IMPLEMENTACE AUTOMATIZOVANÉHO MĚŘENÍ HRTF V MATLABU 2

O. Šupka, F. Rund

Katedra radioelektroniky, fakulta elektrotechnická
České vysoké učení technické v Praze, Česká republika

Abstrakt

HRTF (Head Related Transfer Function) je často používaným prostředkem k vytvoření virtuální reality. Tento článek se zabývá implementací automatizovaného systému pro rychlé měření HRTF v prostředí MATLAB. Velká pozornost byla věnována návrhu GUI, které bude sloužit k ovládání celého systému. To zahrnuje ovládání otočné židle v horizontální rovině, generování a příjem měřicích signálů přes externí zvukovou kartu a dále zpracování získaných dat a jejich export.

1 Úvod

Vytvoření zvukové virtuální reality [1] s využitím sluchátek je často založeno na tzv. HRTF. Tato přenosová funkce, případně, v časové oblasti, impulzní odezva HRIR (Head Related Impulse Response), popisuje směrově závislou filtraci zvuku způsobenou tělem, hlavou a boltcem, z čehož vyplývá, že je individuálně závislá, což komplikuje její využití. Pro věrnou simulaci je třeba změřit HRTF přímo danému jedinci, nebo alespoň vytvořit dostatečně obsáhlou databázi, aby si subjekt dokázal vybrat co nejvhodnější banku.

Jedna z nejrozšířenějších metod měření HRTF [2], využívaná i na našem pracovišti, spočívá v umístění miniaturních binaurálních mikrofonů do uší subjektu a změření impulsních odezev pro zvuk přicházející z různých směrů. Proto je nutné vytvořit síť bodů měření ve tvaru koule, přičemž středem je hlava subjektu. Je zřejmé, že pro věrnou simulaci je požadována hustá síť bodů měření, což znamená velké množství prováděných měření. Data v neměřených částech sítě mohou být částečně dopočítána interpolací [3], ale i přesto jde o časově náročný experiment.

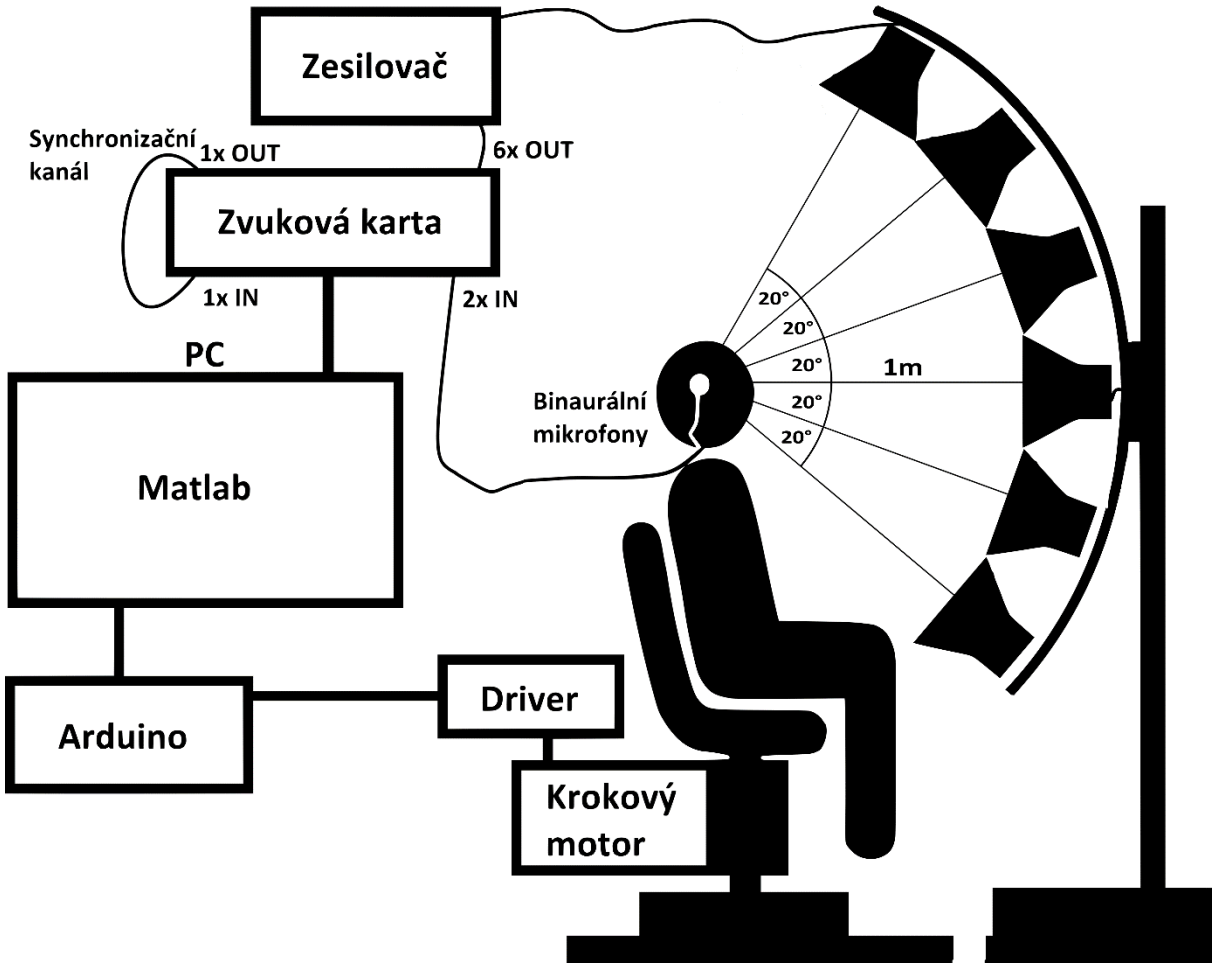
V tomto článku je popisovaná implementace systému pro měření HRTF s využitím prostředí MATLAB. Implementovaný systém se snaží o co nejrychlejší a zároveň dostatečně podrobné měření HRTF. Systém je navržen tak, aby byl schopen provádět kompletní měření, včetně následných výpočtů a exportu do příslušných souborových formátů, bez nutnosti použití dalšího softwaru.

2 Popis systému

Pro měření se nabízejí dvě možnosti – polohovat zdroj zvuku, nebo polohovat subjekt. V naší implementaci je v horizontální rovině polohován subjekt a ve vertikální rovině zdroj zvuku. Pro polohování subjektu byla zhotovena otočná židle [4], která je poháněna krokovým motorem řízeným přes driver M752 a vývojovou desku Arduino Uno. Polohování zdroje dosud probíhá ručně, ve vývoji je však sloup, na kterém bude upevněno několik reproduktorů, každý pro jinou elevaci, což usnadní měření a při použití sofistikovanějších měřicích metod může i velmi značně zkrátit čas nutný k měření.

Z důvodu použití dvojice kondenzátorových mikrofonů vyžadujících fantomové napájení a mikrofonní přezsilovač, soustavy šesti samostatných reproduktorů a i kvůli motivaci použití vyšší vzorkovací frekvence za účelem zkrácení času měření, je nutné použití externí zvukové karty, která zvládne zmíněné požadavky. V naší implementaci používáme kartu RME Fireface 400.

Před samotným měřením musí systém vygenerovat zvolený měřicí signál. Během samotného měření je pak úkolem přehrát vygenerovaný signál z příslušného reproduktoru či reproduktorů, zaznamenat přes binaurální mikrofony, otočit subjektem a tento proces opakovat dle předvolené měřicí sítě. Po měření následuje analýza naměřených dat – výpočet impulsních odezev a přenosových funkcí, s tím je spojeno okénkování za účelem eliminace nežádoucích odrazů, a korekce dle charakteristik vysílačů a přijímačů.



Obr.1. Schéma měřicího sytému.

3 Implementace systému

3.1 Měřicí metody

Úkolem měření je získat sadu impulsových odezev, resp. přenosových funkcí. Impulsová odezva je definována jako odezva lineární soustavy na Diracův impulz, ale pro praktické měření se používá řada metod, které nejsou založeny na přímé realizaci tohoto impulzu. Při výběru vhodné metody musíme uvážit několik aspektů. Jde nám především o relativně rychlé, ale přesné měření v prostorách, které nemusejí být akusticky přizpůsobeny. Pro měření byly vybrány primárně metody SineSweep [5] a MLS [6], které jsou v dnešní době jedny z nejpopulárnějších. Musíme uvažovat, že se jedná o LTI soustavu.

SineSweep je metoda využívající k měření přelaďovaný sinus (většinou logaritmicky – zvláště pro naší aplikaci, jelikož lidské slyšení je logaritmické). Její velkou výhodou je, mimo jiné, že umožňuje měřit a oddělit harmonické zkreslení, což pro nás může být velkým přínosem. Z teorie zjistíme, že impulsní odezvu získáme konvolucí signálu na výstupu soustavy s inverzním filtrem. Inverzní filtr je časově převrácený signál na vstupu soustavy, který je navíc tlumen exponenciálou (pro logaritmicky přelaďovaný sinus). Nevýhodou metody může být například pre-ringing artefakt, který se projevuje jako zvlnění v impulsní odezvě.

V prostředí MATLAB jsme pro generování signálu vytvořili funkci `generate_sine_sweep.m` (rovněž pak pro analýzu `analyze_sine_sweep.m`), kde je signál vypočten dle definičního vztahu, a několik jeho posledních a prvních vzorků je tlumeno příslušnou polovinou Hannova okna. Důsledkem tohoto postupného tlumení konce a začátku, je částečné potlačení zmíněného pre-ringing artefaktu. Sweep je generován od 100 do 22 000 Hz, což je pro naše účely naprosto dostačující. Funkce umožňuje generování posloupnosti několika sweepů, jelikož je pak při měření vhodné používat více opakování a výsledky průměrovat, kvůli možné chybě

způsobené například hlukem v pozadí. Zároveň je třeba zmínit, jak je třeba volit délku jednoho sweepu. S delším sweepem se nám zvyšuje SNR, ale měření bude trvat déle.

Další zmíněnou metodou je MLS – Maximum Length Sequence. Je to pseudonáhodná binární posloupnost délky $N = 2^m - 1$, kde m je celé číslo. Tato posloupnost má několik výhodných vlastností. Autokorelační funkce je rovna Diracově impulsu (se zanedbatelnou chybou, která je dána tím, že sekvence má lichý počet vzorků) a posloupnost je odolná vůči nekorelovaným šumům, což je velkou výhodou. Impulsní odezvu získáme cyklickou korelací výstupu soustavy se sekvencí na vstupu. Opět platí, že je vhodné používat několik sekvencí, přičemž první sekvence slouží pouze k vybuzení soustavy a impulsní odezva se z ní nepočítá. V naší implementaci využíváme funkce z MLS toolkitu [6], kde je implementována Hadamardova transformace pro urychlení výpočtu korelace.

3.2 Komunikace s otočnou židlí

Ovládání židle je zprostředkováno přes vývojovou desku Arduino Uno, za kterou následuje driver pro krokové motory. Arduino se k PC připojuje přes USB, ale umí emulovat sériový port, což nám značně ulehčuje činnost, a proto bylo také k této aplikaci vybráno. Celá komunikace mezi Matlabem a Arduinem (resp. židlí) je založeno na posílání hodnot 0-255, které Arduino dle kódu vyhodnotí a pošle příslušný příkaz driveru a ten se už stará o činnost krokového motoru. Z Matlabu se k Arduino přistupuje jako ke standardnímu sériovému objektu, baudrate je nastavena na 115200 bitů za sekundu.

Hodnota	**Operace**
1-230	otočení o daný počet kroků
238	zapnout motor
239	vypnout motor
240-249	regulace rychlosti (resp. perioda pulzů) – od 9 do 45 sekund na otáčku
250	nastavení směru otáčení po směru hodinových ručiček
251	nastavení směru otáčení proti směru hodinových ručiček
255	inicializace – nastavení základních instrukcí v Arduino a ověření úspěšného připojení

Tab.1. Přehled instrukcí pro Arduino.

Jak lze vyčíst z Tab. 1, maximální počet kroků (resp. mikrokroků) poslaných v jedné instrukci je 230. Motor potřebuje s nastaveným mikrokrokováním vykonat 10800 mikrokroků pro jednu otočku židle (více podrobností lze nalézt v [4]). V případě potřeby vykonat vyšší počet kroků se jednoduše pošle více instrukcí za sebou. Arduino je pak provede postupně. V případě poslání velkého počtu příkazů rychle po sobě je Arduino všechny nezvládne vyhodnotit, proto je třeba odesílat je s časovým odstupem, který bude odpovídat tomu, o kolik kroků se motor otáčí. Tuto dobu v sekundách počítám následovně:

$$t = \left(\left(11 - \frac{slider}{0{,}2}\right) * 200\right) * 10^{-6} * 2 * \mathrm{abs}(angle) * 30 \qquad (1)$$

Pro vysvětlení je třeba se zabývat tím, jak funguje regulace rychlosti. Mluvím o rychlosti, ale měněn je vlastně interval mezi pulzy, které jsou posílány z Arduina do driveru. K otočení o jeden mikrokrok je třeba poslat hodnoty 0 a 1. Cyklus pro otáčení vypadá v jazyku Wiring v Arduinu takto.

```
for (int i = 0; i < val; i++){
   digitalWrite(3,HIGH);
   delayMicroseconds((1000*speed)/10);
   digitalWrite(3,LOW);
   delayMicroseconds((1000*speed)/10);
}
```

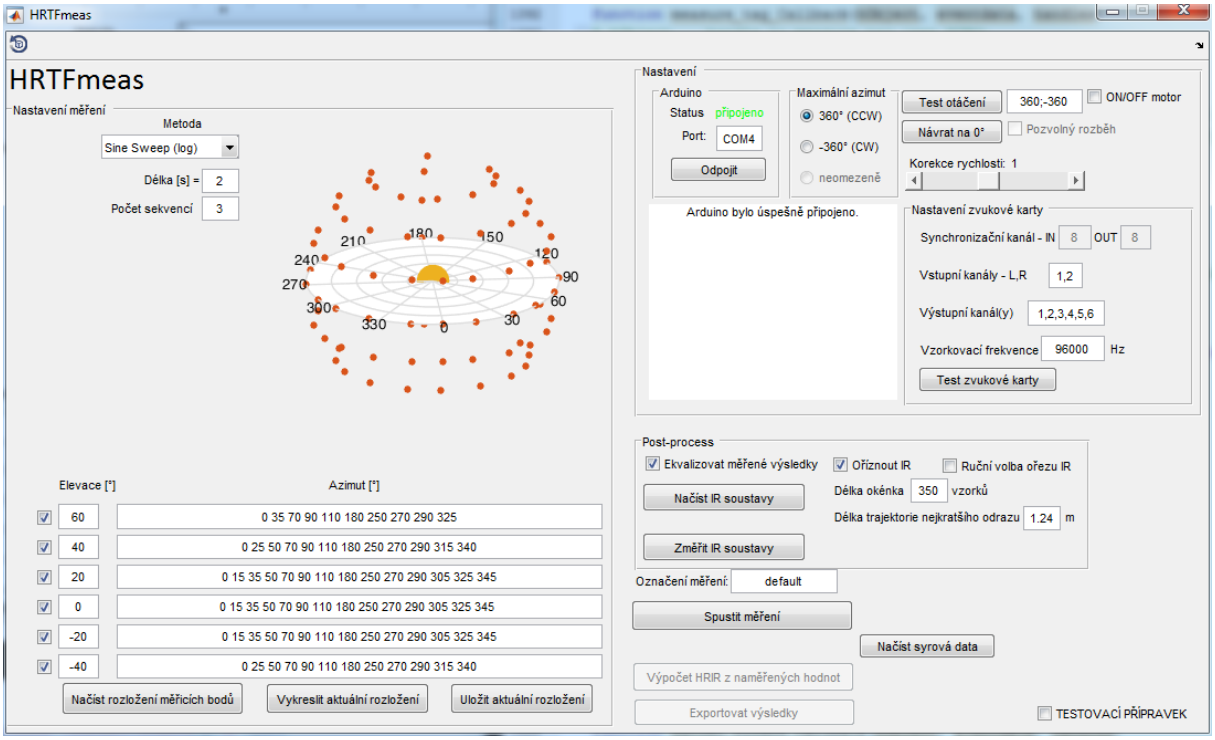
Mezi každou změnou hodnoty je defaultně zpoždění 1 ms, které je násobenou koeficientem, který je čten právě z hodnot 240-249 jako `speed1=(val-239)*2`, kde `val` je poslaná hodnota z Matlabu. Koeficient tedy může nabývat hodnot 2 - 20. Defaultní hodnota je 10. Tyto hodnoty se nastavují posuvníkem v GUI. Při změně hodnoty se okamžitě odešle instrukce do Arduina. Z tohoto lze odvodit první část vzorce (1), která poté násobena $30 * \mathrm{abs}(angle)$, jelikož otočení židle o 1° odpovídá 30 mikrokrokům motoru. Hodnoty zpoždění byly zvoleny experimentálně a v praxi bude ověřeno, jaká rychlost bude pro měření optimální.

3.3 Komunikace se zvukovou kartou

Komunikace se zvukovou kartou RME Fireface 400 je zajištěna funkcemi `dsp.AudioPlayer` a `dsp.AudioRecorder` z DSP system toolboxu prostředí MATLAB. Implementace využívá komunikaci přes ASIO, čímž je snížena latence karty. Abychom byli schopni snadno vypočítat impulsní odezvy z naměřených hodnot, je třeba zajistit synchronizaci mezi přehráváním a nahráváním. Synchronizace je realizována přes jeden přímo propojený vstupní a výstupní kanál zvukové karty. Korelací zaznamenaného signálu s originálem signálu pak z maxima korelace určíme zpoždění, které je využito ke kompenzaci zpoždění v měřicích kanálech.

3.4 GUI

Pro usnadnění měření bylo vytvořeno pro ovládání systému při měření a i následnou analýzu vytvořeno GUI (viz. Obr.2).



Obr.2. GUI pro obsluhu systému.

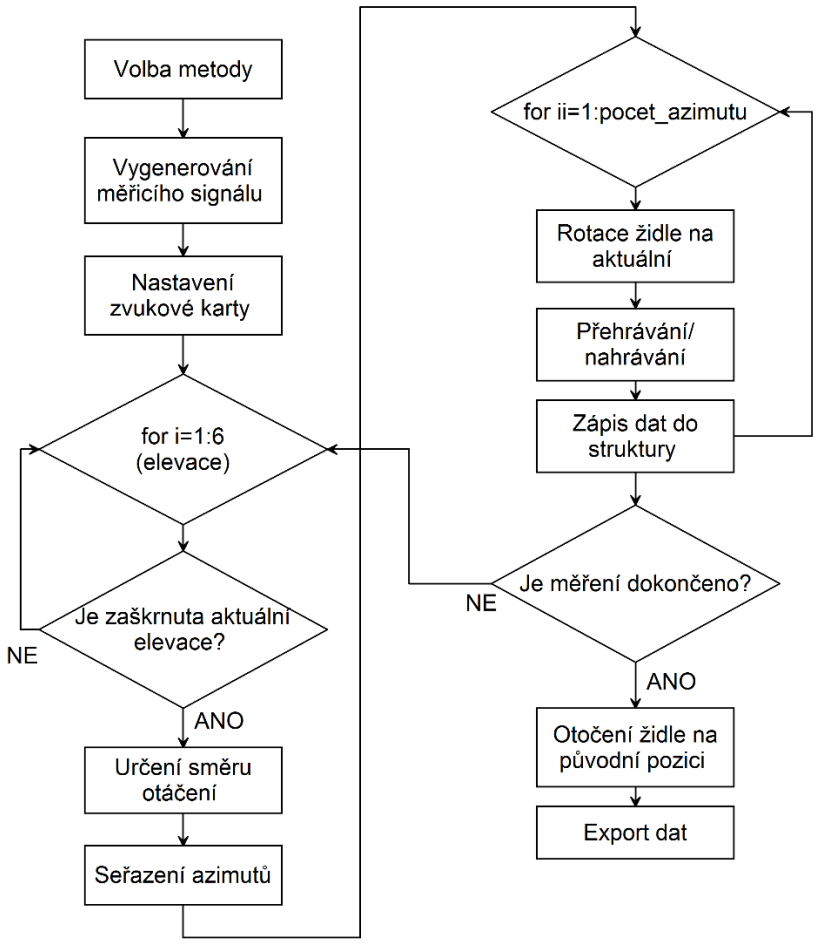
Levá polovina okna slouží k výběru metody a jejích parametrů a zároveň k volbě, konfiguraci a správě měřicích sítí (resp. distribuce měřicích bodů kolem hlavy subjektu). Výchozí měřicí síť byla navržena dle [7]. Pro usnadnění práce lze nakonfigurovanou měřicí síť snadno uložit, resp. načíst ze souboru, kde je uložena jako matice *M* x *N*, kde *M* je počet elevací a *N*-2 je max. počet azimutů. V prvním sloupci jsou uloženy hodnoty azimutů a vektor azimutů je navíc ukončen číslem 361 pro jasnou detekci konce.

Pravá horní polovina je věnována komunikaci s periferiemi. V prvním boxu se připojuje Arduino na příslušném COM portu, vedle něj se volí maximální možný azimut, který je volen z důvodu omezení naší otočené židle, což jsou kabely, které nemohou být zapuštěny do podlahy a zároveň je nelze se židlí přejíždět – tj. musíme vždy před měřením zvolit, kterým směrem se bude židle otáčet. Tento problém poněkud komplikuje měření, což bude rozebráno později. Vpravo nahoře je obsluha otočné židle – na stisknutí tlačítka „Test otáčení“, židle vykoná sekvenci rotací uvedenou

v text boxu vedle. Tick box „ON/OFF motor“ slouží k vypnutí/zapnutí motoru, což se může hodit při pauzách mezi měřením, jelikož motor se dosti hřeje, občas produkuje rušivé zvuky a také je s vypnutým motorem možné ruční polohování židle – může tak být snadno nastavena výchozí pozice pro měření. Tlačítko „Návrat na 0°“ slouží k otočení židle na původní pozici, za tímto účelem je v kódu neustále mapována aktuální poloha židle. O posuvníku na korekci rychlosti jsem se již zmínil v kapitole 3.2, je třeba jen upřesnit, že zobrazovaná hodnota je desetina koeficientu `speed`. Vpravo níže je box na obsluhu zvukové karty, nastavení parametrů a její test. Vlevo je text box informující o aktuálním dění.

Níže je okno pro nastavení parametrů zpracování naměřených signálů. První je možnost ekvalizace dle charakteristik zdroje a přijímače, což je nezbytná součást měření. Druhou záležitostí je ořez impulsové odezvy, resp. odrazů. Ve známém ověřeném prostředí lze volit pevnou vzdálenost ořezu, buď ve vzorcích, nebo dle skutečné vzdálenosti (hodnoty jsou samozřejmě přepočítávány dle vzorkovací frekvence), nebo je možné impulsové odezvy oříznout ručně (jedna volba pro celou sadu, takže je vhodné radši volit s menší rezervou, jelikož se pozice mikrofonů během měření mění). Poslední část GUI slouží už jen ke spouštění dílčích procesů a označení měření, pro přehlednost (například jméno subjektu, atd.).

3.5 Měření



Obr.3. Procesní diagram měření.

Po nastavení parametrů a připojení periferií může být spuštěno měření. Nejprve čte program nastavení z GUI, tj. volbu metody, její parametry, atd. Následně generuje měřicí signál (viz 3.1), potom nastavuje dle zvolených a dalších parametrů zvukovou kartu. Poté začíná měřicí sekvence. Ta se skládá zejména ze dvou `for` cyklů, kde vnější cyklus odpovídá počtu měřených elevací (resp. je pevně nastaven na 6, ale vzápětí následuje podmínka `if` na ověření, zda je aktuální elevace zvolena k měření, případně se přesune na konec cyklu a následuje měření pro další elevaci) a vnitřní cyklus počtu měřených azimutů. Následuje určení směru otáčení, což v zásadě není nutnost, ale měření to

zrychlí. Narážíme zde totiž na problém zmíněný již dříve, a to, že se židle není schopná otáčet přes 360° kvůli přívodním kabelům. Sekvence by tedy mohla probíhat tak, že se při měření bude židle pohybovat stále jedním směrem, ale po měření jedné elevace by se musela vracet zpět. S aktuální implementací se pro jednotlivé elevace směr rotace střídá, není tak nutné otáčení zpět (jen na konci měření v případě lichého počtu elevací). Následuje seřazení azimutů, aby v případě, že nejsou srovnány v GUI, nemusela židle rotovat tam a zpět, ale stále pouze jedním směrem. Poté se dostáváme do vnitřního cyklu, kde už probíhá samotné měření - tj. rotace židle na zvolený azimut, přehrání a záznam měřicího signálu a uložení.

Formát dat jsem zvolil následovně. Jedná se o strukturu ve formátu `subjekt_raw_data.ehodnota_elevace.ahodnota_azimutu.ucho` (například `honza_raw_data.e20.a60.left`). `Left`, resp. `right`, je už vektor obsahující zaznamenaný signál. V této struktuře jsou navíc v `subjekt_raw_data.metadata` uloženy informace o parametrech měření – tzn. vzorkovací frekvence, typ měřicího signálu a jeho parametry. Tyto informace jsou nutné pro analýzu (nebo ji aspoň usnadňují), kdyby se data zpracovávala ihned, tak by tohoto ukládání nebylo třeba, ale není špatné mít i syrová data k analýze (např. kvůli hledání problémů). Ve finále se tedy jedná o jediný soubor ve formátu `*.mat`. Sada impulsových odezev (resp. přenosových funkcí) je po analýze ukládána do struktury stejné architektury, jen se změnou z `raw` na `hrir` (resp. `hrtf`).

4 Závěr

Na základě předchozí publikace [8] byl vytvořen samostatný komplexní systém pro měření HRTF, který je schopen pracovat v akusticky nepřizpůsobených podmínkách a bez použití externího softwaru. Systém mohl být realizován díky dokončení konstrukci otočné židle, která je jeho nedílnou součástí.

V nejbližší době budou zahájena první měření a systém tak bude důkladně otestován. Aktuálně pracujeme na výrobě sloupu s reproduktory, který nám značně ulehčí práci ve vertikální rovině, jelikož momentálně musí být zdroj polohován ručně (kód je tomu přizpůsoben). S touto sestavou je spojená budoucí implentace MESM (Multiple Exponential Sweep Method) [9], která by mohla měření zrychlit teoreticky až šestkrát (v závislosti na počtu použitých reproduktorů).

Poděkování

Tato práce byla podpořena grantem Studentské grantové soutěže ČVUT č. SGS14/204/OHK3/3T/13.

Reference

[1] S. Carlile. Virtual auditory space: Generation and Applications. 1 st ed. Austin, TX: RG Landes, 1996.

[2] G. Enzner, Ch. Antweiler, S. Spors. Trends in Acquisition of Individual Head Related Transfer Functions. In: The Technology of Binaural Listening. Springer Verlag Berlin 201, pp. 57-72.

[3] R. Duraiswaini, D. N. Zotkin, N. A. Gumerov. Interpolation and range extrapolation of HRTFs. In: *Acoustics, Speech, and Signal Processing, 2004*. Proceedings. (ICASSP '04). IEEE International Conference on , vol.4, no., pp.iv-45-iv-48 vol.4, 17-21 May 2004

[4] O. Šupka. System for HRIR measurement – turnable chair desing. In: *Poster 2015*. Prague, May 14.

[5] A. Farina. Simultaneous Measurement of Impulse Response and Distortion with a Swept-Sine Technique. *Audio Engineering Society Convention 108.* 2000. http://www.aes.org/e-lib/browse.cfm?elib=10211 [online 14.10.2015].

[6] M. R. P. Thomas. MLS Theory. [online] 12.3.2014. http://www.commsp.ee.ic.ac.uk/˜mrt102/projects/mls/MLS%20Theory.pdf.

[7] T. Lindner. Optimalizace měření a zpracování HRTF. In: *21st Annual Conference Proceedings Technical Computing Prague 2013*. pp. 191-197.
[8] O. Šupka, F. Rund, J. Bouše. Implementace automatizovaného měření HRTF v Matlabu. In: *Technical Computing Bratislava 2014*, pp. 61-65.
[9] P. Majdak, P. Balazs, B. Laback. Multiple Exponential Sweep Method for Fast Measurement of Head-Related Transfer Functions. *J. Audio Eng. Soc*., 2007, vol. 55, no. 7/8, p. 623 – 637.

Bc. Ondřej Šupka
Katedra radioelektroniky, FEL ČVUT v Praze, Technická 2, 166 27 Praha 6, ČR
email: supkaond@fel.cvut.cz

Ing. František Rund, Ph.D.
Katedra radioelektroniky, FEL ČVUT v Praze, Technická 2, 166 27 Praha 6, ČR
email: xrund@fel.cvut.cz