^2(t) + \hat{s}^2(t)} \quad (2.8)$$

$$\phi(t) = \arctan(s(t), \hat{s}(t)) \quad (2.9)$$

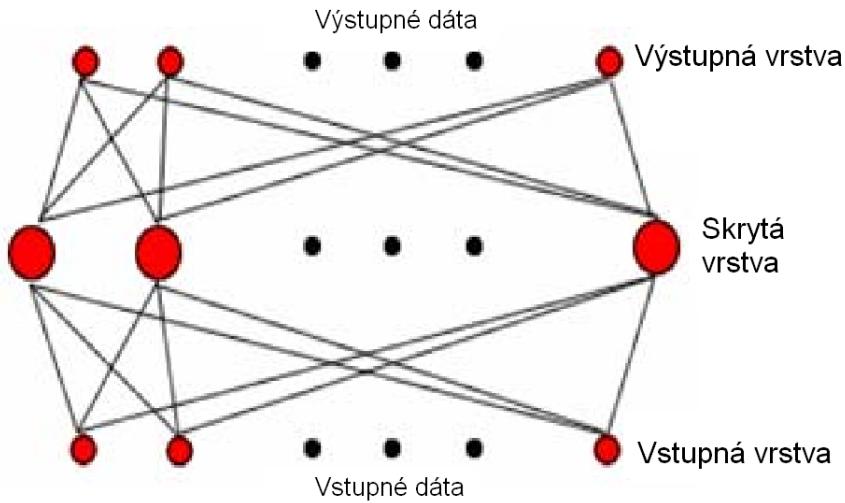
Parameter V_r vieme potom odvodiť podľa 2.5.

2.2.3 Neurónova sieť

V tejto metóde sa používa neurónovú sieť na zachytenie štýlu, akým spevák používa vibráto. Ako sme už spomenuli vibráto sa dá vyjadriť podľa troch parametrov $V_d(t)$, $V_e(t)$ a $V_r(t)$. Pridajme ešte jeden, ktorý popisuje počiatok fázu vibráta $\phi(0)$. Pre každý z týchto parametrov sa zostrojí neurónovú sieť. Algoritmom učenia siete je zvolená spätná propagácia.

Štruktúra neurónovej siete

Štruktúra takýchto neurónových sietí je na obrázku 2.5.



Obrázok 2.5: Štruktúra neurónovej siete pre parametre vibráta[4]

Do vstupnej vrstvy vstupuje kontextová informácia o spievanej slabike. Skrytú vrstvu tvorí 8 vrcholov (tentototo počet sa ukázal ako najlepšia voľba). Z každého vrcholu vstupnej

vrstvy je spojenie do každého vrchola skrytej vrstvy. Výstupnú vrstvu tvorí 32 vrcholov pre parametre $V_d(t)$, $V_e(t)$ a $V_r(t)$. Pre $\phi(0)$ stačí jeden vrchol.

Vstupná vrstva

Povedali sme, že vstupom do tejto vrstvy je informácie o spievanej slabike. Čo je ale z pohľadu vibráta dôležitá informácia o slabike? V tejto metóde sa za také vlastnosti pokladajú **dĺžka aktuálnej slabiky, začiatočný a konečný typ aktuálnej slabiky** (aký začiatok resp. zakončenie má slabika - napr. „m“, „r“, resp. „a“, „-ou“), **dĺžka a konečný typ predchádzajúcej slabiky, dĺžka a konečný typ nasledujúcej slabiky a rozdiely vo výške tónu medzi aktuálnym a predchádzajúcim resp. nasledujúcim tónom.**

Ako vzorka sa v [4] použilo 15 rôznych piesní v mandarínskej čínštine v rôznych tem- pách, ktoré dohromady obsahovali 2841 slabík. Je vidno, že všetkých možných kombinácií parametrov by bolo príliš veľa, a preto sa na ich zredukovanie použilo zaradenie slabík do tried:

Pre aktuálnu slabiku – 5 tried podľa trvania ($0-0.3/0.3-0.5s/0.5-0.8s/0.8-1.3/>1.3s$), 3 triedy trvania pre predchadzajúcu a nasledujúcu slabiku ($0-0.25s/0.25-0.5s/>0.5s$), ďalej 3 triedy pre začiatočný typ slabiky a 4 triedy pre konečný typ slabiky (jednosamohláskové (a)/difóny (ai)/trifóny(iau)/nosové (ang)) a nakoniec 7 tried bolo zvolených pre vyjadrenie rozdielu medzi tónmi ($<-6/-5,-4,-3/-1,-2/0/1,2/+3,+4,+5/>+6$ potónov).

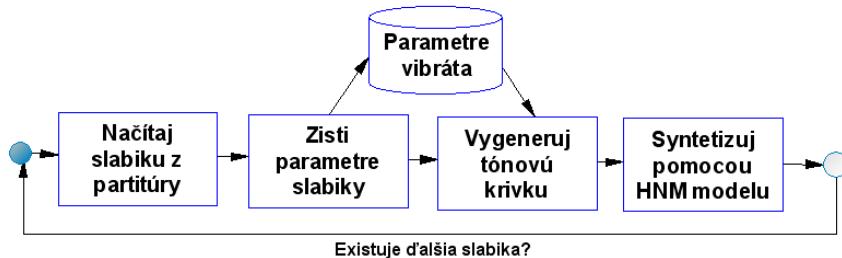
Výstupná vrstva

Povedali sme, že výstupná vrstva sa skladá z 32 vrcholov. Ako ale získať daný parameter? Parameter vibráta (všeobecne V_x) je „rozdelený“ do všetkých vrcholov výstupnej vrstvy ($U_x(i)$) tak, že $U_x(i) = V_x(T.i/31)$, $i = 0, 1 \dots, 31$, kde T označuje časovú dĺžku. Čiže každý vrchol zodpovedá časovému úseku. Výsledný parameter v čase t sa potom nájde ako lineárna interpolácia časovo najbližších uzlov: Nájde sa také k , aby sa v intervale $[T_k, T_{k+1})$ nachádzalo t , pričom $T_i = T.i/31$. Parameter V_x ($x \in \{d, e, r\}$) sa potom získa ako

$$V_x(t) = U_x(k) + (U_x(k+1) - U_x(k)) \cdot \frac{t - T_k}{T_{k+1} - T_k} \quad (2.10)$$

2.2.4 Priebeh syntézy

Ked' je neurónová sieť natrénovaná, priebeh syntézy je celkom jednoduchý. V zdrojovom súbore s partitúrou sa nájde slabika. Zistí sa dĺžka slabiky a jej ďalšie parametre. Parametre sa dajú na vstup neurónovej sieti, ktorá vráti parametre vibráta. Pomocou dĺžky slabiky sa vytvorí tónová krivka, na ktorú sa aplikujú parametre, ktoré vrátila neurónová sieť. Táto krivka sa pošle ako vstup pre HNM model⁴. Ak je v partitúre ešte ďalšia slabika, algoritmus sa vráti na začiatok a pokračuje s touto slabikou. Priebeh ukazuje obr. 2.6



Obrázok 2.6: Priebeh syntézy spevu s použitím neurónovej siete pri modelovaní vibráta

2.2.5 Zhrnutie metódy

V tejto metóde sme si predstavili neodmysliteľný parameter spevu – vibráto. Zistili sme ako sa dá vibráto parametrizovať a ako tieto parametre získať z nahrávky spevu. Bola predstavená neurónová sieť, ktorá sa dá podľa parametrov o slabikách z piesne natrénovať tak, aby vedela následne generovať parametre vibráta pri syntéze. Tieto parametre zaručujú vyššiu naturálnosť spevu, keďže sú inteligentne generované a nie sú zafixované.

Výhody

- Lepšia kvalita ako pri zafixovaných parametroch vibráta
- Predstavený model pre generovanie parametrov vibráta

Nevýhody

- Príliš malá vzorka pre trénovanie
- Trénovanie iba vzorkami od jednej osoby

⁴Harmonic + noise model (viď [8])

2.3 Syntéza spevu pomocou metódy samplingu

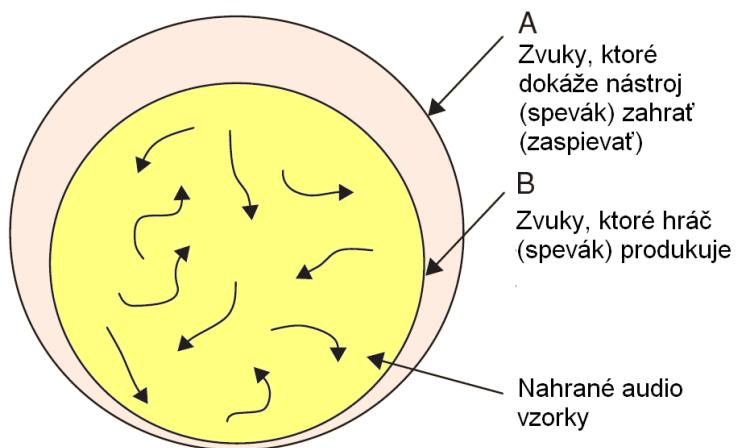
Táto metóda syntézy spevu je založená na spájaní vzoriek spevu z databázy. Výhodou takejto metódy je vysoká naturálnosť spevu, keďže sa v nej spájajú reálne naspevane vzorky.

Je ale jasné, že pre zachytenie celého spektra možných požiadaviek na spev je metóda spájania vzoriek málo flexibilná, keďže máme len obmedzene veľkú databázu vzoriek. Problém nastáva tiež pri spájaní vzoriek, keď slabika vybraná do syntézy je vzdialená od predchádzajúcej a to vo viacerých ohľadoch – foneticky (je z iného slova), tónovo, tempom a i.

Aj keď sa táto metóda používa s úspechom pre syntetizáciu hudobných nástrojov, ktorých spektrum je oproti hlasu oveľa obmedzenejšie, pre hlas existuje oveľa viac parametrov, ktoré je potrebné pokryť. Metóda [3] navrhuje, ako by sa dalo s týmito problémami vysporiadať.

2.3.1 Zvukový priestor

Pozrime sa ako vyzerá priestor zvukov, ktoré je možné zaspievať resp. zahrať na hudobnom nástroji (obr. 2.7).



Obrázok 2.7: Zvukový priestor[3]

Priestor A predstavuje všetky možné zvuky, ktoré možno z nástroja (speváka) vylúdiť.

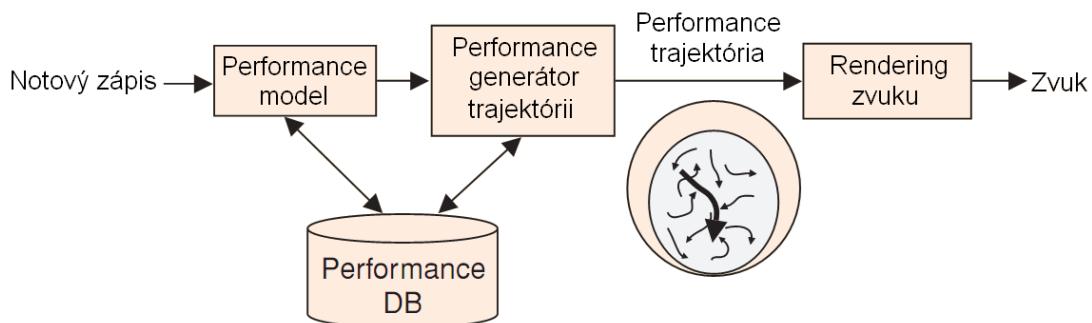
Podpriestor B priestoru A predstavuje také zvuky, ktoré sa bežne vyskytujú v hre či speve. Trajektórie v priestore B reprezentujú nahraté vzorky.

Koľko rozmerný je tento priestor? Zrejme má nekonečne veľa rozmerov (tempo, výška tónu, farba tónu, trvanie, ...). Je teda nevyhnutné sa pokúsiť o jeho approximáciu do priestoru s konečným počtom rozmerov. Za rozmery treba zrejme voliť také parametre, ktoré sú charakteristické pre daný hudobný nástroj či speváka. Trajektórie B teda vlastne vyjadrujú pohyb v takomto priestore.

Problém ale nastáva, keď je potrebná vzorka z priestoru B , ktorá sa v ňom nenachádza.

2.3.2 Komponenty syntézy

Komponenty zúčastňujúce sa syntézy a jej priebeh je na obr. 2.8. Skôr ako popíšeme priebeh syntézy pozrime sa na jej komponenty.



Obrázok 2.8: Priebeh syntézy samplingom[3]

Performance model

Zachytáva také charakteristiky, ktoré sú zadané v notovom zápise a sú pre spev dôležité. Medzi hlavné takéto charakteristiky patrí **tempo**, **zmena dĺžky noty v porovnaní so štandardným trvaním** (napr. v 4/4-takte je štandardná dĺžka noty 1 doba), **výskyt vibráta, hlasitosti**, spôsob **nasadenia a spájania** nôt (prízvuk, legato, staccato⁵).

⁵legato, staccato - hudobné termíny - viazane resp. oddelene

Performance databáza

Rozdeľuje zvukový priestor do troch podpriestorov (*A*, *B* a *C*). *A* pozostáva z takých vzoriek, ktoré sú nahraté, prípadne, ktoré vznikli transformáciou (zmenou tempa, výšky tónu a pod.) a skladaním pri syntéze. *B* a *C* obsahujú také vzorky, ktoré vzniknú transformáciou vzoriek z *A* a vyjadrujú ich špecifické podanie.

A je 4-rozmerný a obsahuje parametre: **tempo** (v BPM⁶), **výška tónu** (frekvencia), **hlasitosť** (naškálovaná do intervalu 0...1 a do 4 tried - veľmi jemne (0.25), jemne (0.5), normálne (0.75) a hlasno(1)) a **fonetický parameter** (pokrýva možné alofóny). Ukážme ako nahrať vzorky do priestoru *A*:

Najprv sa nájdú rôzne alofóny. Keďže za akceptovateľné sa pokladajú transpozície tónov o max. ± 6 poltónov, nasnieva sa každá alofóna v troch rôznych výškach tónu, tak aby sa pokryl celý spevácky rozsah a to v rôznych tempách a pri rôznych hlasitostiach.

B je jednorozmerný a vyjadruje parametre vibráta. *B* neobsahuje vzorky ale šablóny pre úpravu rôznych stupňov vibráta, ktoré boli získane analýzou nahrávok.

C je takisto jednorozmerný a nastavuje parametre artikulácie (napr. nasadenie, viazanie, odsadenie nôt). Rovnako neobsahuje vzorky priamo, ale ich šablóny.

Generátor performance trajektórii

Slúži na aplikovanie charakteristík zachytených Performance modelom. Teda aby ich bolo možné aplikovať, treba na vzorku z *A* aplikovať šablóny z *B* a *C*. Teda generátor performance trajektórii vygeneruje vzorku, ktorá spĺňa príslušnú požiadavku a je na správnom mieste v zvukovom priestore (napr. normálne hlasná slabika „na“ v tempe 120 s miernym vibrátom v trvaní 0.25s).

Rendering zvuku

Slúži na plynulé spájanie jednotlivých vzoriek. Vstupom sú performance trajektórie (nahraté vzorky). Aj keď podľa predstaveného modelu by malo byť možné spájať vzorky bez problémov, v praxi to kvôli zanedbaniu niektorých parametrov (napr. rozlíšenie vzoriek s/bez nádychu) nejde. Preto za účelom plynulého spojenia vzoriek sa vyrátajú rozdielny

⁶beats per minute – počet úderov za minútu

medzi vzorkami v bode spojenia pre rôzne parametre hlasu a okolité sekcie sa patrične upravia.

2.3.3 Priebeh syntézy

Zo zdrojového notového zápisu sa vyberie nota, jej parametre sa pošlú performance modelu, ktorý nájde v performance databáze najbližšiu vzorku (ktorá sa najmenej lísi v parametroch od aktuálnej) a túto posunie performance generátoru trajektórii, ktorý transformuje túto vzorku aby zodpovedala požiadavkám. Táto vzorka je poslaná do renderingu zvuku, ktorý túto vzorku zviaže s predchádzajúcou.

2.3.4 Zhrnutie metódy

Predstavili sme metódu, ktorá narozenie od doteraz spomenutých pracuje so vzorkami reálneho spevu. Ukázalo sa však, že sa nedá pokryť celý zvukový priestor pre spev a teda je potrebné zaviesť parametrizáciu priestoru a metódu na transformáciu vzoriek z databázy na vzorky, ktoré v databáze nemáme. Popísali sme tiež metódu budovania databázy vzoriek a konečné spájanie vzoriek do výsledného zvuku.

Výhody

- Výborná kvalita výsledku
- Spojenie samplingu a parametrického popisu

Nevýhody

- Náročný a zdĺhavý zber vzoriek
- Nedokonalý model zvukového priestoru

2.4 Syntéza prevodom reči na spev pomocou jedinečných akustických znakov spevu

Je jednou z mála metód[13], ktoré sa venujú prevodu reči na spev. Pozostáva z niekoľkých komponentov, ktoré sa zúčastňujú transformácie reči na spev: **Model pre F_0** , **Model pre úpravu trvania** a **Model spektra**.

2.4.1 Komponenty syntézy

Model pre F_0

Ked' sa zamyslíme nad hlasíkovým tónom F_0 , tak pre spev ho vieme presne určiť, pretože zodpovedá výške tónu aktuálnej noty. Zahoďme teda krivku F_0 (krivka frekvencií hlasíkového tónu v čase) pre reč a nahradíme ju krivkou, ktorú zostavíme podľa výšky tónov vo vstupnom notovom zápise. Ďalej krivku F_0 ovplyvňujú znaky spevu:

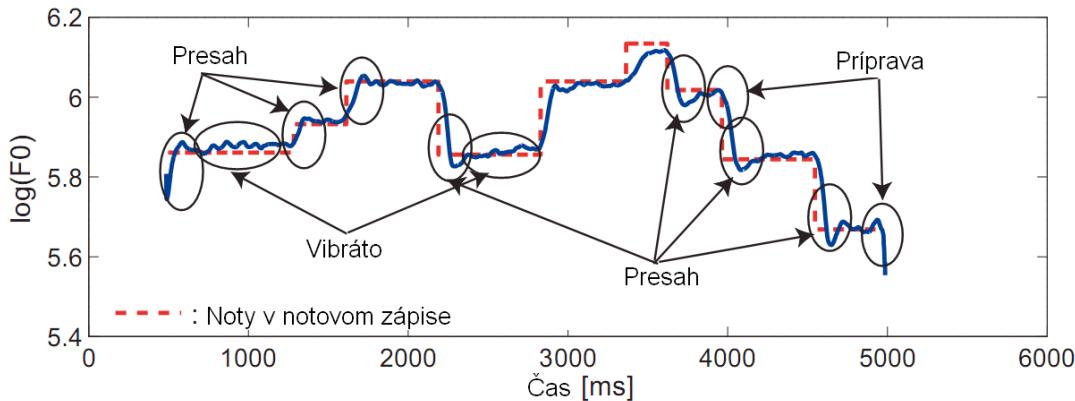
Presah – prekročenie cieľovej noty hned' po zmene noty

Vibráto – kvázi-periodická modulácia frekvencie medzi 5-8Hz

Príprava – pokles/nárast cieľovej noty, ak ďalšia nota stúpa resp. klesá, tesne pred zmenou noty

Jemné kolísanie – nepravidelné výkyvy frekvencie nižšie ako 10Hz,

z ktorých sme predstavili zatial iba vibráto. Tieto efekty ukazuje obr. 2.9.



Obrázok 2.9: Znaky spevu[13]

Presah, vibráto a prípravu môžeme vyjadriť pomocou prenosovej funkcie (viď [5]) danej ako

$$H(s) = \frac{k}{s^2 + 2\zeta\omega s + \omega^2}, \quad (2.11)$$

kde ω je základná frekvencia, ζ je koeficient tlmenia a k je koeficient prírastku systému. Potom odozva signálu sa získa ako

$$h(t) = \begin{cases} \frac{k}{2\sqrt{\zeta^2-1}}(\exp(\lambda_1\omega t) - \exp(\lambda_2\omega t)) & |\zeta| > 1 \\ \frac{k}{\sqrt{1-\zeta^2}} \exp(-\zeta\omega t) \sin(\sqrt{1-\zeta^2}\omega t) & 0 < |\zeta| < 1 \\ kt \exp(-\omega t) & |\zeta| = 1 \\ \frac{k}{\omega} \sin(\omega t) & |\zeta| = 0 \end{cases} \quad \begin{array}{l} \text{Presah, Príprava} \\ \text{Vibráto} \end{array} \quad (2.12)$$

kde charakteristika každého z týchto efektov je riadená pomocou parametrov ω , ζ a k . Pomocou metódy najmenších štvorcov pri porovnávaní vygenerovanej a skutočnej krivky F_0 sa určili parametre $(\omega/\zeta/k)$ pre presah – $(0.0348/0.5422/0.0348)$, vibráto – $(0.0345/0/0.0018)$ a prípravu – $(0.0292/0.6681/0.0292)$.

Efekt jemného kolísania sa získa vygenerovaním bieleho šumu⁷, ktorý sa preženie cez dolnopriepustný filter s orezávaciu frekvenciou 10Hz, ktorého amplitúda bola normalizovaná na max. 5hz. Takto spracovaný signál sa pridá k vygenerovanej krivke F_0 .

Model pre úpravu trvania

Z notového zápisu podľa tempa a typu noty zistí trvanie noty. Rozsegmentuje slabiku pri slúchajúcu k danej note podľa foném a natiahne slabiku tak, aby jej dĺžka sedela s trvaním noty. Zároveň sa určí hraničná oblasť (pri prechode medzi spoluhláskou a samohláskou), ktorá sa nemení. Jej dĺžka sa experimentálne určila na 40ms, kde 10ms je pred hranicou a 30ms je za hranicou.

Hraničná oblasť sa nepredlžuje.

Spoluhláska sa konštantne predĺži podľa experimentálne zistenej konštanty (**1.58 pre frikatíva** tj. trené spoluhlásky - napr. f, s, z; **1.13 pre explozíva** tj. výbušné spoluhlásky (napr. p, t, č, b, g) a „j“; a **1.77 pre nosové spoluhlásky** - napr. m,n)

Samohlásky sa potom predĺžia tak, aby súčet trvaní samohlásky, hraničnej časti a spoluhlások dával dokopy dĺžku noty.

⁷Ide náhodnú zmenu tlaku v čase, ktorá, ak rátame s neobmedzene spojitým priestorom, má navyše zhodne vysokú energiu vo všetkých frekvenčných pásmach[1]

Model spektra

Pridáva do spektrálnej obálky znaky špecifické pre spev. Je to tzv. **spevácky formant**, ktorý sa objavuje ako výraznejší vrchol v spektrálnej obálke spevu a leží v okolí 3kHz a je pridávaný ako

$$S_{sg}(f) = W_{sf}(f)S_{sp}(f) \quad (2.13)$$

kde $S_{sp}(f)$ a $S_{sg}(f)$ sú spektrálne obálky reči a spevu a $W_{sf}(f)$ je váhovacia funkcia, ktorá zvýrazňuje spevácky formant v rečovom spektre $S_{sp}(f)$ daná ako

$$W_{sf}(f) = \begin{cases} (1 + k_{sf})(1 - \cos(2\pi \frac{f}{F_b+1})) & \text{ak } |f - F_s| \leq \frac{F_b}{2} \\ 1 & \text{inak} \end{cases} \quad (2.14)$$

kde F_s je frekvencia takého vrcholu v $S_{sp}(f)$ blízkeho 3kHz, F_b je rozsah zvýraznenia (koľko frekvencií v okolí vrchola sa zvýrazní) a k_{sf} je koeficient, ktorý ovplyvňuje mieru zvýraznenia.

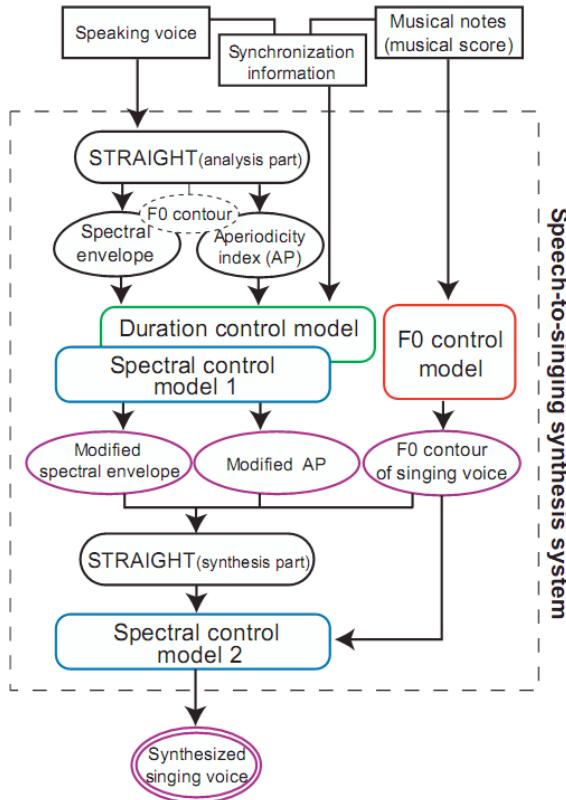
Ďalším znakom, ktorý spektrálny model pridá je modulácia amplitúdy pri spievaní vibráta. Iste nie je prekvapivé, že časti spevu počas vibráta, ktoré sú položené vyššie sa spievajú silnejšie ako tie, ktoré sa spievajú nižšie. Mieru zvýraznenia týchto častí určuje

$$E_{sg}(t) = (1 + k_{am} \sin(2\pi f_{am}t))E_{sp}(t) \quad (2.15)$$

kde $E_{sp}(f)$ a $E_{sg}(f)$ sú obálky amplitúd (energie) reči a spevu, f_{am} je miera modulácie a k_{am} je koeficient jej príspevku.

2.4.2 Priebeh syntézy

Je ukázaný na obr. 2.10. V tejto metóde je použitý na analýzu a syntézu systém STRAIGHT predstavený v [2]. Nahrávka reči sa posunie na analýzu systému STRAIGHT a notový zápis sa pošle do Modelu pre F_0 . STRAIGHT zistí spektrálnu obálku, rečovú krivku F_0 . Tieto parametre prejdú do modelu trvania a aplikuje sa na ne tá metóda z modelu spektra, ktorá zvýrazňuje spevácky formant. Modifikovaná spektrálna obálka a spevová krivka F_0 vygenerovaná modelom pre F_0 sa dajú ako vstup do syntetizačnej časti STRAIGHT. Nakoniec sa použije metóda modifikácie amplitúdy pri vibráte zo spektrálneho modelu.



Obrázok 2.10: Priebeh syntézy[13]

2.4.3 Zhrnutie metódy

Ukázali sme metódu, ktorá ako jedna z mála konvertuje reč na spev. Využíva na to 3 základné komponenty, ktoré generujú výslednú melódiu (krivku F_0) a aplikujú na ňu efekty, z ktorých sme si tu ukázali prípravu, presah, vibráto a jemné kolísanie. Metóda ďalej aplikuje dva znaky, ktoré odlišujú reč od spevu – spevácky formant a zmenu energii pri spievaní vibráta. Metóda síce nepoužíva vzorky aj keď spektrum pre fonémy do výsledného spevu sa získa z nahrávky. Napriek tomu sú výsledky veľmi dobré.

Výhody

- Vysoká kvalita vygenerovaného spevu

Nevýhody

- Nedostupnosť systému STRAIGHT
- Nezahŕňa efekty na zmenu hlasitosti

Kapitola 3

Návrh a implementácia

V tejto kapitole rozoberieme ako sme postupovali v návrhu aplikácie, aké techniky a algoritmy sme použili a ako vyzerala implementácia. Kapitola je členená na sekcie, ktoré ukazujú kľúčové časti práce a záverečná kapitola ukazuje ako tieto časti spájame do výslednej aplikácie.

3.1 Reprezentácia piesne

Pri návrhu toho ako by sme mohli reprezentovať pieseň som vychádzal z jej vlastností, ktoré sú pre budúcu syntézu dôležité. Popíšme si, ktoré to sú a ako ich budeme reprezentovať:

tempo - udáva ako dlho trvajú jednotlivé hudobné útvary (noty, pauzy), v hudbe sa zvykne zadávať buď slovne napr. *andante*, *moderato* alebo ako počet úderov za minútu pre príslušnú notu. Napr. zápis $\text{J} = 120$ udáva, že štvrtová nota do jednej minúty vojde 120 krát. Hoci je možné v priebehu času tempo meniť, v našom návrhu považujeme tempo za konštantné¹. Tempo budeme reprezentovať ako počet úderov za štvrtovú notu.

výška tónu - udáva v hudobnej terminológii ako vysoko zaspievať resp. zahrať daný tón a určuje sa odčítaním z notovej osnovy. Výška každého tónu má však takisto svoje slovné pomenovanie, napr. komorné *A* sa označuje ako *A4* a zodpovedá frekvencii 440Hz. Výšku tónu budeme reprezentovať jeho slovným pomenovaním.

¹V skutočnosti nejde o obmedzenie, nakoľko môžeme prerátať dĺžku nôt v novom tempe do starého tempa

dĺžka tónu/pauzy - udáva ako dlho spievať daný tón resp. čakať. Odčítava sa z notovej osnovy a určuje sa podľa typu noty, ktorá tón reprezentuje. Udáva sa v tzv. dobách. Trvanie jednej doby je určené ako trvanie štvrtovej noty. Hovoríme preto, že štvrtová nota trvá 1 dobu, pôlová 2 doby a celá štyri doby. Dĺžku tónov a páuz budeme reprezentovať ako počet dôb, ktorý daný tón/pauza trvá.

text - určuje akú slabiku spievať na určitý tón. Pre každý tón budeme reprezentovať slabiku jej fonetickým prepisom.

3.1.1 Súbor s piesňou

Ukážme si ako teda vyzerá súbor, ktorým reprezentujeme pieseň. Na prvom riadku je povinne udané tempo (TEMPO) ako celé číslo. Na ďalších riadkoch sú popísané tóny alebo pauzy, pričom v súbore sa musí aspoň raz vyskytnúť popis tónu.

Tón popisujeme ako trojicu *meno dĺzka slabika*:

meno popisuje výšku tónu a je v tvare $\{C, D, E, F, G, A, B\}[1]\{\#, \%\}[0 - 1]\{1 - 9\}[1]$, teda napr. *C4* alebo *F#5*, pričom číslo určuje poradie oktavy a $\#$ resp. $\%$ značí tón o poltóna zvýšený resp. znížený

dĺzka určuje počet dôb a ide o desatinné číslo

slabika definuje foneticky prepis slabiky spievanej na danom tóne v tvare

-fonema₁-fonema₂-fonema₃-...-fonema_N (napr. *h/-U-b*) ak je daná slabika celé slovo, v tvare

-fonema₁-fonema₂-fonema₃-...-fonema_N (napr. *-a:-r*) ak slabika je pokračovaním slova a zároveň v tejto slabike slovo končí a v tvare

-fonema₁-fonema₂-fonema₃-...-fonema_{N-} (napr. *-E-s-*) ak je slabika uprostred slova.

Pauzy majú tvar *P n* (napr. *P 2*), kde *P* je kľúčové slovo pre pauzu a *n* je desatinné číslo, ktoré vyjadruje počet dôb, ktoré pauza trvá.

Štruktúra súboru je ukázaná na obrázku 3.1 a konkrétny súbor s popevkami „doremi“ a tempom 120 je ukázaný na obrázku 3.2.

```

TEMPO
TONE_1 LENGTH_1 SYLLABLE_1
TONE_2 LENGTH_2 SYLLABLE_2
...
P PAUSE_LENGTH_1
TONE_N LENGTH_N SYLLABLE_N
...

```

Obrázok 3.1: Ukážka štruktúry súboru s piesňou

```

120
C4 1 d-0
P 1
D4 1 r-E
P 1
E4 1 m-I

```

Obrázok 3.2: Ukážka súboru *doremi.song* v tempe 120 a s pauzou po každom tóne

3.2 Určenie melódie

Stanovili sme si ako budeme reprezentovať vstupný súbor s piesňou. Ďalším dôležitým krokom je z nej vyextrahovať melódiu. Ako melódiu si zadefinujme funkciu frekvencie od času. Melódia teda určuje, akou frekvenciou sa spieva v určitom čase. Krivka, ktorá určuje melódiu nazývame krivka hlasíkového tónu (F_0). Ako ale vytvoriť túto krivku? Pozrime sa na vstupný súbor. Poznáme tempo, výšky tónov a dĺžky páuz. Náš algoritmus bude teda vyzerať nasledovne:

- 1. Zisti dĺžku doby** - Keďže tempo udáva koľko dôb (štvrťových nôt) sa zmestí do jednej minúty môžeme ľahko zistiť dĺžku jednej doby T_{doba} v sekundách ako

$$T_{doba} = 60/\text{tempo} \quad (3.1)$$

- 2. Prejdi všetky tóny a pauzy, zisti ich dĺžku a frekvenciu** - nakoľko poznáme dĺžku 1 doby nie je problém zistiť dĺžku tónu alebo pauzy. Ako ale určíme ich frekvenciu? Poznáme výšku tónu podľa pomenovania. Zároveň vieme, že tón τ a tón zvýšený o oktávu τ_{+1} majú takú vlastnosť, že frekvencia zvýšeného tónu je dvojnásobkom pôvodného tónu a zároveň tón o oktávu znížený τ_{-1} má polovičnú frekvenciu

pôvodného.

$$\begin{aligned} f(\tau_{+1}) &= 2f(\tau) \\ f(\tau_{-1}) &= \frac{f(\tau)}{2} \end{aligned} \tag{3.2}$$

Je zrejmé, že ak poznáme frekvencie tónov v niektornej oktáve vieme vypočítať tóny v akejkoľkovek oktáve. Oktáva, ktorá nám bude slúžiť ako predloha je oktáva, ktorá obsahuje komorné A označené ako $A4$.

V skutočnosti ale netreba poznať presné frekvencie tónov v tejto oktáve ale len počet poltónov, o ktoré sú vzdialené od $A4$. Tóny v 4. oktáve a ich vzdialenosť od $A4$ možno vidieť v tabuľke 3.1.

Názov tónu	Oktáva	Vzdialosť od $A4$
C	4	-9
C# (D%)	4	-8
D	4	-7
D# (E%)	4	-6
E	4	-5
F	4	-4
F# (G%)	4	-3
G	4	-2
G# (A%)	4	-1
A	4	0
A# (B%)	4	+1
B	4	+2

Tabuľka 3.1: Tóny v 4. oktáve a ich vzdialenosť od $A4$

Vidíme, že oktáva teda obsahuje 12 poltónov², preto môžeme vypočítať frekvenciu akéhokoľvek tónu ako

$$f = 2^{n/12} \cdot 440 \tag{3.3}$$

kde n je „počet“ poltónov od $A4$ po daný tón a to taký, že ak je tón nižší je táto vzdialenosť záporná. Napr. frekvenciu tónu $C\#5$ zistíme takto: $f_{C\#5} = 2^{4/12} \cdot 440 \approx 554.37\text{Hz}$, nakoľko $C\#5$ je vzdialený 4 poltóny od $A4$.

Pozrime sa na všeobecný prípad pre tón v tvare $menoM$ (napr. $F3$), kde *meno* je názov tónu (*meno* $\in \{\text{"C"}, \text{"C\#"}, \text{"D"}, \text{"D\#"}, \dots\}$) a M je číslo oktavy:

²poltóny sú od seba zvukovo rovnako vzdialené, ale keďže ucho vníma frekvenciu logaritmicky, ležia poltóny na exponenciálnej krivke, viď [15]

Nech n je vzdialenosť, ktorú odčítame z tabuľky 3.1 podľa riadku, ktorý obsahuje meno a nech $k = M - 4$ je rozdiel medzi číslom oktávy tónu a číslom referenčnej oktávy. Potom frekvenciu tónu $menoM$ zistíme podľa:

$$f(menoM) = 2^{12 \cdot k + n/12} \cdot 440 \quad (3.4)$$

3. **Zostroj krivku hlasíkového tónu** - Máme vypočítané frekvencie, trvania tónov a páuz, teraz už nie je problém zostrojiť krivku hlasíkového tónu. Jedinú vec, ktorú treba mať na zreteli je, že sa pohybujeme v diskrétnom signáli, a tak budeme musieť krivku navzorkovať.

Algorithm 1 Zostrojenie krivky hlasíkového tónu

```

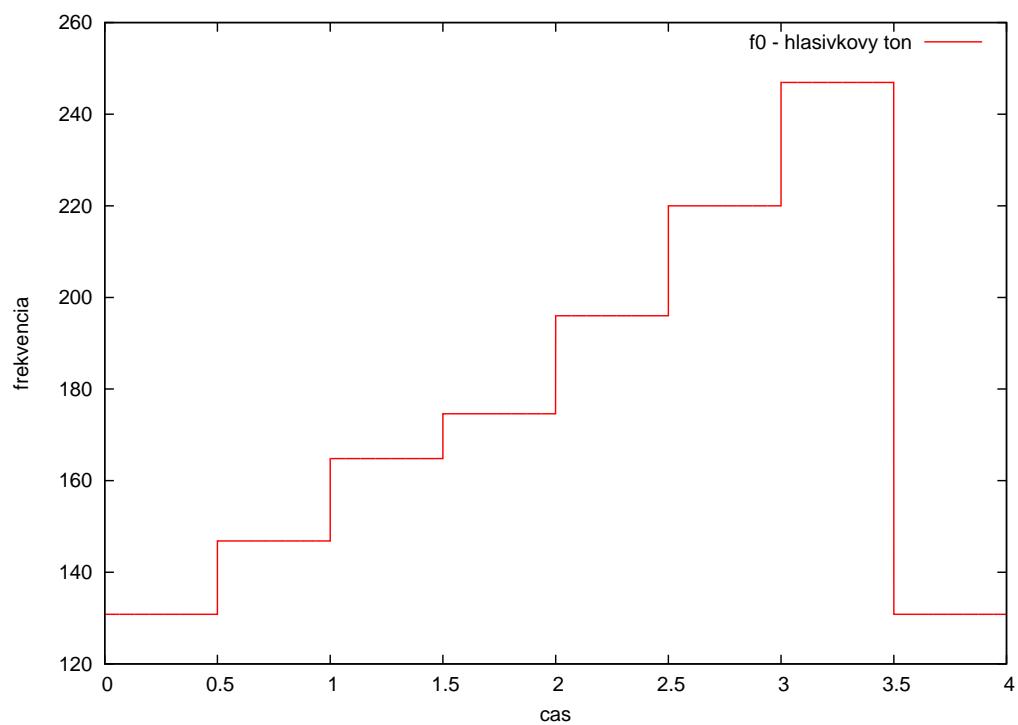
{vstup: vzorkovacia frekvencia, piesen vystup: krivka}
cas ← 0
cislo_vzorky ← 0
for all tony a pauzy do
    prvok ← aktualny ton alebo pauza
    if prvok JE ton then
        frekvencia ← prvok.frekvencia {Zisti sa podla bodu 2}
    else
        frekvencia ← 0 {je to pauza}
    end if

    nove_cislo_vzorky ← cislo_vzorky + prvok.trvanie * vzork. frek.
    for i = cislo_vzorky to nove_cislo_vzorky do
        krivka[i] ← frekvencia {Pre cele trvanie tonu/pauzy nastav frekvenciu}
    end for

    cas ← cas + prvok.trvanie {Zisti sa podla bodu 1,2}
    cislo_vzorky ← nove_cislo_vzorky
end for
return krivka

```

Ukážku krivky hlasíkového tónu možno vidieť na obrázku 3.3.



Obrázok 3.3: Ukážka krivky hlasivkového tónu

3.3 Tvarovanie krivky hlasivkového tónu F_0

V predchádzajúcej časti sme vytvorili základnú krivku hlasivkového tónu len na základe notového zápisu. Zatiaľ sme sa nezaujímali ako vyzerá krivka hlasivkového tónu (krivka F_0) u spevákov.

Ako sme spomínali v druhej kapitole, tvarovanie hlasivkového tónu sa ukázalo ako dôležité pre zvýšenie kvality výslednej syntézy. Znaky ktoré sme implementovali sú vibráto, jemné kolísanie hlasu, prípravu a vyskúšali sme aj priemerovací filter. Pre zvukovú vizualizáciu krivky F_0 sme tiež zstrojili jej syntetizér.

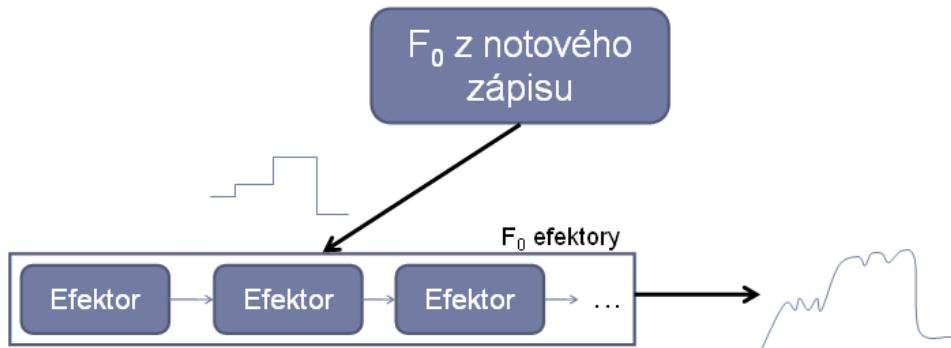
Popíšme, aký mechanizmus používame na tvarovanie krivky F_0 .

3.3.1 F_0 Efektor

Je nadriedou tried, ktoré budeme používať na postupné tvarovanie krivky F_0 . Ako vstup do takejto triedy je nejaká krivka F_0 a výstupom je iná pozmenená krivka F_0 , ktorá môže byť vstupom do ďalšej triedy atď. Takéto odvodené triedy budeme ďalej nazývať F_0 efektory.

Výhodou tohto prístupu je možnosť pospájať F_0 efektory za sebou v ľubovoľnom poradí a tiež priamočiarosť a prehľadnosť takéhoto riešenia.

Budeme mať teda F_0 Efektor pre aplikovanie každého znaku (napr. vibráto, kolísanie). Schému použitia reťazenia F_0 efektorov znázorňuje obrázok 3.4



Obrázok 3.4: Reťazenie F_0 efektorov

3.3.2 Vibráto

Vibráto je najdôležitejším znakom, ktorým sa vyznačuje ľudský spev. Dokonca čím je hlas viac trénovaný, tým viac pravidelnejšie a výraznejšie vibráto v ňom môžeme pozorovať. Zopakujme, že vibráto je kvázi-periodická modulácia frekvencie medzi 5-8Hz kde sa rozsah modulácie pohybuje od ± 50 až ± 150 centov (kde 1200 centov ≈ 1 oktáva).

Aby sme mohli implementovať efektor pre vibráto potrebujeme získať parametre. Budú to frekvencia modulácie f_m a jej rozsah R . Parameter f_m sme predvolili na 7Hz a rozsah modulácie R sme určili na ± 60 centov.

Vibráto vytvoríme pomocou sínusovky s frekvenciou f_m a amplitúdou $A(t)$, ktorú pripočítame k pôvodnej krivke F_0 .

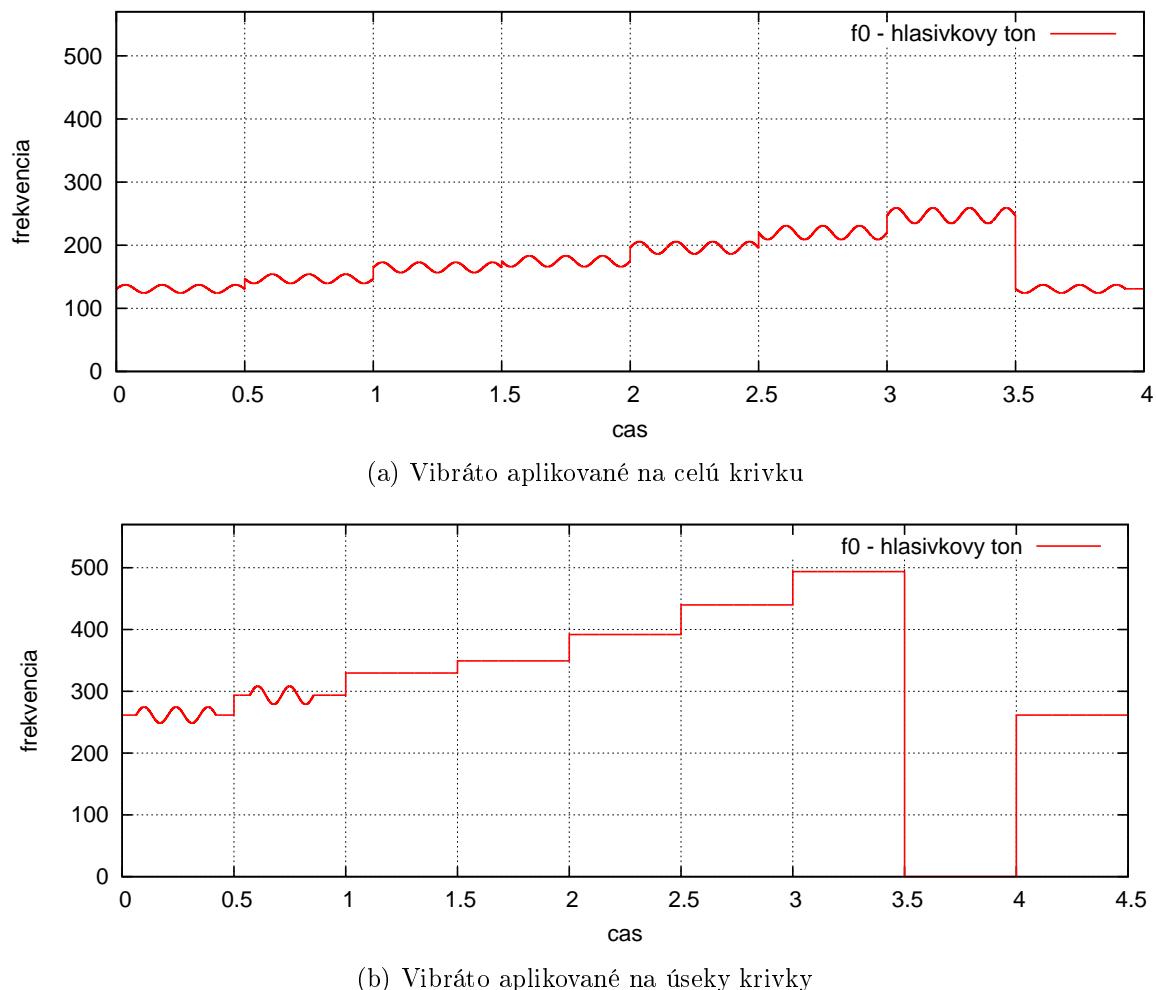
Amplitúda $A(t)$ závisí od rozsahu modulácie R , ale pre rôzne vysoké tóny (s rôznou frekvenciou) bude rôzna. Súvisí to s tým, že pri vyšších tonoch sú frekvenčné rozdiely medzi tónmi vyššie. $A(t)$ predstavuje priemerný frekvenčný rozdiel medzi R danými centami v danej oktáve. Ak napríklad rátame $A(t)$ pri pôvodnej frekvencii 440 Hz (A4), vieme, že tón A v ďalšej oktáve má frekvenciu $A5 = 880$ Hz. Jedna oktáva má teda frekvenčný rozsah rovný frekvencii pôvodného tónu, v tomto prípade 440 Hz. Keďže oktáva obsahuje 1200 centov je zrejmé, že $A(t)$ získame ako

$$A(t) = \frac{R}{1200} f_0(t) \quad (3.5)$$

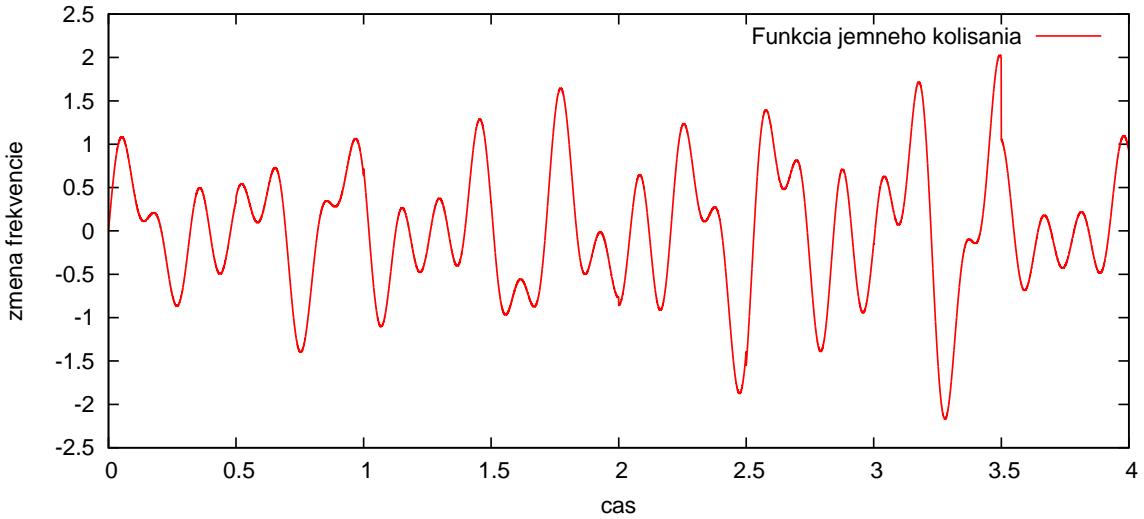
a keď poznáme amplitúdu pre daný tón, novú krivku f'_0 v čase t dostaneme ako

$$f'_0(t) = f_0(t) + A(t) \sin(2\pi f_m t). \quad (3.6)$$

Efektor, ktorý sme vytvorili pre aplikáciu vibráta umožňuje pridať vibráto tak na celú krivku, ako aj na jej úseky – intervale krivky. Toto sa dá využiť napríklad na aplikovanie vibráta len na úsek, ktorý zodpovedá samohláske. Ako vyzerá krivka F_0 po aplikovaní vibráta ukazuje obrázok 3.5a, vibráto aplikované len na niektorých úsekokach prezentuje obrázok 3.5b.



Obrázok 3.5: Ukážka aplikovania efektora pre vibráto na krivku F_0



Obrázok 3.6: Priebeh jemného kolísania podľa kvázináhodnej funkcie

3.3.3 Jemné kolísanie

Ďalší dôležitým znakom, ktorým sa vyznačuje hlas je jeho náhodné kolísanie. Jeho aplikovaním prinášame určitú náhodnosť do vytvorenej krivky. Prvý prístup pre aplikovanie jemného kolísania je pomocou kvázináhodnej funkcie prezentovanej v [7]:

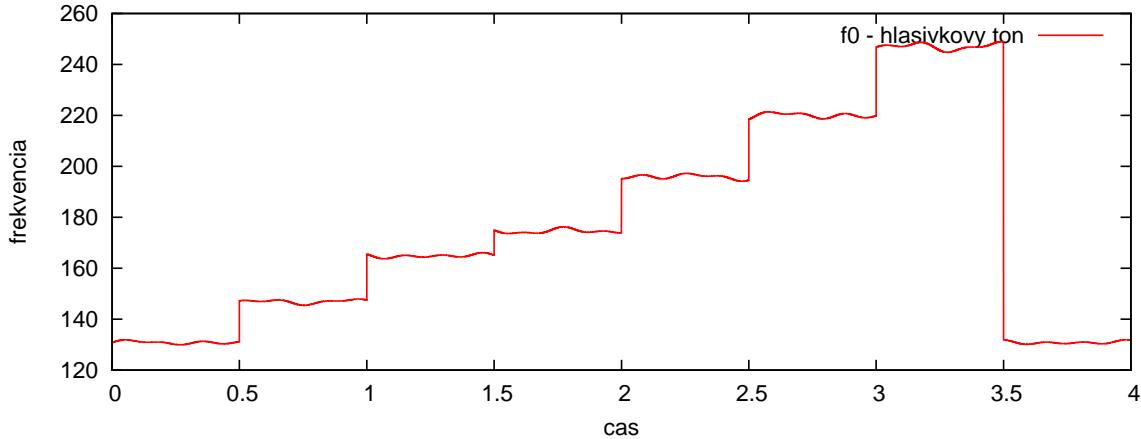
$$\Delta f_0(t) = \frac{f_0(t)}{100} \cdot \frac{\sin(12.7\pi t) + \sin(7.1\pi t) + \sin(4.7\pi t)}{3}, \quad (3.7)$$

kde $\Delta f_0(t)$ je zmena hlasivkového tónu v čase t . V efektore pre jemné kolísanie potom novú (pozmenenú) krivku f'_0 získame ako

$$f'_0(t) = f_0(t) + \Delta f_0(t). \quad (3.8)$$

Priebeh funkcie $\Delta f_0(t)$ zobrazuje obrázok 3.6. Krivku F_0 s pridaným kolísaním prezentuje obrázok 3.7.

Ďalší prístup pre tvorbu jemného kolísania si predstavíme neskôr, keďže tento prístup využíva digitálne filtre, o ktorých budeme rozprávať v ďalšej sekcií.



Obrázok 3.7: Krivka F_0 vytvorená z notového zápisu s pridaním jemného kolísania

3.3.4 Digitálne filtre

V ďalších efektoroch F_0 budeme využívať filtrovanie pomocou digitálnych filtrov. Budeme vychádzať z [13], kde boli filtre uvedené v analógovom tvare. Budeme sa zapodievať ich prevodom na digitálny filter.

Analógové filtre

Slúžia na odstránenie nejakého komponentu zo signálu. Spravidla je týmto komponentom frekvencia, pretože sa väčšinou snažíme potlačiť niektorú frekvenciu alebo pásmo frekven- cii, napr. pri odstraňovaní šumu. Veličina, ktorá charakterizuje filter sa nazýva prenosová funkcia a pri analógovom signále je táto funkcia definovaná nad komplexnou frekvenciu s .

Prenosová funkcia je definovaná ako podiel výstupného signálu $Y(s)$ a vstupného signálu $X(s)$ v doméne komplexnej premennej $s = \sigma + j\omega$. Dá sa vyjadriť ako podiel dvoch polynómov b a a na premennej s .

$$H(s) = \frac{Y(s)}{X(s)} = \frac{b(s)}{a(s)} = \frac{b_m s^m + \dots + b_1 s + b_0}{s^n + a_{n-1} s^{n-1} + \dots + a_1 s + a_0} \quad (3.9)$$

My ale využijeme filtre, kde $b_m = b_{m-1} = \dots = b_1 = 0$ a $b_0 \neq 0$ a dajú sa zapísať v tvare

$$H(s) = \frac{Y(s)}{X(s)} = \frac{b_0}{s^n + a_{n-1} s^{n-1} + \dots + a_1 s + a_0} = \frac{k}{s^n + a_{n-1} s^{n-1} + \dots + a_1 s + a_0} \quad (3.10)$$

Digitálne filtre

Môžeme ich definovať ako objekt, ktorého vstupom je jedna sekvencia čísel (vstupný signál) a výstupom je druhá sekvencia (filtrovaný signál)[16].

Charakteristikou digitálneho filtra je jeho prenosová funkcia. Na rozdiel od analógového signálu je ale definovaná v z -doméne, kde z je komplexné číslo $z = Ae^{j\varphi}$

$$H(z) = \frac{Y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + \dots + b_N z^{-N}}{1 + a_1 z^{-1} + \dots + a_M z^{-M}} \quad (3.11)$$

Ak chceme vypočítať filtrovaný signál na vzorke n a poznáme $H(z)$ budeme postupovať podľa

$$y[n] = - \sum_{k=1}^N a_k y[n-k] + \sum_{k=0}^M b_k x[n-k] \quad (3.12)$$

Predstavili sme si analógové a digitálne filtre a povedali ako budeme počítať filtrovaný signál. Zostáva ukázať ako budeme prevádzkať analógový filter na digitálny.

Bilineárna transformácia

Jedná sa o aparát na prevod funkcie v s -doméne do z -domény. Základom je substitúcia z :

$$z = e^{sT} = \frac{e^{sT/2}}{e^{-sT/2}} \approx \frac{1 + sT/2}{1 - sT/2}, \quad (3.13)$$

kde T je čas trvania jednej vzorky. Presvedčme sa že rovnosť 3.13 platí:

$$\begin{aligned} z &= e^{sT} & s &= \sigma + j\omega \\ z &= e^{(\sigma+j\omega)T} = e^{(\sigma T+j\omega T)} \\ z &= e^{\sigma T} e^{j\omega T} \end{aligned} \quad (3.14)$$

$$\begin{aligned} A &= e^{\sigma T} \in \mathcal{R} \\ z &= Ae^{j\omega T} = Ae^{j\varphi} \end{aligned}$$

Ak teraz z approximácie $z \approx \frac{1+sT/2}{1-sT/2}$ vyjadríme s , dostávame

$$s \approx \frac{2}{T} \frac{z-1}{z+1} \quad (3.15)$$

Pri prevode funkcií budeme za s dosádzať túto aproximáciu.

Prevod analógových filtrov

Analógové filtre budeme na digitálne prevádztať pomocou bilineárnej transformácie ich prenosovej funkcie. Pozrime sa ako previesť prenosovú funkciu typu $H(s) = \frac{k}{s^n + a_{n-1}s^{n-1} + \dots + a_1s + a_0}$.

Dosadíme za s :

$$H(z) = H(s)|_{\frac{2}{T}\frac{z-1}{z+1}} = \frac{k}{(\frac{2}{T}\frac{z-1}{z+1})^n + a_{n-1}(\frac{2}{T}\frac{z-1}{z+1})^{n-1} + \dots + a_1(\frac{2}{T}\frac{z-1}{z+1}) + a_0}$$

Vyjmeme konštantu $\frac{2}{T}$:

$$H(z) = \frac{k}{(\frac{2}{T})^n (\frac{z-1}{z+1})^n + a_{n-1}(\frac{2}{T})^{n-1} (\frac{z-1}{z+1})^{n-1} + \dots + a_1(\frac{2}{T})(\frac{z-1}{z+1}) + a_0}$$

Vyčleňme konštanty $(\frac{2}{T})^i$:

$$H(z) = H(z)|_{K_i=(\frac{2}{T})^i} = \frac{k}{K_n(\frac{z-1}{z+1})^n + a_{n-1}K_{n-1}(\frac{z-1}{z+1})^{n-1} + \dots + a_1K_1(\frac{z-1}{z+1}) + a_0}$$

Upravme menovateľa na spoločného menovateľa $(z+1)^n$:

$$H(z) = \frac{k}{K_n(z-1)^n + a_{n-1}K_{n-1}(z-1)^{n-1}(z+1) + \dots + a_1K_1(z-1)(z+1)^{n-1} + a_0(z+1)^n} \quad (3.16)$$

Prenásobme $\frac{(z+1)^n}{(z+1)^n}$:

$$H(z) = \frac{k(z+1)^n}{K_n(z-1)^n + a_{n-1}K_{n-1}(z-1)^{n-1}(z+1) + \dots + a_1K_1(z-1)(z+1)^{n-1} + a_0(z+1)^n}$$

Vyjadrime ako podiel polynómov $\frac{M(z)}{L(z)}$

$$H(z) = \frac{M_n z^n + M_{n-1} z^{n-1} + \dots + M_0}{L_n z^n + L_{n-1} z^{n-1} + \dots + L_0}$$

Predeľme čitateľa a menovateľa $L_n z^n$:

$$H(z) = \frac{\frac{M_n}{L_n} + \frac{M_{n-1}}{L_n} z^{-1} + \dots + \frac{M_0}{L_n} z^{-n}}{1 + \frac{L_{n-1}}{L_n} z^{-1} + \dots + \frac{L_0}{L_n} z^{-n}}$$

Odčítajme koeficienty a_i a b_i do prenosovej funkcie 3.11

Ukážme si prevod prenosovej funkcie 2. rádu ($n=2$):

$$\begin{aligned}
 H(s) &= \frac{k}{s^2 + a_1 s + a_0} \\
 &\dots \\
 H(z) &= \frac{k(z+1)^2}{K_2(z-1)^2 + a_1 K_1(z-1)(z+1) + a_0(z+1)^2} \\
 H(z) &= \frac{k(z^2 + 2z + 1)}{K_2(z^2 - 2z + 1) + a_1 K_1(z^2 - 1) + a_0(z^2 + 2z + 1)} = \frac{kz^2 + 2kz + k}{z^2(K_2 + a_1 K_1 + a_0) + z(-2K_2 + 2a_0) + (K_2 - a_1 K_1 + a_0)} \\
 H(z) &= \frac{\frac{k}{K_2 + a_1 K_1 + a_0} + \frac{2k}{K_2 + a_1 K_1 + a_0} z^{-1} + \frac{k}{K_2 + a_1 K_1 + a_0} z^{-2}}{1 + \frac{-2K_2 + 2a_0}{K_2 + a_1 K_1 + a_0} z^{-1} + \frac{K_2 - a_1 K_1 + a_0}{K_2 + a_1 K_1 + a_0} z^{-2}}
 \end{aligned}$$

Odčítajme koeficienty a_i a b_i :

$$\begin{aligned}
 \vec{a} &= \left\{ \frac{-2K_2 + 2a_0}{K_2 + a_1 K_1 + a_0}, \frac{K_2 - a_1 K_1 + a_0}{K_2 + a_1 K_1 + a_0} \right\} \\
 \vec{b} &= \left\{ \frac{k}{K_2 + a_1 K_1 + a_0}, \frac{2k}{K_2 + a_1 K_1 + a_0}, \frac{k}{K_2 + a_1 K_1 + a_0} \right\}
 \end{aligned} \tag{3.17}$$

Vieme teda ako manuálne vypočítať prevod prenosovej funkcie. Teraz predstavíme spôsob ako tento výpočet implementovať.

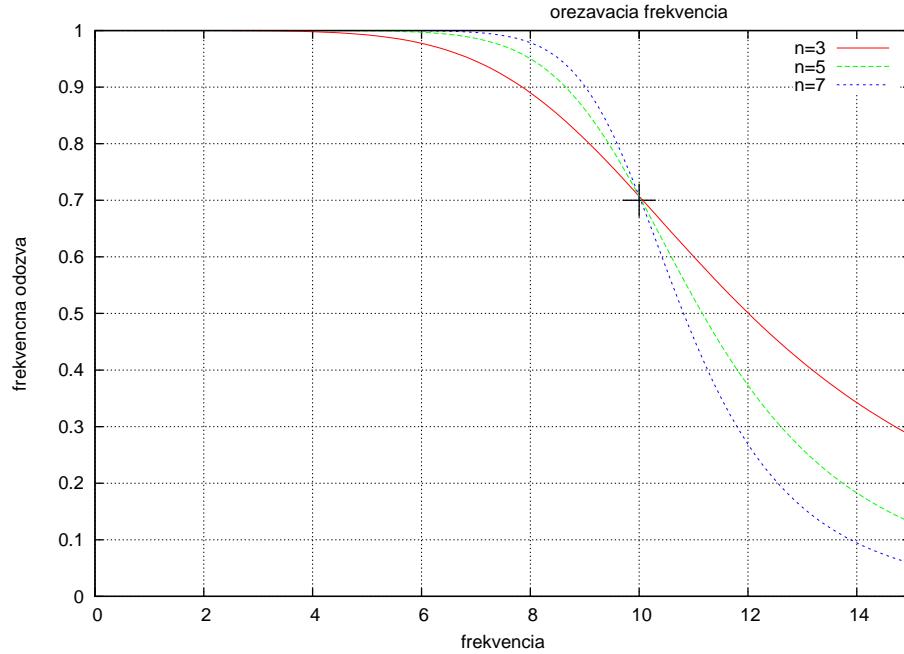
3.3.5 Trieda filtra

Vytvoríme triedu, ktorá bude reprezentovať filter. Vstupom do tejto triedy bude vzorkovacia frekvencia f_{vz} a parametre reprezentujúce prenosovú funkciu analógového filtra $H(s) = \frac{k}{s^n + a_{n-1}s^{n-1} + \dots + a_1s + a_0}$ tj. k a polynom p , ktorý reprezentuje menovateľ prenosovej funkcie.

Na reprezentáciu polynómu použijeme triedu, ktorá umožňuje operácie na polynómoch: sčítanie, odčítanie a násobenie dvoch polynómov, umocňovanie polynómu a delenie polynómu číslom. Táto trieda je kľúčovou pre implementáciu triedy filtra. Algoritmus implementuje postup 3.16, teda prevádzka analógový filter na digitálny.

Začnime s krokom kde $H(z) = \frac{k(z+1)^n}{K_n(z-1)^n + a_{n-1}K_{n-1}(z-1)^{n-1}(z+1) + \dots + a_1K_1(z-1)(z+1)^{n-1} + a_0(z+1)^n}$, pričom vypočítame T -čas jednej vzorky $T = 1/f_{vz}$ a $K_i = (2/T)^i = (2f_{vz})^i$. Čitateľa a menovateľa reprezentujeme ako polynom a ďalej pokračujeme podľa postupu 3.16, kde upravujeme čitateľa a menovateľa pomocou metód triedy polynómov. Koeficienty a_i a b_i uložíme v triede filtra.

Trieda poskytuje metódu, ktorá filtriuje vstupný signál podľa koeficientov a_i a b_i podľa vzťahu 3.12.



Obrázok 3.8: Frekvenčná odozva dolnopriepustného butterworthovho filtra stupňa 3,5 a 7 s orezávacou frekvenciou 10 Hz, ktorá sa potlačí faktorom $\frac{1}{\sqrt{2}} \approx 0.7$

3.3.6 Butterworthov dolnopriepustný filter

Je jedným z najpoužívanejších filtrov v spracovaní signálu a to najmä preto, že jeho frekvenčná odozva je hladká a neobsahuje žiadne výkyvy v prieplustnom pásme (obrázok 3.8).

Prenosová funkcia

Je ju možné zapísť v tvare

$$H(s) = \frac{1}{(\frac{s}{\omega_c})^n + a_{n-1}(\frac{s}{\omega_c})^{n-1} + \dots + a_1(\frac{s}{\omega_c}) + 1}, \quad (3.18)$$

kde ω_c je orezávacia frekvencia v rad/s , n je rád a a_i sú koeficienty filtra. a_i získame odčítaním koeficientov Butterworthových polynómov:

$$\begin{aligned} B_n(s) &= \prod_{k=1}^{\frac{n}{2}} \left[s^2 - 2s \cos\left(\frac{2k+n-1}{2n}\pi\right) + 1 \right], \text{ pre } n \text{ párne} \\ B_n(s) &= (s+1) \prod_{k=1}^{\frac{n-1}{2}} \left[s^2 - 2s \cos\left(\frac{2k+n-1}{2n}\pi\right) + 1 \right], \text{ pre } n \text{ nepárne} \end{aligned} \quad (3.19)$$

Ukážme, ako získame prenosovú funkciu pre $n = 2$ a orezávaciu frekvenciu $f_c = 10\text{Hz}$ ($\omega_c = 2\pi f_c = 62.832 \text{ rad/s}$) :

- Vypočítame $B_2(s) = s^2 - 2s \cos\left(\frac{3}{4}\pi\right) + 1 = s^2 + \sqrt{2}s + 1 = s^2 + 1.414s + 1$
a odčítame koeficienty: $a_1 = 1.414$

- Dosadíme do predpisu prenosovej funkcie

$$H(s) = \frac{1}{(\frac{s}{\omega_c})^2 + a_1(\frac{s}{\omega_c}) + 1} = \frac{1}{(\frac{1}{\omega_c})^2 s^2 + a_1(\frac{1}{\omega_c})s + 1} = \frac{1}{(\frac{1}{62.832})^2 s^2 + 1.414(\frac{1}{62.832})s + 1} = \frac{1}{0.00025s^2 + 0.02250s + 1}$$

Implementáciu tvorby prenosovej funkcie dolnopriepustného filtra zabezpečuje trieda polynómov, pomocou ktorej najprv vypočítame príslušný Butterworthov polynom podľa rádu filtra. Následne zostrojíme menovateľ prenosovej funkcie ako polynom, do ktorého vstupujú koeficienty vypočítaného but. polynómu a orezávacia uhlová frekvencia. Nakoniec analógový butterworthov filter prevedieme na digitálny pomocou bilineárnej transformácie v triede filtra.

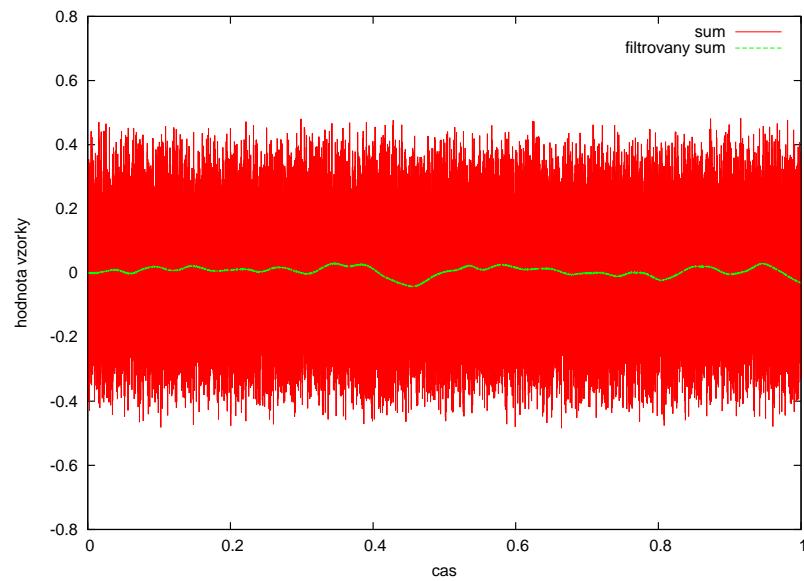
Teraz, keď sme si predstavili všetky potrebné komponenty môžeme sa vrátiť k efektom, ktoré využívajú na tvarovanie krivky F_0 filtro.

3.3.7 Jemné kolísanie pomocou filtra

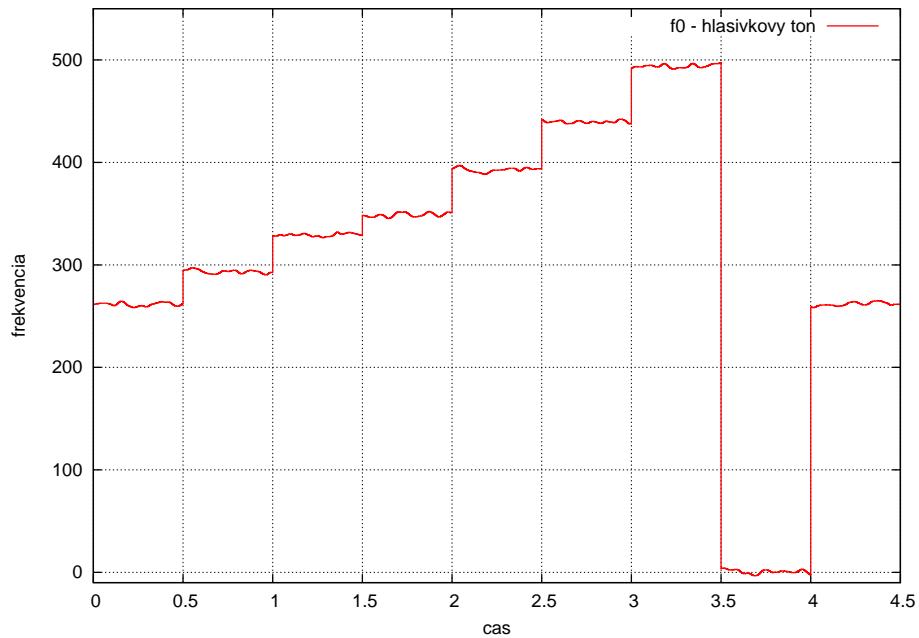
Implementácia jemného kolísania pomocou filtra bude prebiehať v štyroch krokoch:

1. **Vytvoríme biely šum**, ktorého dĺžka bude zodpovedať dĺžke krivky F_0 , ktorú budeme tvarovať.
2. **Prefiltrujeme šum** dolnopriepustným Butterworthovym filtrom 3. rádu s orezávacou frekvenciou 10 Hz.
3. **Normalizujeme** prefiltrovaný signál, tak aby sa amplitúda pohybovala v rozsahu $\pm 5 \text{ Hz}$.
4. **Pripočítame** prefiltrovaný normalizovaný signál k pôvodnej krivke.

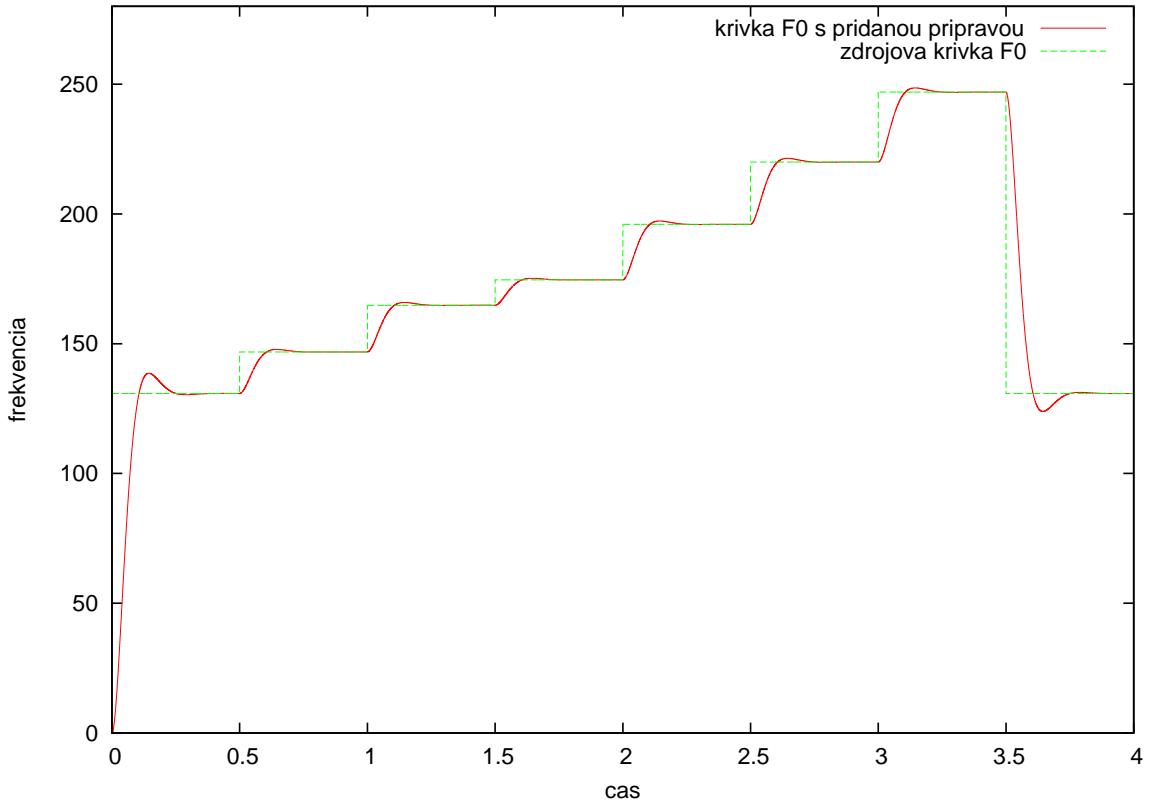
Na obrázku 3.9 vidno biely šum a signál, ktorý vznikne jeho prefiltrovaním a na obrázku 3.10 je krivka F_0 , ktorá vznikla z notového zápisu s pridaným jemným kolísaním.



Obrázok 3.9: Biely šum a biely šum prefiltrovaný butterworthovym dolnopriepustným filtrom s orezávacou frekvenciou 10 Hz



Obrázok 3.10: Krivka F_0 s pridaným jemným kolísaním

Obrázok 3.11: Ukážka zdrojovej krivky F_0 a krivky s pridanou prípravou

3.3.8 Príprava

Popísali sme ju v časti 2.4.1. Jej implementácia spočíva v aplikovaní filtra s prenosovou funkciou

$$H(s) = \frac{k}{s^2 + 2\zeta\omega s + \omega^2} \quad (3.20)$$

na zdrojovú krivku F_0 , kde parametre nastavíme $\zeta = 0.6681$, $\omega = 29.2$ a $k = \omega^2 = 852.64$. Tieto parametre sme prevzali z [13]. Ukážka zdrojovej krivky F_0 a krivky s pridanou prípravou je na obrázku 3.11.

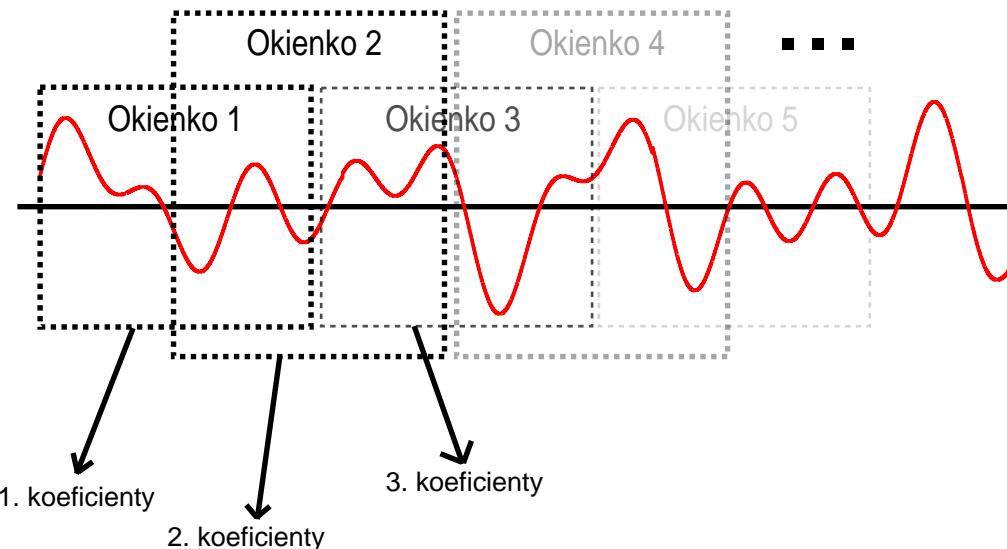
3.4 Syntéza

V tejto časti ukážeme ako bude vyzeráť syntéza spevu, aké budú vstupy do syntézy a akým spôsobom z nich vytvoríme výstupný–syntetizovaný spev. Predstavíme si dva prístupy k syntéze.

3.4.1 Syntéza prostredníctvom LPC koeficientov

LPC–Lineárne prediktívne kódovanie je metóda, ktorej primárny využitím je analýza akustického signálu. LPC funguje na krátkodobom základe a pracuje so signálom rozdeľeným na menšie časti tzv. okienka. Rozdelenie signálu na okienka je na obrázku 3.12, kde sa okienka prekrývajú polovicou svojej dĺžky. Prekryv môžeme ale zvoliť ľubovoľne.

Vstupom do LPC je jedno okienko signálu a výstupom je lineárny digitálny filter (sada koeficientov filtra) pre dané okienko.



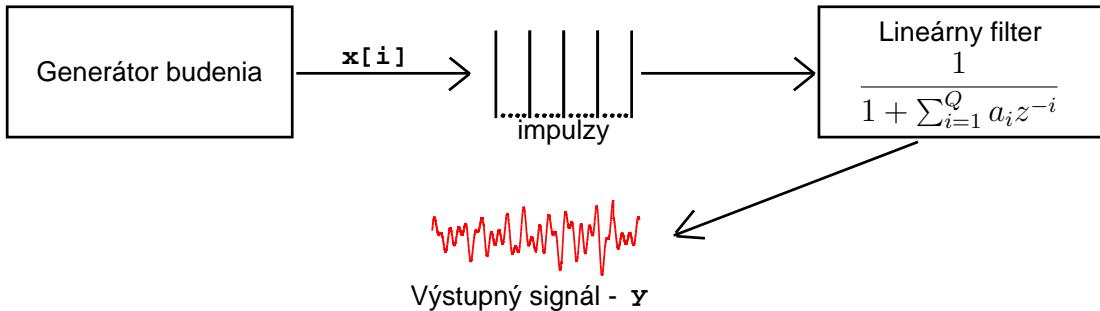
Obrázok 3.12: Rozdelenie vstupného signálu na okienka

Toto metódou sa odhadujú parametre modelu vytvárania reči. Predstavme si tento model.

Model vytvárania reči s lineárnym filtrom

Skladá sa z generátora budenia a zo sady lineárnych filtrov (obr. 3.13).

Generátor budenia v prípade znelých zvukov budí systém na základe hlasivkového tónu F_0 . Ak bol $F_0 = 200\text{Hz}$ znamená to že generátor budenia vyprodukuje impulz jednotkovej veľkosti rovnomerne 200 krát za sekundu; všeobecne dá generátor jednotkový impulz každú $\frac{1}{F_0}$ sekundy, inak dá nulový impulz[12].



Obrázok 3.13: Model vytvárania reči

V prípade neznelých zvukov produkuje generátor náhodný (biely) šum.

Sada lineárnych filtrov je reprezentovaná sadou koeficientov a_i pre digitálny filter s prenosovou funkciou

$$H(z) = \frac{1}{1 + \sum_{i=1}^Q a_i z^{-i}}, \quad (3.21)$$

kde Q je rám filtra.

LPC je založený na predpokladu, že k -ta vzorka signálu $y[k]$ sa dá získať ako lineárna kombinácia Q predchádzajúcich vzoriek a budenia, tj.

$$y[k] = - \sum_{i=1}^Q a_i y[k-i] + x[k], \quad (3.22)$$

kde $x[k]$ je k -ty impulz budenia.

Výpočet LPC

Na výpočet koeficientov LPC využijeme Levinson-Durvinovú rekurzívnu metódu. Výpočet prebieha vrámci jedného okienka signálu v cykle pre $i = 1, 2, \dots, Q$ [12]

$$\begin{aligned} E^{(0)} &= R(0), \\ k_i &= -[R(i) + \sum_{j=1}^{i-1} a_j^{(i-1)} R(i-j)]/E^{(i-1)}, \\ a_i^{(i)} &= k_i, \\ a_j^{(i)} &= a_j^{(i-1)} + k_i a_{i-j}^{(i-1)}, \quad 1 \leq j \leq i-1, \\ E^{(i)} &= (1 - k_i^2)E^{(i-1)}, \end{aligned} \quad (3.23)$$

pričom R je autokorelačná funkcia na okienku:

$$R(i) = \sum_{k=0}^{N-1-i} y[k]y[k+i] \quad (3.24)$$

a N je počet vzoriek v okienku. Výsledné koeficienty filtra budú $a_i = a_Q^{(i)}$, pre $i = 1, \dots, Q$.

Použitie LPC na spev

Doteraz sme hovorili o LPC iba ako o nástroji pre prácu s rečovým signálom. Ako ale použiť LPC na spev? V našom návrhu budeme vychádzať z faktu, že pri speve je frekvencia hlasíkového tónu rovná frekvencii tónu, ktorý spievame. Tvorbe krivky F_0 sme sa už venovali a vieme ju získať. Dôležité je teraz zstrojiť komponenty, pre model vytvárania reči (obr. 3.13): **impulzy** a **sady filtrov**:

Impulzy získame prevodom krivky F_0 :

Algorithm 2 Zstrojenie impulzov z krivky F_0

```
{vstup: vzorkovacia frekvencia -  $f_{vz}$ , krivka  $F_0$  -  $f_0$ , výstup: impulzy -  $imp$ }
pocetVzoriek  $\leftarrow f_{vz} * f_0.trvanie$ 
omeskanie  $\leftarrow f_{vz}/f_0[0]$  {O kolko vzoriek bude dalsi impulz}
for  $i = 1$  to  $pocetVzoriek - 1$  do
     $omeskanie \leftarrow omeskanie - 1$ 
    if  $omeskanie = 0$  then
         $omeskanie \leftarrow f_{vz}/f_0[i]$  {O kolko vzoriek bude dalsi impulz}
         $imp[i] = 1$ 
    else
         $imp[i] = 0$ 
    end if
end for
return  $imp$ 
```

Sady filtrov dostaneme zduplicovaním potrebného počtu okienok vstupného signálu. Týmto vlastne budeme niektorý filter používať viackrát V našom prípade budú vstupom okienka (koeficienty filtra) získané z nejakej vyslovenej samohlásky. Postup duplikácie určuje algoritmus 3 a zobrazuje obrázok 3.14

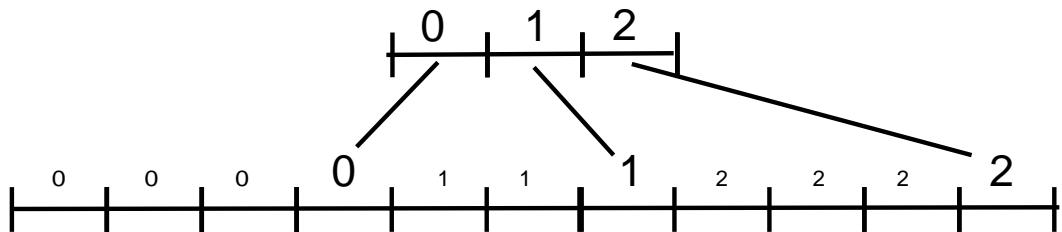
Takže teraz máme vypočítane oba komponenty modelu a môžeme sa pustiť do syntézy. Budeme postupovať podľa vzťahu 3.22, kde $x[k]$ je impulz k -tej vzorky, pričom budeme

Algorithm 3 Duplikácia okienok

```

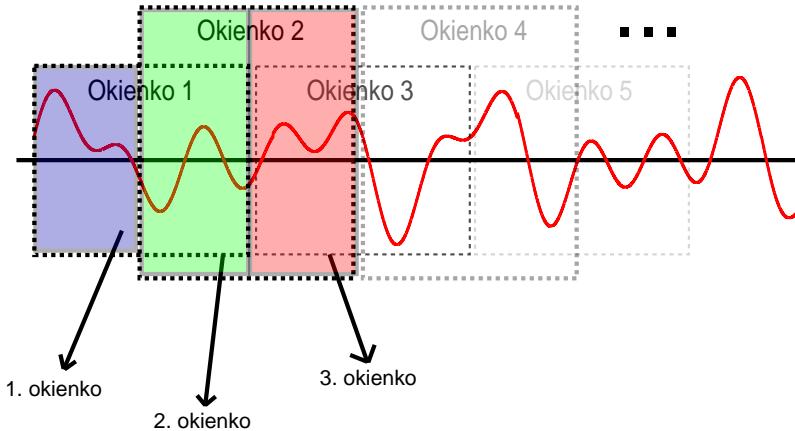
{vstup: krivka  $F_0 - f_0$ ,  $windows[]$  - pole okienok, výstup:  $newWins[]$  - pole nových
okienok}
faktorPosunu  $\leftarrow (f_0.trvanie/windows[].trvanie)$ 
pocetNovychOkienok  $\leftarrow faktorPosunu * windows[].pocet$ 
for  $i = 0$  to  $pocetNovychOkienok - 1$  do
     $newWins[i] = windows[i/faktorPosunu]$ 
end for
return  $newWins$ 

```



Obrázok 3.14: Ukážka duplikácie okienok. Horný zdrojový signál má dĺžku 3 okienka, výstupný má dĺžku 11 okienok. Zdrojové okienka sa rovnomerne rozdelia do výstupného signálu

meniť koeficienty a_i podľa okienka, v ktorom sa nachádzame (vždy zvolíme to okienko, ktoré nie je prekryté ďalším okienkom viď obr. 3.15).



Obrázok 3.15: Znázornenie výberu správneho okienka pri syntéze

Kvalita spevu

Tento postup sme vyskúšali na nahovorenej samohláske „a“, kde bol výsledný zvuk po úprave príliš nekvalitný a značne plechový. Výsledok bol natoľko neuspokojivý, že sme sa rozhodli od tejto metódy upustiť a ďalej ju nerozvíjať. To nás viedlo k ďalšej metóde syntézy.

Ukážku si je možné vypočuť na priloženom médiu.

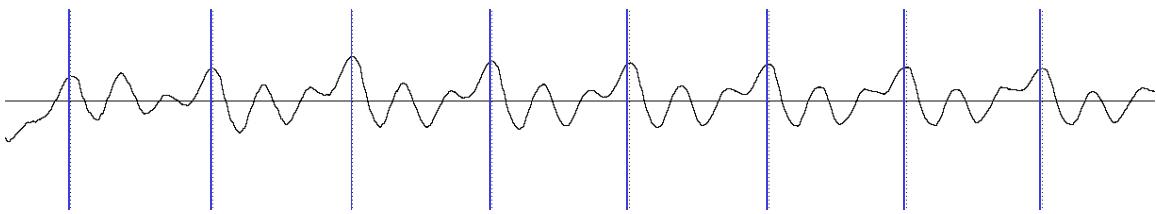
3.4.2 Syntéza prostredníctvom algoritmu PSOLA

Algoritmus PSOLA (Pitch Synchronous Overlap Add Method) sa používa v prvom rade na konkatenačnú syntézu reči. Techniky založené na algoritme PSOLA reč priamo nevytvárajú ale umožňujú zrečazovať uložené rečové segmenty a meniť pritom periódou hlasivkového tónu[12].

Jedná sa teda presne o to čo potrebujeme – poznáme frekvenciu hlasivkového tónu v čase a máme k dispozícii rečové segmenty.

Algoritmus

Vstupom do nášho algoritmu je signál nejakého znelého zvuku (v našom prípade vyslovenej samohlásky), jeho vzorkovacia frekvencia f_{vz} , umiestnenie hlasivkových impulzov v tomto signáli a časť krvky F_0 , zodpovedajúca výslednému zvuku. U samohlások sú hlasivkové impulzy väčšinou pozície lokálnych máxim. Ilustrácia hlasivkových impulzov je na obrázku 3.16.



Obrázok 3.16: Hlasivkové impulzy (modré) v samohláske „u“

Na získanie hlasivkových impulzov existujú automatizované nástroje, ktoré ale nedávajú úplne presné výsledky a treba ich manuálne prezrieť a poopraviť. Vyskúšali sme knižnicu Aubio³.

Výstupom je nasynthetizovaný spev zdrojovéj samohlásky (resp. zvuku).

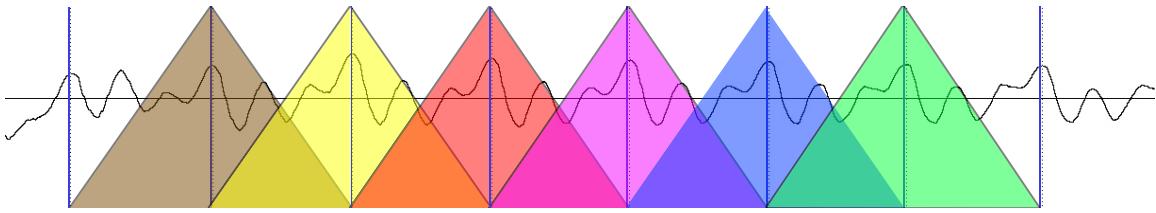
Postup:

1. Rozdeľ vstupný signál na okienka tak, aby okienko začínalo v predchádzajúcim impulze a končilo v nasledujúcim. Prvé okienko nezarad' medzi okienka, keďže môže byť na hranici so spoluhláškou⁴.
2. Aplikuj na okienko váhovanie. Tu môžeme použiť rôzne váhové okienka⁵, pri pokusoch sa najviac osvedčilo trojuholníkové okienko: $w(n) = \frac{2}{N-1} \cdot \left(\frac{N-1}{2} - \left| n - \frac{N-1}{2} \right| \right)$ kde n je pozícia v okne a N je veľkosť okna. Výsledok algoritmu po bode 2 je na obrázku 3.17.
3. Podľa vstupnej krvky F_0 zisti výslednú dĺžku zvuku t_{new} a umiestnenie (indexy) impulzov (viď algoritmus 2, kde si teraz budeme pamätať indexy, pri ktorých bol impulz nenulový).

³<http://aubio.org/>

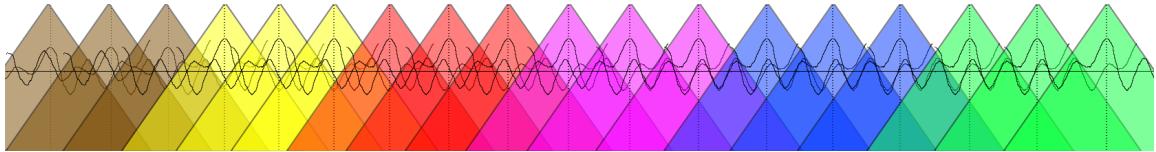
⁴Môže obsahovať časť spoluhlášky a taktiež nepoznáme jeho začiatok

⁵často sa zvykne využívať Hammingovo a von Hannovo váhové okienko



Obrázok 3.17: Zdrojový signál rozdelený na okienka a ováhovaný

4. Vytvor sadu nových okienok. Ich počet je rovný počtu impulzov v zdrojovej krvke F_0 . Okienka rovnomerne zaplň pôvodnými okienkami (obdobne ako na obr. 3.14 pomocou algoritmu 3). Ako vyzerá stav po tomto kroku ak frekvencia hlasivkového tónu je väčšia znázorňuje obrázok 3.18 (farby okienok zodpovedajú okienkam na obr. 3.17)



Obrázok 3.18: Nové okienka vytvorené z pôvodných, rovnomerne rozmiestnené do veľkosti výstupného signálu podľa polohy hlasivkových impulzov

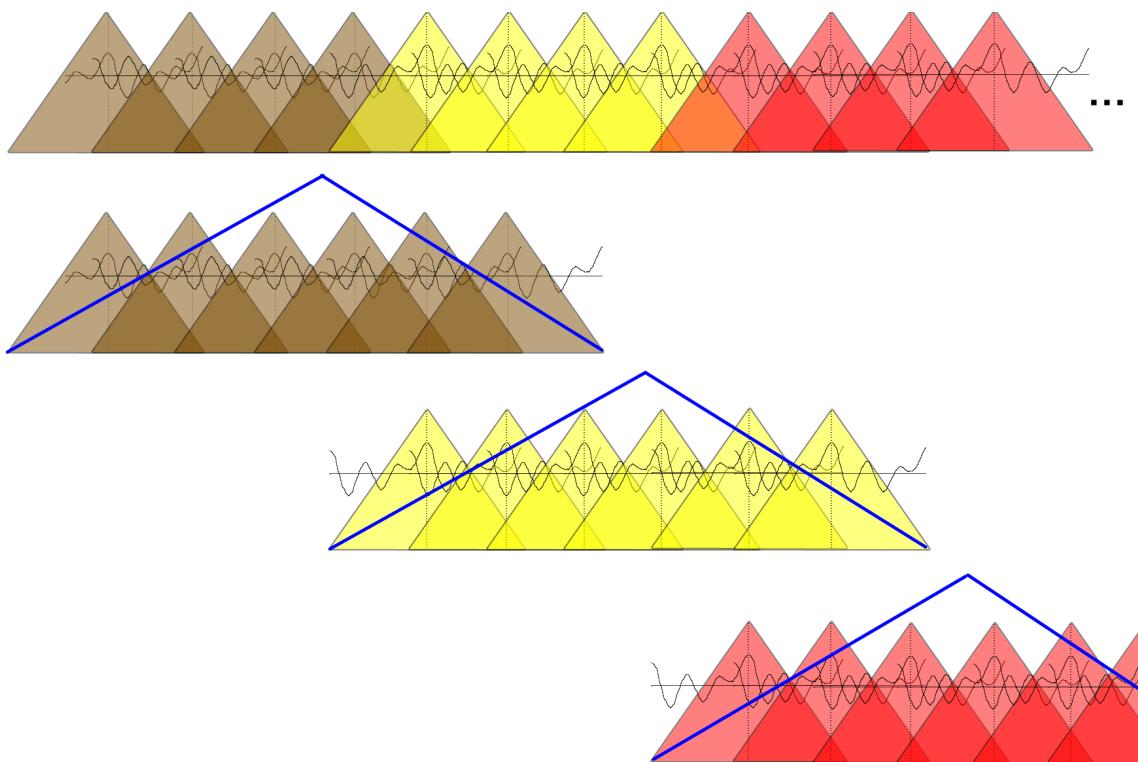
5. Prejdi v cykle cez celú dĺžku (čas) výsledného signálu (k od 0 po $t_{new}f_{vz}$), zisti, ktoré okienka zasahujú do daného času a zisti ich váhu v tomto čase. Výslednú vzorku potom vypočítaj ako $\frac{\sum_{m=0}^M w_m(k)*x_m[k]}{\sum_{m=0}^M w_m}$, kde $w_m(k)$ je váha m -tého okienka, ktoré zasahuje do času k , $x_m[k]$ je vzorka v tomto čase a M je počet okienok, ktoré zasahujú do času k .

Vylepšenie algoritmu

Pri podrobnejšom prezretí predchádzajúceho algoritmu si všimnime fakt – ako sú spájané jednotlivé okienka. Ak spájame rovnaké okienka (napr. iba hnedé z obrázka 3.18) je ich spoj plynulý (zvukovo bez puknutia), keďže na spoji sú vzorky veľmi podobnej intenzity. Keď ale spájame rozdielne okienka (napr. hnedé a žlté) môže na spoji prísť k náhlemu zvýšeniu intenzity (k puknutiu). Všimnime si, že na obrázku 3.17 sú okienka vždy prekryté približne do polovice, zatiaľ čo na obrázku 3.18 sa prekrývajú oveľa väčšou časťou.

Na riešenie "puknutí" medzi rozdielnymi okienkami sme navrhli takúto techniku:

1. Vygeneruj segmenty zložené z rovnakých zdrojových okienok podobne ako v pôvodnom algoritme len s tou zmenou, že okienka jedného typu budeme umiestňovať až do polovice nasledujúceho pôvodného segmentu. Všetko objasní obrázok 3.19
2. Prelož cez tieto nové segmenty váhové okienko. Výsledok algoritmu po tomto kroku je znázornený na obrázku 3.19.



Obrázok 3.19: Predĺženie pôvodných segmentov rovnakých okienok a ich preloženie váhovým okienkom

3. Vypočítaj výsledný signál ako súčet ováhovaných vzoriek nových segmentov. Postup je rovnaký ako v bode 5. pôvodného algoritmu, s tým rozdielom, že za okienka považujeme nové segmenty.

Týmto spôsobom je prechod medzi segmentami rovnakých okienok plynulejší ako tomu bolo v pôvodnom algoritme, kde sa segmenty rovnakých okienok spájali len na hranici medzi segmentami.

Kvalita spevu

Navrhovaný algoritmus sme vyskúšali na vzorkách nahovorených samohlások „a“, „e“, „i“, „o“ a „u“. Kvalita spevu bola veľmi dobrá, blížiaca sa reálnemu spevu. Najlepšie výsledky sme dostali pri samohláskach „o“ a „a“. Oproti pôvodnému algoritmu sa kvalita zlepšila. Pre najlepšiu kvalitu spevu však treba dbať na to, aby v zdrojovom zvukovom súbore bolo čo najmenej šumu.

3.4.3 Celkový priebeh syntézy

V tejto časti popíšeme aký je priebeh celkovej syntézy v aplikácii, ktorú sme vytvorili.

Programovací jazyk a operačný systém

Programovacím jazykom bola zvolená Java 6, pretože v tomto jazyku sme mali už pred prácou naprogramované niektoré triedy pre prácu so zvukom. Aplikácia beží len pod OS Windows, nakoľko využíva spustiteľné súbory pre Windows.

Vstupy

Vstupom do syntézy je **zvukový súbor s nahratou rečou, súbor s piesňou, súbor s umiestnením hlasíkových impulzov a parametre syntézy**:

Zvukový súbor s nahratou rečou - súbor typu WAV (Waveform Audio File Format)
s jedným kanálom a ľubovoľnou vzorkovacou frekvenciou

Súbor s piesňou - súbor popisujúci tempo, tóny, pauzy a slabiky piesne (viď. časť 3.1.1)

Súbor s umiestnením hlasíkových impulzov - súbor obsahujúci postupnosť desatiných čísel – umiestnení hlasíkových impulzov v čase

Parametre syntézy - umožňujú nastaviť druh syntézy:

Spievanie iba samohlásky - zaspieva namiesto každej slabiky iba samohlásku, ktorú obsahuje.

Spievanie na samohlásku z n -tej slabiky - zaspieva celú pieseň na samohlásku, ktorú obsahuje n -ta slabika.

Kombinácie efektorov F_0 - umožňuje kombinovať efektory F_0 , dajú sa prepínať prednastavené kombinácie.

Výstupy

Výstupom aplikácie je zvukový WAV súbor s nasynthetizovaným spevom.

Postup

1. **Rozpoznaj jednotlivé fonémy:** Na rozpoznanie foném používame knižnicu HTK, ktorá nám určuje hranice jednotlivých foném. HTK pracuje na princípe skrytých Markovych modelov. Tieto modely boli natrénované slovenskými rečníkmi a získali sme ich od školiteľa.
2. **Priprav krivku F_0 :** Zo vstupného súboru s piesňou vytvoríme krivku F_0 a aplikujeme na ňu efektory.
3. **Syntetizuj:**

- (a) Pre tón zisti príslúchajúcemu slabiku a dĺžky foném v tejto slabike

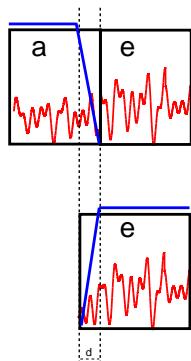
Ak sa spievajú samohlásky aj spoluohlásky:

- i. Zisti dĺžku samohlásky v slabike a časť krivky F_0 patriacej tejto samohláske a nasynthetizuj vylepšeným algoritmom PSOLA so vstupným zvukom príslušnej samohlásky.
- ii. K zvuku samohlásky pripoj pôvodný zvuk spoluohlásky (pred samohlásku resp. za samohlásku podľa polohy spoluohlásky v slabike).
- iii. Opakuj (a),i.,ii.,iii. pre všetky tóny a pospájaj zvuky jednotlivých tónov dohromady.

Ak sa spievajú len samohlásky alebo sa spieva na n -tu samohlásku:

- i. Dĺžku samohlásky nastav na dĺžku slabiky, ak si aspoň na druhej slabike pripočítaj k dĺžke samohlásky dĺžku d prechodovej oblasti (d sme predvolili na 5 ms), zisti časť krivky F_0 patriacej tejto samohláske. Ak sa spieva na n -tu samohlásku vstupom do PSOLy je vždy zvuk tejto samohlasky, inak je to zvuk samohlásky príslušnej slabiky. Nasynthetizuj vylepšeným algoritmom PSOLA.

- ii. Ak nie si na prvej slabike urob prechod medzi samohláskami a to tak, že zober posledných k vzoriek z predchádzajúcej samohlásky zodpovedajúce dlžke d a urob lineárnu interpoláciu aktuálnej a predchádzajúcej samohlásky na k vzorkách predchádzajúcej samohlásky (viď obr. 3.20).
 - iii. Opakuj (a),i.,ii.,iii. pre všetky tóny a pospájaj zvuky jednotlivých tónov dohromady.



Obrázok 3.20: Ukážka spájania samohlások pomocou lineárnej interpolácie. Modrá krivka vyznačuje mieru, akou do výsledného zvuku zasahuje príslušná samohláska. Prerušované čiary vymedzujú prechodovú oblasť dĺžky d

Kapitola 4

Záver

4.1 Zhrnutie

Prvú časť našej práce sme venovali prehľadu techník syntézy, ktorými sme sa inšpirovali pri návrhu a implementácii. Následne sme sa venovali reprezentácií piesne a tvorbou melódie. Potom sme sa zaoberali tvorbou a tvarovaním hlasivkového tónu F_0 . Navrhli a implementovali sme triedy efektorov F_0 pre aplikáciu znakov spevu na krivku hlasivkového tónu. Predstavili sme triedu pre digitálny filter a triedy efektorov F_0 , založené na využití filtrov. Po získaní krivky F_0 sme sa venovali syntéze. Vyskúšali sme dva prístupy k syntéze – pomocou LPC a prostredníctvom algoritmu PSOLA. Syntéza pomocou LPC sa ukázala kvalitatívne nedostačujúca, a preto sme zvolili pre syntézu algoritmus PSOLA. Implementovali sme ho a podarilo sa nám tento algoritmus vylepšiť. Nakoniec sme popísali a implementovali výsledný syntetizér.

4.2 Anketa

Vytvorili sme anketu¹, v ktorej sme sa respondentov pýtali na kvalitu syntézy zaspievanej oktávy. Oktáva bola naspevaná v ukážkovom súbore na samohlásky „a“, „e“, „i“, „o“ a „u“.

Otázka na respondetov znala: „Ako veľmi sa nahrávka „a“ podobá reálnemu spevu?“, kde respondenti mali možnosť hodnotiť nahrávku stupnicou 1-5, kde jednotlivé stupne znamenajú: 1–veľmi podobná, 2–značne podobná, 3–trochu podobná, 4–ide ešte o spev a 5–vôbec sa nepodobá na spev.

¹<http://spreadsheets.google.com/viewform?formkey=dE1US3dOV1Y0cE1pUE1rVzJYMndnR1E6MQ&ifq>

	„a“	„e“	„i“	„o“	„u“
priemerná známka:	1.55	1.70	2.20	1.60	2.35
najčastejšia známka (medián):	1.5	1.5	2	1	2
priemerná známka všetkých samohlások spolu :				1.88	
počet respondentov:				20	

Tabuľka 4.1: Tabuľka s výsledkami ankety

Obdobne sme sa pýtali na všetky samohlásky.

V ankete sa zúčastnilo 20 respondentov, ktorých odpovede zhrňuje tabuľka 4.1.

Z ankety vyplýva, že nasynthetizovaný spev samohlások pôsobí na poslucháča veľmi reálne. Najviac sa reálnemu spevu podobá spev samohlásky „a“ a „o“, no aj kvalita ostatných spievaných samohlások je veľmi dobrá.

4.3 Výsledky

Navrhli a zostrojili sme počítačový syntetizér spevu. Syntetizér dáva veľmi dobré výsledky pri syntéze spevu samohlások. Pri syntéze samohlások dokáže syntetizér vytvoriť spev veľmi blízky reálnemu spevu. Tento spev má charakteristickú farbu rečníka.

V prípade syntézy len samohlások sa nám cieľ podarilo splniť.

Kvalita syntézy celých slabík však zostáva nízka, nakoľko spoluuhlásky sú v priebehu syntézy len prilepené k samohláskam bez ďalšieho spracovania z pôvodnej reči. Spojit verne spoluuhlásky a samohlásky sa nám nepodarilo.

Kvalitu spevu ovplyvňuje aj zdrojová nahrávka reči. Pre najlepšiu kvalitu treba zabezpečiť, aby táto nahrávka obsahovala čo najmenej šumu.

Výstupy z aplikácie si možno vypočuť na priloženom médiu.

4.4 Budúca práca

Spočíva najmä v odstránení hlavného nedostatku – spájania spoluuhlások a samohlások. Ďalšie vylepšenia by mohli spočívať v aplikovaní efektov týkajúcich sa dynamiky (akcenty) a v štúdiu a implementácii legáta (viazanie tónov v speve). Pozornosť by sa taktiež mohla venovať automatizácii hľadania hlasivkových impulzov v samohláskach.

Literatúra

- [1] Pavol Adam. *Úvod do metód spracovania zvuku v súčasnom multimediálnom prostredí*. Máj 2006. Diplomová práca, Univerzita Komenského, Bratislava, Fakulta matematiky, fyziky a informatiky, Katedra aplikovanej informatiky, študijný odbor: informatika, vedúci: Ľubomír Lúčan, CSc.
- [2] Hideki Banno, Hiroaki Hata, Masanori Morise, Toru Takahashi, Toshio Irino, and Hideki Kawahara. Implementation of realtime straight speech manipulation system: Report on its first implementation. *Acoustical Science and Technology*, 28(3):140–146, 2007.
- [3] J. Bonada and X. Serra. Synthesis of the singing voice by performance sampling and spectral models. *Signal Processing Magazine, IEEE*, 24(2):67–79, March 2007.
- [4] Hung-Yan Gu and Zheng-Fu Lin. Mandarin singing voice synthesis using ann vibrato parameter models. In *Machine Learning and Cybernetics, 2008 International Conference on*, volume 6, pages 3288–3293, July 2008.
- [5] Michael Hardy. *Transfer function*. Január 2010.
http://en.wikipedia.org/wiki/Transfer_function.
- [6] Jongin Lim Kihong Kim, Jinkeun Hong. Multiresolution sinusoidal speech model using elliptic band pass filter. *NON-LINEAR SPEECH PROCESSING, Barcelona, 19-22 April, 2005*, 2005.
- [7] Wen-Hsing Lai. F0 control model for mandarin singing voice synthesis. In *Digital Telecommunications, 2007. ICDT '07. Second International Conference on*, pages 12 –12, july 2007.

- [8] J. Laroche, Y. Stylianou, and E. Moulines. Hnm: a simple, efficient harmonic+noise model for speech. In *Applications of Signal Processing to Audio and Acoustics, 1993. Final Program and Paper Summaries., 1993 IEEE Workshop on*, pages 169 –172, oct 1993.
- [9] M.E. Lee and M.J.T. Smith. Digital singing voice synthesis using a new alternating reflection model. In *Circuits and Systems, 2002. ISCAS 2002. IEEE International Symposium on*, volume 2, pages II–863–II–866 vol.2, 2002.
- [10] M.W. Macon, L. Jensen-Link, J. Oliverio, M.A. Clements, and E.B. George. A singing voice synthesis system based on sinusoidal modeling. In *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, volume 1, pages 435–438 vol.1, Apr 1997.
- [11] Hansjorg Mixdorff. *Intonation Patterns of German - Model-based Quantitative Analysis and Synthesis of F0 contours*. Máj 1998. Dissertation thesis in Electrotechnik, University of Technology, Dresden, Germany.
- [12] J. Psutka, L. Müller, J. Matoušek, and V. Radová. *Mluvíme s počítačem česky*. Academia, Prague, 2006.
- [13] Takeshi Saitou, Masataka Goto, Masashi Unoki, and Masato Akagi. Speech-to-singing synthesis: Converting speaking voices to singing voices by controlling acoustic features unique to singing voices. In *Applications of Signal Processing to Audio and Acoustics, 2007 IEEE Workshop on*, pages 215 –218, oct. 2007.
- [14] P. Salvo Rossi, F. Palmieri, and F. Cutugno. Inversion of f0 model for natural-sounding speech synthesis. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03). 2003 IEEE International Conference on*, volume 1, pages I–520–I–523 vol.1, April 2003.
- [15] SharkD. *Image:Musical frequency diatonic.svg*, August 2008.
http://upload.wikimedia.org/wikipedia/commons/8/8e/Music_frequency_diatonic_scale-.svg
- [16] Julius O. Smith. *Introduction to Digital Filters with Audio Applications*. <http://ccrma.stanford.edu/~jos/filters/>, accessed (date accessed). online book.

Prílohy

Priložené médium obsahuje:

- Výslednú aplikáciu
- Ukážky zvukových výstupov
- Súbor s pokynmi pre používanie média
- Túto prácu vo formáte pdf.