VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

PSYCHOAKUSTICKÁ MĚŘENÍ BINAURÁLNÍCH VLASTNOSTÍ LIDSKÉHO SLUCHU

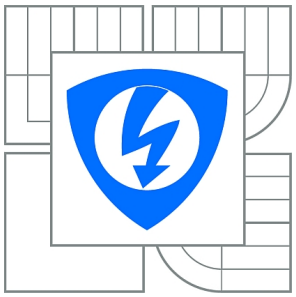
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE Bc. OTA NOVOTNÝ
AUTHOR

BRNO 2010

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

PSYCHOAKUSTICKÁ MĚŘENÍ BINAURÁLNÍCH VLASTNOSTÍ LIDSKÉHO SLUCHU

PSYCHOACOUSTICAL MEASUREMENT OF BINAURAL HEARING CHARACTERISTICS

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

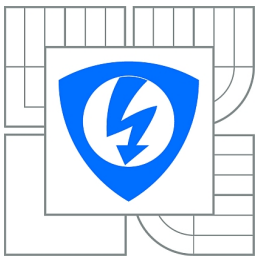
AUTOR PRÁCE — Bc. OTA NOVOTNÝ

AUTHOR

VEDOUCÍ PRÁCE — Ing. JIŘÍ SCHIMMEL, Ph.D.

SUPERVISOR

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Bc. Ota Novotný	***ID:***	22032
Ročník:	2	***Akademický rok:***	2009/2010

NÁZEV TÉMATU:

Psychoakustická měření binaurálních vlastností lidského sluchu

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte metody subjektivního vyhodnocování binaurálních sluchových vjemů a zvolte měřicí metodu vhodnou pro psychoakustický experiment zaměřený na subjektivní hodnocení interaurálních časových a intenzitních diferencí a směru přicházejícího zvuku v horizontální i mediální rovině skupinou posluchačů. Tyto metody implementujte v jazyce Java jako aplikaci s vlastním grafickým uživatelským rozhraním. Pro 3D panorámování virtuálního zdroje zvuku využijte korelaci zvukového signálu se změřenými impulsními chrakteristikami umělé hlavy.

DOPORUČENÁ LITERATURA:

[1] Melka, A. Základy experimentální psychoakustiky. Akademie múzických umění v Praze, 2005. ISBN 80-7331-043-0
[2] Zwicker, E., Fastl, H. Psychoacoustics, Facts and Models, 2nd updated Ed. Springer, 1999. ISBN 3-540-65063-6
[3] Blauert, J. Spatial Hearing. The MIT Press, 1997. ISBN 0-262-02413-6
[4] Streicher, R., Everest, F., A. The New Stereo Soundbook, 3rd ed. Audio Engineering Asociates, 2006. ISBN 978-0-9665162-1-0

Termín zadání:	29.1.2010	***Termín odevzdání:***	26.5.2010

Vedoucí práce: Ing. Jiří Schimmel, Ph.D.

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

UPOZORNĚNÍ:

Anotace

Tato diplomová práce pojednává o problematice binaurálního slyšení (tzn. slyšení pomocí obou uší), schopnosti lidského sluchu lokalizovat zdroj v trojrozměrném prostoru a o parametrech, které tuto schopnost ovlivňují. Ve druhé části se zaměřuje na psychoakustický experiment, jeho hlavní rysy a chyby, které mohou při experimentu nastat a které ovlivňují věrohodnost závěrů. Blíže je rozvedena metoda párového srovnávání. Poslední částí této práce je vlastní řešení psychoakustického experimentu v jazyce Java. Program má disponovat grafickým rozhraním umožňujícím zaregistrovat nového uživatele a provést psychoakustický experiment. Ten spočívá v tom, že aplikace náhodně zvolí pozici předem definovaného zdroje zvuku na předem definovaném rozsahu a na stisk tlačítka jej přehraje uživateli do sluchátek. Úkolem uživatele je nastavit ovládací prvky aplikace, které vyjadřují pozici virtuálního zdroje, tak, odkud zvuk přichází. Na stisk druhého tlačítka přehraje tentýž zvuk z pozice, kterou uživatel nastavil. Párovým srovnáváním pak uživatel upravuje pozici do té doby, až oba zvuky přichází ze stejné pozice. Aplikace má poté na stisk dalšího tlačítka tyto hodnoty uložit k pozdějšímu zpracování. Dále jsou popsány principy tvorby testovacích signálů (harmonický tón, úzkopásmový šum a zvukový soubor typu wav) a princip 3D panorámování virtuálního zdroje pomocí korelace zvukového signálu se změřenými impulsovými charakteristikami umělé hlavy. Na základě změřených hodnot je v závěru zhodnocena schopnost lidského sluchu lokalizovat pozici zdroje v závislosti na parametrech zvuku.

Klíčová slova

psychologická, akustika, psychoakustika, binaurální, sluch, lokalizace, měření, experiment, java

Abstract

This diploma thesis deals with a binaural hearing issues (it means hearing by both of ears), a human hearing ability to locate position of a sound source at three-dimensional space and parameters that affect this ability. In the second part, it focuses on psychoacoustic experiment, its main features and errors that can occur and which affect a results credibility. Method of pair comparisons is described more closely here. The last part of this thesis describes a technical solution of experiment in Java environment. The application should have a graphical interface and should be able to register a new user and perform a psychoacoustical experiment. The process of experiment is following. The aplication selects a random position of defined virtual sound source on the defined range and it plays this sound into headphones on button click. The users task is to set the application controls, representing a virtual sound source position, that way, where the user hear the sound come from. On another button click the application plays the same sound, but this sound comes from application controls set position (set by user). User compares this pair of sounds and modifies the position of second sound source until these two positions are same. The application stores these results for later processing on another button click. Principles of generating testing sound sources (sine wave, narrowband noise and sound file with wav extension) and their 3D positioning by measured head model impulse responses correlation are described thereinafter. An ability of human hearing system to locate a virtual sound source in dependence on sound parameters is discussed in conclusion.

Keywords

psychological, acoustic, psychouacoustics, binaural, hearing, localization, measurement, experiment, java

Citace vysokoškolské kvalifikační práce

NOVOTNÝ, O. *Psychoakustická měření binaurálních vlastností lidského sluchu.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2010. 53 s. Vedoucí diplomové práce Ing. Jiří Schimmel, Ph.D.

Prohlášení

Prohlašuji, že svou diplomovou práci na téma „Psychoakustická měření binaurálních vlastností lidského sluchu“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne ....

Podpis autora

Poděkování

Děkuji vedoucímu práce Ing. Jiřímu Schimmelovi, Ph.D. za velmi užitečnou metodickou pomoc a cenné rady při zpracování diplomové práce.

V Brně dne ....

Podpis autora

1 Obsah

2 Seznam obrázků

3 Seznam tabulek

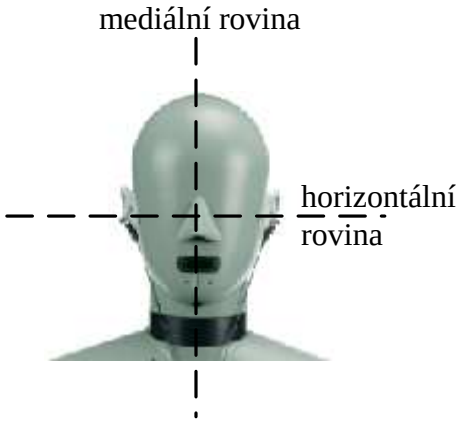
4 Úvod

Cílem této diplomové práce je seznámení se s problematikou binaurálního slyšení (tzn. slyšení pomocí obou uší). Jednou z vědních disciplín, která se touto problematikou zabývá, je psychologická akustika, nebo též zkráceně psychoakustika [1]. Jedná se o mezioborovou vědní disciplínu spojující psychologii a akustiku. Diplomová práce se v první části zaměřuje na analýzu problematiky binaurálního slyšení, především na schopnost člověka určit směr zdroje příchozího zvuku v horizontální i mediální rovině, jeho vzdálenost a dále popis nejdůležitějších faktorů, které tuto schopnost ovlivňují. Druhá část pojednává o experimentu jako takovém, o jeho náležitostech a o vyhodnocení jeho výsledků. Poslední část práce potom popisuje řešení experimentu zjišťujícího schopnost pozorovatele určit azimut a elevaci zdroje příchozího zvuku, řešeného v jazyce Java.

5 Lokalizace zdroje zvuku

Vnímání zvuku oběma ušima je základním předpokladem pro prostorový zvuk. Existuje řada mechanismů, jakým sluch analyzuje příchozí zvuk, jehož směr je popsán horizontálním úhlem (*azimut*) a vertikálním úhlem (*elevace*). Každý z těchto mechanismů poskytuje člověku nějakou informaci o zdroji zvuku a jejich spojením pak člověk dokáže více či méně určit směr příchozího zvuku, jeho vzdálenost, popř. i velikost zdroje.

Než se ale budeme zabývat samotnými mechanismy, je nutné definovat roviny popisující polohu zdroje zvuku. Jedná se o *horizontální rovinu* a *mediální rovinu* (prochází osou souměrnosti hlavy). Na obr. 5.1 jsou tyto roviny znázorněny.



Obr. 5.1: Definice rovin [2].

5.1 Parametry ovlivňující schopnost lokalizace

Každý z následujících parametrů má svůj význam a svou váhu při určování směru příchozího zvuku a lidský sluch pak při jeho analýze používá kombinaci metod zohledňujících více či méně jednotlivé parametry. Minimální postřehnutelná změna v úhlu dopadu zvukové vlny (*minimální poslechový úhel*) je závislý především na následujících parametrech:

a) *Kmitočet* – je jedním z hlavních parametrů ovlivňujících schopnost lokalizace. Na kmitočtech, jejichž vlnová délka je srovnatelná s šířkou hlavy, je zejména u harmonických signálů schopnost lokalizace znatelně horší než u ostatních kmitočtů. Minimální poslechový úhel (diskriminační práh) je malý (kolem 1º) pro nízké a vysoké kmitočty a velký (10º i více) pro kmitočty mezi 1500 a 2000 Hz. Pro zdroj přímo vpředu je menší a pro zdroj nacházející se po straně hlavy je velký [3].

b) *Spektrální složení zvuku* – je dalším, neméně významným parametrem ovlivňujících schopnost člověka určit směr příchozího zvuku. Jako zdroj zvuku při těchto experimentech může být harmonický signál, úzkopásmový šum, popř. hluk. Experimentálně bylo zjištěno, že

lidský sluch je při lokalizaci v trojrozměrném prostoru (dále jen prostor) přesnější u složených zvuků nebo hluků než u čistých tónů.

c) *Věková kategorie* – lidský sluch se s postupem věku stává méně citlivým na nízkých a vysokých kmitočtech a proto je na nich lokalizace obtížnější. Na druhou stranu se ale tato schopnost vyvíjí během života zkušenostmi a spojováním zrakových a sluchových vjemů [4].

d) *Stálost pozice zdroje zvuku / pozorovatele* – pro přesnost lokalizace zdroje v prostoru je u lidského sluchu zásadní, zda se daný objekt pohybuje, popř. zda pozorovatel může otáčet hlavou, či jsou oba elementy statické. U pohybujících se zdrojů, popř. pozorovatele se mění fáze zvukové vlny, která dopadá do obou uší a tím usnadňuje lokalizaci. Drobné pohyby hlavou, které pomáhají této identifikaci, uskutečňujeme nevědomky [4].

e) *Akustické pole* – svou roli hraje i akustické pole, ve kterém lokalizace probíhá. Ve volném poli bývá určení směru složitější, než v uzavřené místnosti, kde lidský sluch využívá prvotních odrazů, difúzního dozvuku a jejich kmitočtového spektra a na základě předchozí zkušenosti pak lokalizuje objekt přesněji.

f) *Délka trvání zvuku* – zvuky delšího trvání je snadnější lokalizovat, než zvuky kratšího trvání popř. impulsy.

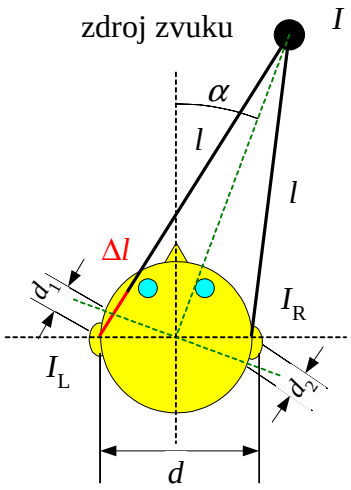
g) *Přechodné jevy* – mohou lokalizaci usnadnit

5.2 Hlavní principy lokalizace v horizontální rovině

Hlavními činiteli při lokalizaci v horizontální rovině jsou rozdíly intenzity a času zvuku příchozího do levého a pravého ucha.

5.2.1 Interaurální časová diference

Lokalizace pomocí interaurální časové diference (Interaural Time Difference, ITD) je založen na rozdílné vzdálenosti zdroje zvuku od obou uší. Leží-li zdroj zvuku v mediální rovině, je vždy vzdálenost k oběma uším stejná. Dojde-li ale k přemístění zdroje zvuku mimo mediální rovinu (přemístěním zdroje, či pozorovatele), dorazí zvuk do jednoho ucha dříve, než do druhého. Na obr. 5.2 je znázorněn pozorovatel a objekt ležící mimo mediální rovinu.



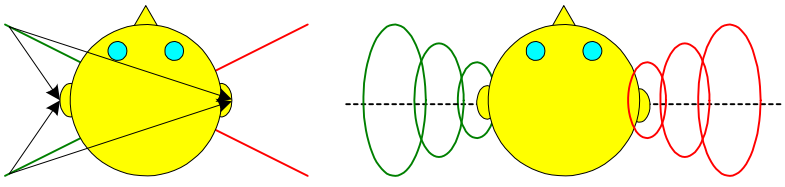
Obr. 5.2: Rozdíl vzdáleností uší od zdroje zvuku [2].

Objekt o intenzitě I (viz obr. 5.2) je vychýlen od mediální roviny pod azimutem α a tím způsobuje binaurální rozdíl Δl, který je příčinou časového zpoždění a útlumu intenzity. Binaurální rozdíl Δl lze popsat rovnicí (4.1) [2],

$$\Delta l = d_1 + d_2 = d \sin\alpha, \tag{4.1}$$

kde d je vzdálenost obou uší. Při výpočtech se používá průměrná hodnota d = 15 cm.

Vzdálenost od obou uší ale není bod v prostoru ani v rovině definovaný jednoznačně. V horizontální rovině si lze tyto body představit jako dva páry polopřímek. Každá z nich vychází z jednoho ucha do bodu zdroje. Pokud by interaurální časová diference byla jedinou metodou lokalizace, nebylo by pro pozorovatele na obr. 5.3 jednoznačné, zda je zdroj zvuku nalevo vpředu, nebo nalevo vzadu. V prostoru jsou pak tyto body tvořeny rotací těchto polopřímek a vzniká tak tzv. *kužel konfúze*.



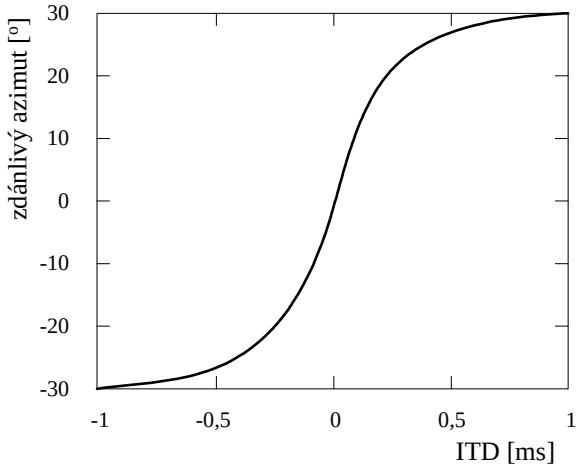
Obr. 5.3: Body se stejným rozdílem vzdáleností od obou uší [2].

U ostrého krátkého ostrého impulsu lze rozeznat časový posun o 0,1 ms. Zvětšuje-li se tento interval nad hodnotu 2 ms, jsou již vnímány dva rozdílné impulsy, každý v jiném uchu. Přesnějším měřením bylo zjištěno, že dva rozdílné zvuky již člověk vnímá již u časového

posunu 630 ms, což je pravděpodobně způsobeno tvarem zvukové vlny zkoumaného impulsu [5].

U zvuků periodických se interaurální časová diference projeví jako fázový rozdíl. Dojde-li k fázovému posunu, je zvuk vnímán uchem, kde fáze zvuku předbíhá. Posun fáze více jak o jednu polovinu trvání periody způsobí, že původně fázově zpožděný zvuk bude rázem vnímán jako fázově předbíhající a zvuk přeskočí do druhého ucha. Směrový účinek tohoto principu se plně projevuje u harmonických tónů asi do 800 Hz. U tónů s vyšším kmitočtem (polovina vlnové délky je menší nebo rovna vzdálenosti obou uší) není dosaženo dostatečného časového posuvu v obou uších, odchylka směru se stává malou a dochází ke konfuzím – zvuky nepatrně kolísají kolem střední roviny, nebo se může zdát, že zvuk přichází z více míst [5].

Na obr. 5.4 je zobrazen průběh zdánlivého azimutu v závislosti na časovém rozdílu signálu v uších. Záporné hodnoty azimutu značí, že se jedná o polohu zdroje nalevo, stejně tak jako záporné hodnoty v časovém rozdílu značí o jakou dobu dorazil zvuk do levého ucha dříve, než do pravého.



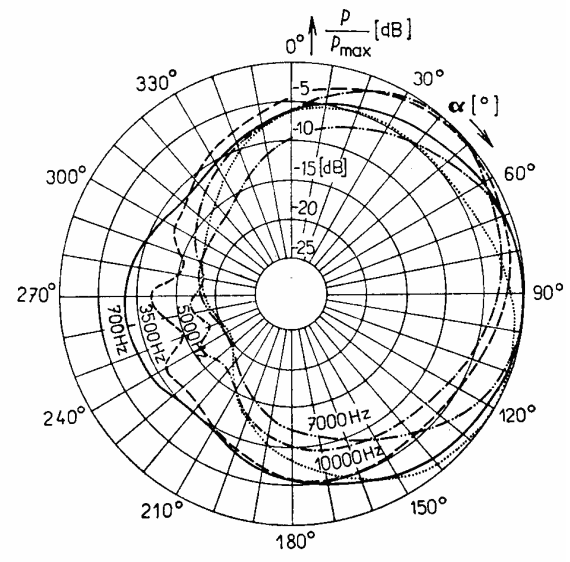
Obr. 5.4: Zdánlivá poloha zdroje zvuku v závislosti na časovém rozdílu signálu v uších [5].

Tento princip je efektivní především u blízkých zdrojů (řádově vzdálenost obou uší), kdy se může projevit fázový rozdíl. U vzdálených zdrojů je fázový rozdíl nepatrný, proto se k lokalizaci využívá dalšího principu, založeného na rozdílu intenzit.

5.2.2 Interaurální intenzitní diference

Lokalizace pomocí interaurální intenzitní diference (Interaural Intensity Diference, IID) je založen na rozdílu intenzit zvuku dopadajících do obou uší. Je-li zdroj zvuku umístěn v mediální rovině, dopadá zvuk do obou uší se stejnou intenzitou. Odchýlením od mediální roviny dojde nejen k časovému (fázovému) rozdílu, ale i k rozdílu intenzity v obou uších. To

je způsobeno jednak tím, že vzdálenost zdroje od odvráceného ucha je větší, než k uchu přivrácenému, ale hlavně akustickým stínem hlavy. Princip lokalizace pomocí IID se na rozdíl od principu ITD projevuje především u vyšších kmitočtů, protože pro hluboké tóny (cca do 200 Hz) není hlava dostatečnou překážkou. Tóny těchto kmitočtů hlavu obejdou a dorazí do obou uší se stejnou intenzitou. Se vzrůstajícím kmitočtem dochází k větším rozdílům intenzit zvuku dopadajícím do ucha v závislosti na azimutu [5] (viz. obr. 5.5)



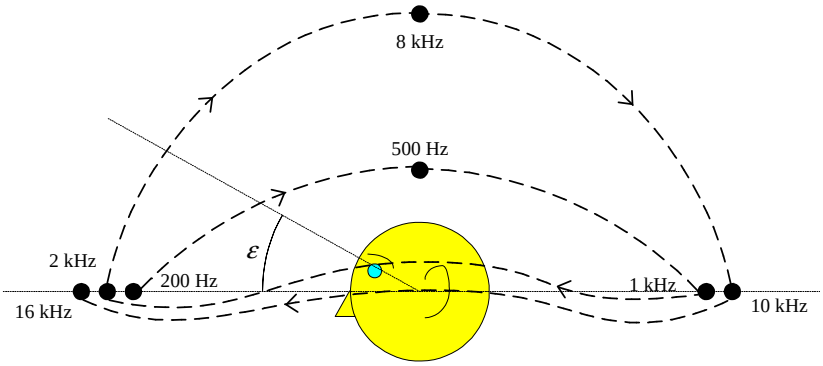
Obr. 5.5: Směrová charakteristika pravého ucha [5].

Akustický stín hlavy tedy způsobuje změnu spektra vnímaného signálu, což rovněž dokresluje představu o poloze zdroje. Lidé, kteří jsou dlouhodobě odkázáni na poslech jedním uchem, jsou díky tomuto mechanismu spolu s předchozí zkušeností schopni do jisté míry určit směr zdroje zvuku. Experimentálně bylo zjištěno, že odchylka lokalizace při binaurálním poslechu je zhruba 3°, při monoaurálním poslechu 30°, pokud má posluchač možnost vykonávat pohyb hlavou a tím porovnávat změny [5].

5.3 Externalizace a určení pozice zdroje zvuku v mediální rovině

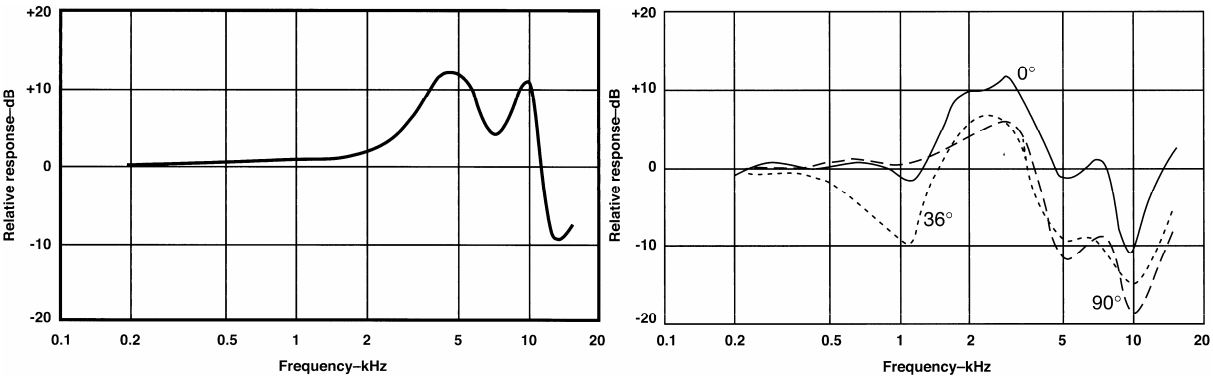
Lokalizace zdroje zvuku v mediální rovině je již obtížnější, než v horizontální rovině. Samotné lokalizaci zvuku předchází tzv. lateralizace, která vychází z principů časové a intenzitní diference popsané výše. Dopadá-li zvuková vlna do obou uší se stejnou fází a intenzitou a s časovým posunem menším než 2 ms, nastává tzv. *splývání* (binaurální fúze) v jediný vjem subjektivně umístěn uprostřed hlavy [5]. Lateralizace má z experimentálního hlediska význam pouze s poslechem na sluchátka – lokalizace v prostoru musí být doplněna tzv. externalizací sluchového vjemu.

Jak již bylo uvedeno výše, nízkým tónům (cca do 300 Hz) nepůsobí hlava dostatečnou akustickou překážku a proto je (zejména u harmonických průběhů) poměrně obtížné rozlišit, zda zvuk přichází zepředu, nebo zezadu. Při těchto experimentech je úsudek chybný téměř ve 40%. Chyba určení směru u harmonických tónů nad 300 Hz klesá zhruba na 15%, jelikož při těchto kmitočtech se již projeví pokles intenzity zdroje zvuku vzadu důsledkem akustického stínu ušního boltce [3]. U složených tónů, které obsahují složky na vysokých kmitočtech je lokalizace značně usnadněna tím, že příchodem takovéhoto zvuku zezadu dojde k odfiltrování vysokých kmitočtů opět následkem akustického stínu ušního boltce. Lokalizace tohoto typu vzniká především na základě zkušeností spojených se zrakovou kontrolou [3]. Sluchový aparát se tedy v případě lokalizace v mediální rovině opírá hlavně o spektrální změny způsobené ušním boltcem a také částečně i odchýlením zvukového kanálu od osy spojující obě uši. Tento princip se ale projevuje teprve u vyšších kmitočtů cca nad 800 Hz (viz. obr. 5.6).



Obr. 5.6: Zdánlivý směr zvukového signálu v mediální rovině v závislosti na kmitočtu [6].

Dalším faktorem externalizace jsou odrazy od ramen a přechodné děje časového průběhu signálu [4]. Na tomto principu se podílí zejména přenosová funkce zvukovodu a vnějšího ucha, která je znázorněna na obr. 4.8. Maxima přenosové funkce zvukovodu jsou v okolí kmitočtů 5 a 10 kHz, což značí fyzikální vlastnosti samotného zvukovodu. Maxima přenosové funkce vnějšího ucha se nachází v oblasti mezi 1 a 5 kHz v závislosti na příchozím směru zvuku. obr. 5.7 b) demonstruje závislost přenosové funkce vnějšího ucha na elevačním úhlu příchozího zvuku. Pro lokalizaci jsou směrodatné pozice minim těchto přenosových funkcí [7].



Obr. 5.7: Přenosová funkce zvukovodu (a) a vnějšího ucha (b) [7].

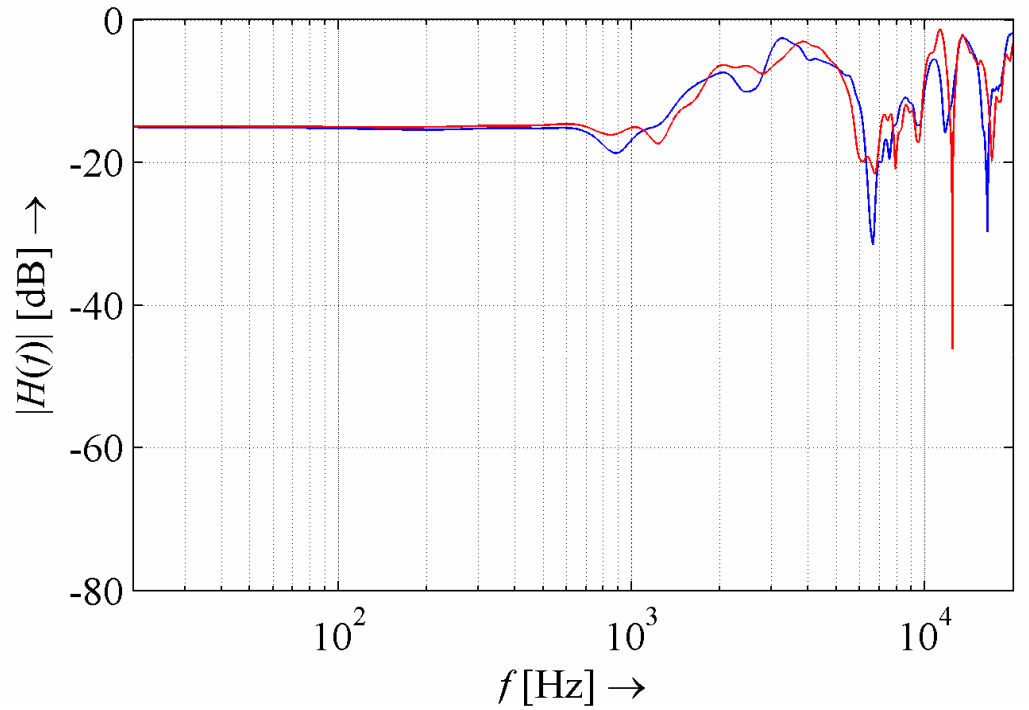
5.4 Přenosová funkce hlavy

Přenosová funkce hlavy (Head-Related Transfer Function, HRTF) je přenosová funkce popisující přenos zvuku od zdroje v dané pozici v prostoru až do lidského ucha. Používá se především k simulaci akustických vlastností místností a prostorových efektů. Přenosová funkce hlavy není závislá na fyzických proporcích daného jedince.

Pro měření přenosové funkce hlavy je potřeba umělou hlavu, reproduktor a je nutno jej provádět v bezodrazové místnosti. Reproduktor přehrává testovací zvuk, nejčastěji pseudonáhodný šum, např. signál MLS (Maximum-Length Sequence) z požadovaného směru. Následně se z naměřené odezvy provede dekonvoluce tohoto šumu. Po změření řady impulsních odezev z různých směrů lze tyto impulsní charakteristiky (Head-Related Impulse Response, HRIR) použít jako koeficienty filtru typu s konečnou impulsní odezvou (Finite Impulse Response, FIR).

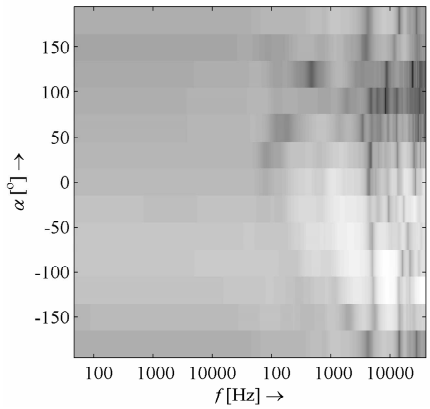
Fourierovou transformací impulsních odezev hlavy potom získáme přenosové funkce hlavy pro dané azimuty a elevační úhly. Výsledky těchto měření pak mohou mít různé podoby, nejčastěji tyto:

a) sada dvojic (pro každé ucho) 2D grafů s azimutem a elevačním úhlem jako parametry (viz obr. 5.8)



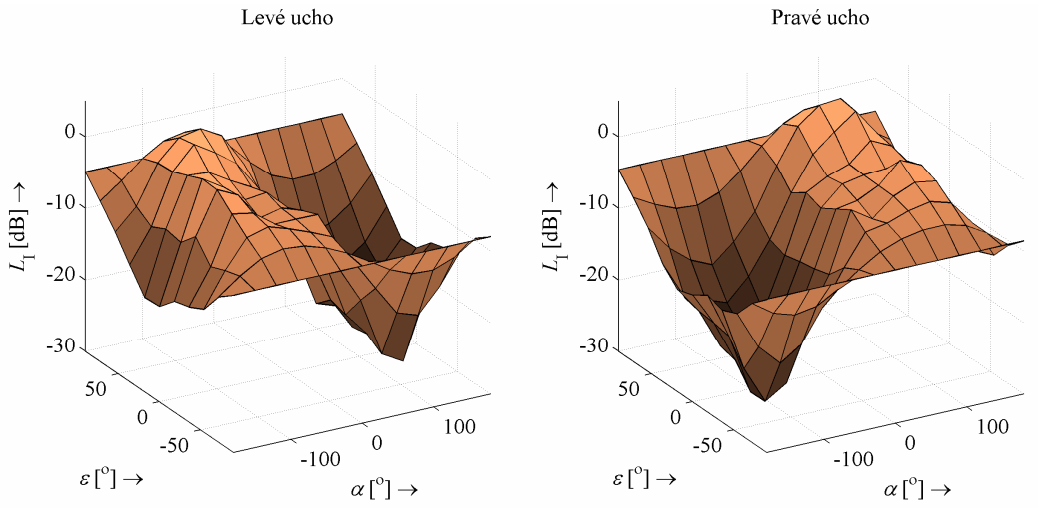
Obr. 5.8: Přenosová funkce hlavy pro levé a pravé ucho pro daný azimut a elevační úhel [2].

b) sada dvojic 3D grafů znázorňujících závislost modulu přenosové funkce na kmitočtu a některého z úhlů (azimut, či elevace) (viz obr. 5.9)



Obr. 5.9: Zobrazení přenosové funkce pro levé ucho pro nulový elevační úhel [2].

c) sada dvojic 3D grafů znázorňující závislost modulu přenosové charakteristiky na azimutu a elevaci při určitém kmitočtu dopadajícího zvuku (viz obr. 5.10)



Obr. 5.10: Závislost přenosové funkce hlavy na azimutu a elevačním úhlu pro kmitočet 1 kHz [2].

6 Psychoakustický experiment

Experiment má původ v oblasti přírodních věd a do psychologie pronikl prostřednictvím fyziologie smyslů. Moderní psychologie, zejména tzv. *behaviorismus* [1] rozšířila pojem psychologie vypracováním některých vlastním metod měření a umožnila tak popsat složité psychologické jevy *kvantitativním* způsobem. Při psychologickém experimentu je ale nutné zabývat se i psychologickými proměnnými, které nelze pozorovat ani měřit v čase a které lze uchopit jen *kvalitativně*. U psychoakustických experimentů často převládá průzkum neboli výběrové šetření, tj. sběr informací od skupiny osob formou dotazníku. V literatuře [1] se píše, že psychoakustický experiment je specifickou formou psychologického experimentu, zaměřenou na sledování účinků zvukových podnětů.

Psychologické experimenty charakterizujeme jako srovnávací experimenty. Při nich vycházíme z úvah, které jsou podložené hypotézami, že za jistých experimentálních podmínek bude možné pozorovat jisté specifické účinky vyvolané těmito podmínkami a že změnami experimentálních podmínek dosáhneme určitých specifických změn účinků a dokazujeme, že tyto účinky opravdu existují.

6.1 Proměnné

Při psychologických experimentech pracujeme se dvěma typy proměnných – závislými a nezávislými.

Závislé proměnné – jsou proměnné, jejichž hodnoty při experimentu zkoumáme.

Nezávislé proměnné – jsou proměnné, jejichž účinky při experimentu zkoumáme a dělíme je na další dvě podskupiny. Nezávislé proměnné, které experimentátor záměrně během experimentu mění, jsou aktivní proměnné popř. manipulativní prediktory. Druhou podskupinou nezávislých proměnných jsou proměnné určené neboli klasifikační. Jedná se o proměnné, které nejsou obsaženy v experimentálních podmínkách a jedná se většinou o individuální atributy jedince, objektu nebo prostředí, které se proto mohou měnit, avšak ne na vůli experimentátora.

Rušivé proměnné – jedná se o proměnné, které mají vztah k závislým proměnným a které mohou zkreslovat úvahy o kauzálním vztahu nezávislé proměnné na závisle proměnnou. Zatímco manipulace se závislou proměnnou a měření nezávislé proměnné nepředstavují v psychoakustice většinou žádné závažnější problémy, kontrola rušivých problémů je poměrně složitá. Její vliv lze omezit např. jejich vyloučením z experimentu (pokud je to možné), zkonstantněním (udržováním jejich hodnot na stálé hodnotě během celého

experimentu), změřením (tzn. v podstatě převedením na nezávislou proměnnou), nebo použitím vhodného plánu experimentu.

Podle literatury [1] je všem psychologickým experimentům je společné, že při nich experimentátor za podmínek jím samotným co nejpřesněji připravených, kontrolovaných a pozměňovaných zjišťuje a zaznamenává (ať kvantitativně či kvalitativně) určité reakce, způsoby chování, projevy nebo prožitky pokusných osob, aby ověřil určitou hypotézu o kauzální závislosti těchto výsledků na experimentálních podmínkách. Experiment lze tedy chápat jako prostředek ke zjišťování kauzálních vztahů mezi proměnnými.

6.2 Opakování

Vlivem působení většího či menšího počtu proměnných na průběh experimentu má za následek variabilitu průměrné hodnoty cílové proměnné. Opakováním experimentu se snažíme tuto variabilitu minimalizovat. V literatuře [1] se píše, že smyslem opakování je snaha snížit u jednotlivých experimentálních jednotek náhodnou variabilitu průměrné hodnoty cílové proměnné a prokázat tak existence účinku ošetření s větší jistotou. Opakováním se v experimentální psychologii rozumí:

a) opakování v různém čase, tzn. vícenásobné uskutečnění experimentu s touž pokusnou osobou,

b) opakování na různých subjektech, tzn. uskutečnění stejného experimentu na více pokusných osobách.

6.3 Příprava, realizace a vyhodnocení psychoakustického experimentu

Strukturu psychoakustického experimentu můžeme rozdělit do následujících etap [1]:

a) kritické studium zdrojů informací,

b) vymezení problému a formulování hypotéz,

c) výběr podnětů,

d) výběr pokusných osob,

e) návrh experimentální procedury,

f) sestavení plánu experimentu,

g) sběr dat,

h) statistická analýza dat,

i) interpretace výsledků a napsání zprávy

6.4 Metoda párového srovnávání zvukových podnětů

Párové srovnávání patří k nejstarším, a zároveň také z teoreticky i technicky nejpropracovanějším psychologickým měřícím metodám. Výhodou této metody je její široká použitelnost pro škálování nejrůznějších psychologických proměnných. Jejími hlavními nevýhodami je jednak časová náročnost a jednak její velké požadavky na trpělivost a soustředěnost pokusných osob. Tato metoda bude následně použita v praktické části experimentu, jenž by měla být náplní diplomové práce navazující na semestrální projekt.

6.4.1 Princip metody

Princip této metody je založen na párovém srovnávání každého z celkem n posuzovaných objektů se všemi zbývajícími. U každého dílčího srovnání vždy pokusná osoba označí jeden z právě posuzovaných podnětů S_j a S_k za dominantní v předem definovaném smyslu (např. hlasitější, vyšší, temnější, plnější, ostřejší atd.). Tuto dominanci lze matematicky vyjádřit nerovností $S_k > S_j$, v případě, že podnět k je dominantnější, než podnět j, popř. $S_j > S_k$, v případě, že podnět j je dominantnější, než podnět k. Pokud ale pokusná osoba není schopna rozlišit dominanci posuzovaného podnětu (tzv. neutrální soud), používá se technika nucené volby, což znamená, že pokusné osobě nezbývá nic jiného než hádat. To ale může, zejména při častějším výskytu těchto situací, vést k deprimaci pokusné osoby a tím pádem k větším chybám. Proto se někdy zavádí třetí forma odpovědi – neutrální, jejíž hlavní nevýhoda spočívá v tom, jak s touto hodnotou naložit při zpracování. Nejčastěji se používají dva způsoby zpracování neutrálních soudů. Buď se připíše půl bodu ve prospěch každého z podnětů, nebo se počet neutrálních soudů rozdělí mezi podněty S_j a S_k v poměru četnosti jednotlivých odpovědí.

Metodu párového srovnávání lze aplikovat třemi různými způsoby:

a) jedna osoba (popř. několik málo pokusných osob) mnohokrát opakovaně posuzuje všechny páry podnětů. Tento způsob se používá v případě, chceme-li důkladně změřit jednoho jedince (či malou skupinu jedinců),

b) mnoho pokusných osob posuzuje všechny páry podnětů jen jednou. Tento způsob použijeme, zajímá-li nás průměrná škála nějaké populace,

c) několik pokusných osob několikrát opakovaně posuzuje všechny páry podnětů. Tento způsob je v psychoakustice pro řešení praktických problémů nejčastější.

6.4.2 Chyby metody

Nejčastěji uváděnými chybami zkreslujícími výsledky párového srovnávání jsou:

a) *únava pokusných osob* způsobuje snížení pozornosti a ztrátu zájmu o experiment. Lze jí předejít vhodnou délkou experimentu, popř. rozdělení experimentu na více časových úseků proložených krátkými přestávkami, určenými k odpočinku.

b) *stupeň zacvičení pokusných osob* je nezanedbatelným zdrojem nepřesností v průběhu experimentu a proto je nutné na začátku experimentu u každé osoby provést důkladný zácvik.

c) *Fechnerova chyby místa* se u akustických experimentů párového srovnávání projevuje především tak, že jeden podnět v páru je preferován z důvodu své časové polohy (první podnět zazní vždy až po druhém). Tento negativní vliv lze potlačit např. tak, že zajistíme, aby se každý podnět vyskytoval stejně často na prvním i na druhém místě v páru.

d) *Fechnerova chyba času* je způsobena tím, že určitý podnět na pokusnou osobu nestačil odeznít a je jí již předkládán další pár k posouzení. Tento negativní vliv eliminujeme dostatečnými časovými prodlevami mezi jednotlivými páry.

Vlivy těchto čtyř hlavních chyb omezujeme vhodným rozvržením vzájemných časových a prostorových poloh podnětů v párech, dále vhodným uspořádáním časového pořadí, ve kterém jsou tyto páry předkládány pokusným osobám. Nejčastěji se používá náhodného uspořádání polohy podnětů v párech a zároveň náhodného uspořádání časového sledu párů podnětů. Tento postup je použit i v experimentu této diplomové práce.

7 Princip změny polohy zdroje zvuku ve sluchátkách

V semestrálním projektu, který předcházel této diplomové práci, bylo úkolem měnit polohu jednokanálového (monofonního) zdroje zvuku horizontální rovině. Toho bylo dosaženo principem interaurální intenzitní diference, tzn. postupnou změnou poměru hlasitostí v jednotlivých sluchátkách tak, aby relativní hlasitost byla vždy rovna 1. Pokud byl zvuk nastaven přesně uprostřed, byla relativní hlasitost v obou sluchátkách rovna hodnotě 0,5 a pozice zvuku tak byla vnímána uprostřed hlavy. Vychýlení polohy se pak docílilo změnou poměru úrovní hlasitostí v levém a pravém sluchátku.

Nyní však je potřeba nastavovat i polohu v mediální rovině, čehož již takto docílit nelze, neboť výše uvedený princip nezohledňuje interaurální časovou diferenci ani změnu kmitočtového spektra v závislosti na příchozím směru.

7.1 Impulsní charakteristika a konvoluce

Aby bylo možné vysvětlit princip změny polohy zdroje zvuku ve sluchátkách, je třeba nejprve připomenout některá fakta z oblasti teorie signálů a soustav. Stěžejní záležitostí je impulsní charakteristika a operace konvoluce.

7.1.1 Impulsní charakteristika

Impulsní charakteristika $h[n]$ je odezva diskrétního lineárního, časově invariantního systému na jednotkový impuls $\delta[n]$ [8]. V tomto případě má impulsní charakteristika formu zvukového souboru typu wav o dvou kanálech a délce 8192 vzorků, kdy každý z kanálů je pořízen mikrofonem v uchu umělé lidské hlavy. Přehráním této impulsní charakteristiky je ve sluchátkách slyšet lupnutí přicházející z daného směru.

7.1.2 Konvoluce

Konvoluce diskrétního signálu $s[n]$ a impulsní charakteristiky $h[n]$ je dána vztahem [9]:

$$w[n] = s[n] * h[n] = \sum_{m=0}^{n} s[n-m]h[m]. \qquad (6.1)$$

Provedením konvoluce původního jednokanálového zdroje zvuku s levým kanálem impulsní charakteristiky se získá zvuk pro levé sluchátko. Analogicky, provedením konvoluce s pravým kanálem impulsní charakteristiky se získá zvuk pro pravé sluchátko. Tím vznikne tentýž, ale tentokrát dvoukanálový (stereofonní) zvuk s tím, že **zdroj zvuku změnil svou původní pozici uprostřed hlavy za pozici, odkud byla změřena daná impulsní**

charakteristika pomocí umělé hlavy. Proto je pro nastavování pozice zvuku potřeba celá sada impulsních odezev nameřená ze všech směrů.

Pro tento experiment byly použity naměřené impulsové charakteristiky [10] s krokem azimutu 15° po celém rozsahu, tedy 360°, kde azimut 0° je pozice přímo před obličejem a roustoucí azimut je proti směru hodinových ručiček. Elevace je taktéž naměřena s krokem 15 °, ale pouze od 45 ° pod horizontem (t.j. elevace 315°) do 45 ° nad horizontem (t.j. elevace 45°). Vzdálenost reproduktoru od umělé hlavy je 195 cm.

8 Aplikace v prostředí Java

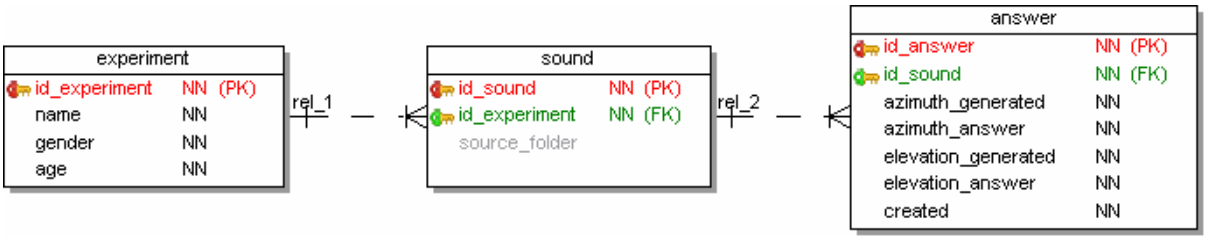
Jak už bylo zmíněno v předchozí části, psychoakustický experiment, který byl v této diplomové práci proveden, je založen na párovém srovnávání dvou zvukových podnětů. Předmětem tohoto zkoumání bylo zjistit, jakou přesností disponuje lidský sluchový systém při určování směru příchozího zvuku v závislosti na jeho kmitočtovém složení, a také na věkové kategorii pokusné osoby.

Bylo tedy nutné naprogramovat aplikaci, která v prvé řadě umožní zaregistrovat nového experimentátora, popř. vybrat jeho již vytvořený profil z možného seznamu pokusných osob, které se experimentu zúčastnily. Dále tato aplikace měla umět vygenerovat na výstupu zvukové karty různé druhy zvuků s různým kmitočtovým složením, které bude pokusná osoba posuzovat. K těmto účelům byl zvolen harmonický tón, úzkopásmový šum s předem nastavitelnými parametry, a dále možnost vybrat zvukový soubor ve formátu wav. Samotný experiment měl potom probíhat tak, že aplikace vygeneruje náhodnou pozici zdroje zvuku a na stisk tlačítka přehraje zvolený zvuk pokusné osobě do sluchátek tak, jako by přicházel z tohoto směru (princip této vlastnosti bude popsán dále viz. kap. 8.2.5). Pokusná osoba pak má v okně aplikace k dispozici jednak ovládací prvky, kterými postupně nastavuje směr, ze kterého daný zvuk slyší, a jednak druhé tlačítko, které na stisknutí přehraje tentýž zvuk, ale ze směru, který je aktuálně nastven těmito ovládacími prvky. Úkolem experimentátora je nastavit ovládací prvky aplikace tak, aby přehráním obou zvuků přicházel tento tvuk ze stejného směru. Stiskem tlačítka “Další” se uloží jednak vygenerovaná a jednak zodpovězená pozice, vygeneruje se nová náhodná pozice a celá tato procedura se opakuje až do té doby, dokud má pokusná osoba o experiment zájem a neprojevuje znatelné známky únavy, či nesoustředěnosti, jelikož tyto faktory výrazně ovlivňují přesnost měření a důvěryhodnost výsledků. Po nasbírání určitého počtu hodnot se provede jejich analýza a vyvodí závěry.

8.1 Databáze a její struktura

Jedním ze základních kamenů této aplikace je databáze. V ní se ukládají všechny hodnoty, které experimentátor zadá od svého jména, popř. jména experimentu po seznam naměřených hodnot. Jazyk SQL pak umožňuje velice snadno a efektivně z této databáze vybírat data, která jsou při zobrazování grafů zapotřebí. Za tímto účelem jsem zvolil databázi MySQL, protože se v případě nekomerčních projektů jedná o freeware a také z důvodu její rozšířenosti, a tím pádem i podpory. Na obr. 8.1 je znázorněna databázová struktura, pořízená programem Toad Data Modeller, což je nástroj pro modelování databází. Vytvoření tabulek,

jejich klíčů a atributů popř. vazeb mezi tabulkami je sice možné i napsáním SQL skriptu, ale tento způsob je poněkud nepraktický. Proto se v praxi používá programů pro modelování databází, ve kterých je na první pohled vidět struktura dat a vztahy mezi jednotlivými entitami.



Obr. 8.1: Databázová struktura.

Z takto vytvořeného databázového modelu se pomocí Toad Data Modelleru vygeneruje SQL skript, který se poté spustí nad databází MySQL. Tím se vytvoří databáze *diploma_thesis* a v ní tři následující tabulky spolu s relacemi.

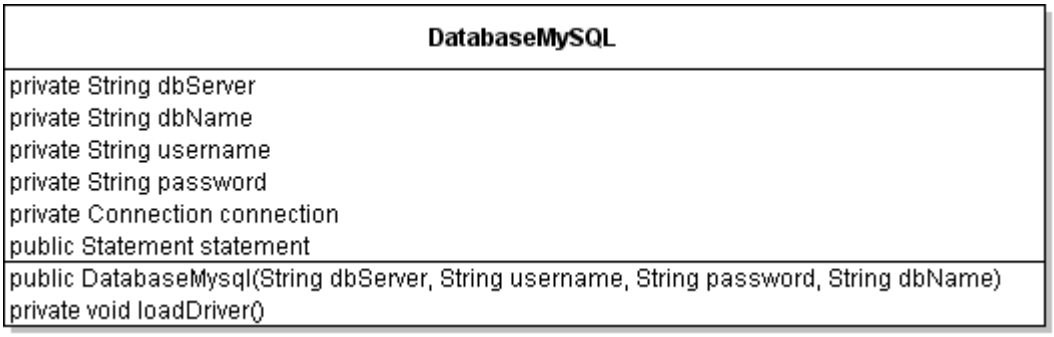
Tabulka *experiment* obsahuje základní informace o pokusné osobě a experimentu, čili název experimentu, popř. jméno osoby, dále pohlaví a věk. Tabulka *sound* představuje seznam všech zvuků, na kterých již uživatel měřil a zprostředkovává tak vazbu typu many-to-many mezi uživatelem a všemi jeho naměřenými hodnotami. Jednotlivé zvuky jsou reprezentovány názvem adresáře (důvod vysvětlen dále, viz. kap. 8.2.5). V tabulce *answer* jsou již pak obsaženy všechny naměřené hodnoty, čili správný směr (tzn. směr, který vygenerovala aplikace) a zodpovězený směr – oba reprezentované azimutem a elevací ve stupních.

8.2 Třídy a jejich hlavní metody

Po zvážení vstupních požadavků a důkladné analýze byla funkčnost celé aplikace rozdělena na níže uvedené funkcionality, implementované pomocí tříd. Objekty těchto tříd mají na starost vždy jednu konkrétní oblast problematiky a jejich spojením je tvořena aplikace jako komplexní celek. Jednou z hlavních výhod objektového programování je přehlednost, intuitivní a rychlá představa o jednotlivých funkcionalitách a tím pádem i snadná rozšiřitelnost v případě dodatečných požadavků.

8.2.1 Třída DatabaseMySQL

Objekt třídy DatabaseMySQL má na starosti navázání spojení s databází, ukládání a čtení informací z ní. Veškerý informační tok mezi aplikací a databází je vždy zprostředkován instancí této třídy. Na obr. 8.2 je znázorněn UML diagram této třídy.

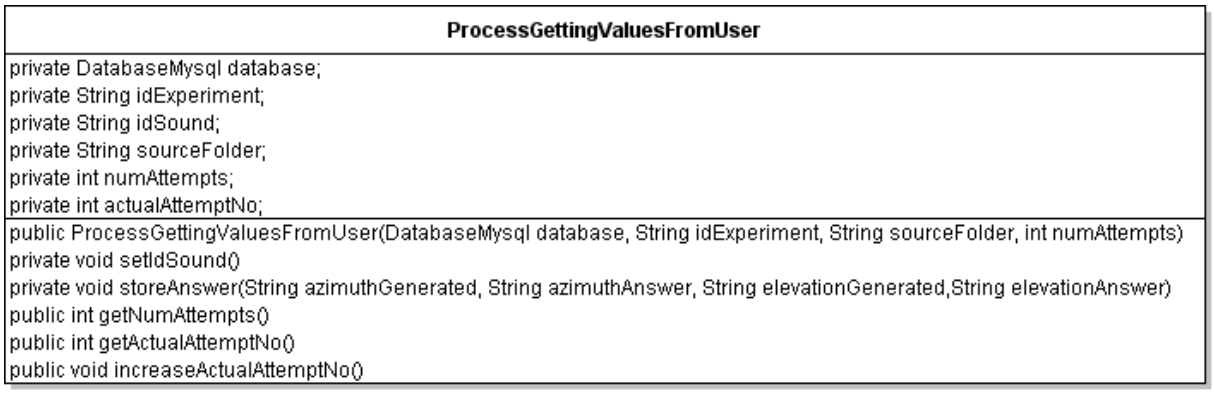


Obr. 8.2 UML diagram třídy MySQL.

Konstruktor obsahuje parametry potřebné pro připojení k databázi, tedy URL adresu MySQL serveru (*dbServer*), uživatelské jméno (*username*), heslo (*password*) a jméno databáze (*dbName*). Po uložení všech předaných parametrů do příslušných privátních atributů se zavolá metoda `loadDriver()`, která iniciuje ovladač pro připojení k databázi, který bylo nutné nejprve stáhnout jako soubor s příponou .jar a přidat ho do projektu. Nyní je už možné se pomocí třídy *DriverManager* připojit k databázi a iniciovat privátní atribut *statement*, pomocí kterého se již zadávají jednotlivé SQL dotazy.

8.2.2 Třída ProcessGettingValuesFromUser

Úkolem této třídy je získávat údaje z grafického rozhraní a pomocí výše uvedeného objektu `DatabaseMySQL` je ukládat.



Obr. 8.3 UML diagram třídy ProcessGettingValuesFromUser.

Konstruktor přebírá objekt *database*, což je instance třídy `DatabaseMySQL`, dále primární klíč *idExperiment*, který identifikuje daný experiment (t.j. pokusnou osobu). Třetím parametrem je název adresáře (*sourceFolder*), ze kterého byl čten zdojový zvukový soubor, a nakonec počet hodnot, které bude uživatel měřit (*numAttempts*).

Metody `getNumAttempts()` a `getActualAttemptNo()` jsou pouze tzv. gettery, což je typ metody, která se používá k získání hodnoty privátního atributu. Metoda

`getNumAttempts()` vrací počet měřených hodnot, `getActualAttemptNo()` vrací číslo aktuální měřené hodnoty.

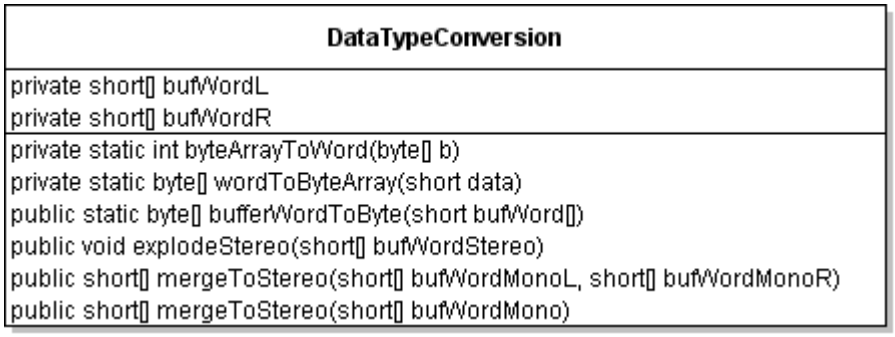
Metoda `increaseActualAttemptNo()` patří mezi tzv. settery, cože typ metody, která naopak umožňuje měnit hodnotu privátního atributu, v tomto případě zvyšuje číslo měřené hodnoty o jedničku.

Metoda `setIdSound()` slouží k výběru, popř. i vytvoření mezivazebního záznamu mezi tabulkami *experiment* a *answer*. Než totiž započne celý experiment, musí uživatel zvolit svůj profil, čímž sdělí aplikaci primární klíč *id_experiment*. Dále musí zvolit zvuk, který je dán názvem adresáře, ze kterého bude aplikace číst zvukové soubory v závislosti na příchozím směru. Tím určí klíč *id_sound*, který je nutný znát pro uložení odpovědi, protože jsou vázané na zvuk a ten pak na experiment. Pokud uživatel na zvoleném zvuku ještě neměřil, je nutné nejprve tento mezivazební záznam vytvořit. Poté co byl vytvořen, popř. nalezen nastaví se privátní atribut *idSound* na tuto hodnotu.

Ve chvíli, kdy známe klíč *id_sound* lze zavolat metodu `storeAnswer()` která přečte uloží správný směr příchozího zvuku a směr, který uživatel zodpověděl. Doplňkovou informací je záznam o datu a čase pořízení jednotlivých hodnot, která může sloužit kupříkladu pro určení nejdelší, nejkratší, popř. průměrné doby, která byla potřeba pro určení jedné hodnoty.

8.2.3 Třída DataTypeConversion

Třída `DataTypeConversion` je klíčovou třídou aplikace z hlediska práce se zvukovými daty. Jazyk Java čte i zapisuje zvuková data pomocí hodnot typu `byte`, tedy osmibitové hodnoty. Zvukové soubory typu wav jsou kódované PCM modulací a k vyjádření jednoho vzorku je často potřeba více bitů, než osm. Navíc soubor typu wav může obsahovat více kanálů, nežli jeden. Z toho plyne, že se při načítání hodnot vzorků ze souboru typu wav musí přečíst nejen samotná data, ale i informace o počtu kanálů a počtu bitů na vzorek (celistvý násobek osmi), které se nacházejí v hlavičce souboru. V závislosti na těchto hodnotách lze pak poskládat skutečné hodnoty vzorků, provést s nimi potřebné operace a před zápisem na zvukovou kartu je opět rozdělit na osmibitové hodnoty. Na obr. 8.4 je znázorněn UML diagram třídy `DataTypeConversion`.



Obr. 8.4 UML diagram třídy DataTypeConversion.

Jak již bylo uvedeno, jazyk Java čte zvuková data po bytech, tedy po osmi bitech bez ohledu na bitovou hloubku, počet kanálů či vzorkovací kmitočet zdrojového zvuku. Protože tato aplikace potřebuje ke své schopnosti měnit směr příchozího zvuku nejen vstupní zvukový soubor o jednom kanále, ale i druhý zvukový soubor s impulsní charakteristikou o dvou kanálech, vychází se z předpokladu, že oba zvukové soubory budou vzorkovány se stejným vzorkovacím kmitočtem 44.1 kHz a stejnou bitovou hloubkou 16 bitů na vzorek. Tyto parametry jsou na vstupu ošetřeny při vytváření nové zvukové sady (viz. kap. 8.3.3).

Stěžejními metodami jsou `byteArrayToWord()` a `wordToByteArray()`. Jedná se o dopředný / zpětný převod dvouprvkového pole typu `byte` na šestnáctibitové číslo typu `short`, které se též nazývá slovo, neboli `word`. Metoda `byteArrayToWord()` tedy jako parametr přebírá dvouprvkové pole typu `byte`. První prvek ponechá beze změny, kdežto u druhého provede operaci bitového posunu na pozice významějších bitů. Hodnota typu `short` na výstupu je potom tvořena operací bitového součtu. Zdrojový kód vypadá následovně.

```
private static int byteArrayToWord(byte[] b) {
      return (int)((b[1] << 8) | (b[0] & 0xff));
}
```

Inverzní metoda `wordToByteArray()` pracuje analogicky tak, že šestnáctibitovou hodnotu rozdělí na dvě osmibitové hodnoty s tím, že druhý prvek musí bitově posunout o osm pozic, tentokrát směrem k méně významným bitům. Takto získané hodnoty uloží do dvouprvkového pole typu `byte` a vrátí jej na výstupu.

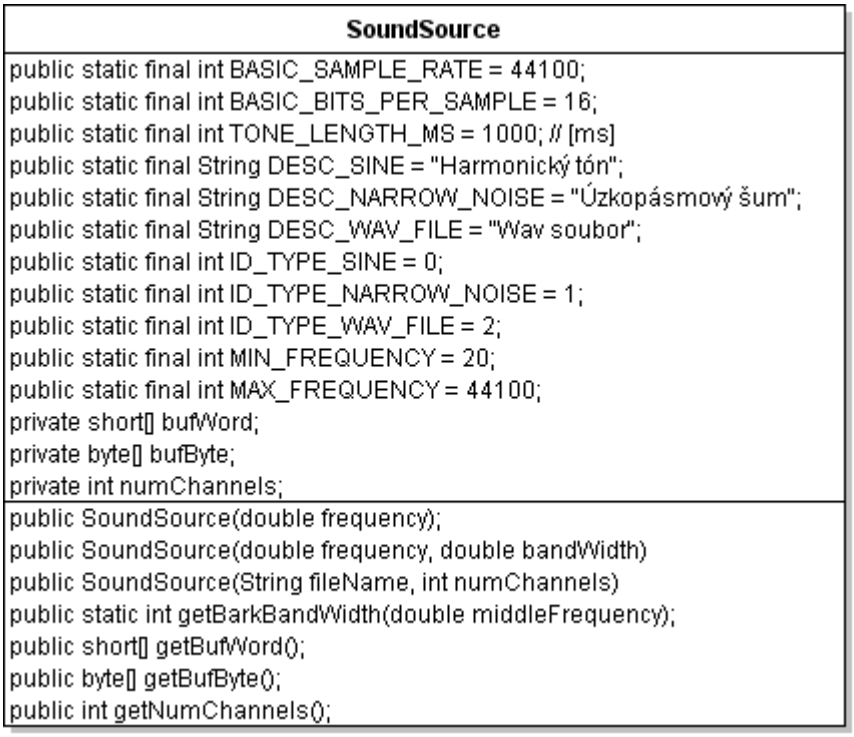
```
private static byte[] wordToByteArray(short data) {
      return new byte[] {(byte)(data), (byte)(data >> 8)};
}
```

Protože celá aplikace pracuje s tzv. vyrovnávacími paměťmi (buffer), což jsou pole předem definovaného typu (v případě této práce se jedná o `byte` a `word`) byla třída `DataTypeConversion` doplněna ještě další dvě inverzní rutiny, a to `bufferByteToWord`() a `bufferWordToByte`(). Obě tyto metody jsou založeny na volání výše uvedených metod v cyklu a jejich úkolem je převod celých vyrovnávacích pamětí z typu `byte` na `word` a zpět.

Poslední oblastí, kterou třída `DataTypeConversion` řeší, jsou převody dvoukanálových vyrovnávacích pamětí na jednokanálové vyrovnávací paměti a zpět. Metoda `mergeToStereo()` má za úkol ze dvou jednokanálových vyrovnávacách pamětí, které přebírá jako parametry, učinit jednu dvoukanálovou tak, že v cyklu za sebe sériově skládá jednotlivé prvky střídavě z jedné a z druhé jednokanálové vyrovnávací paměti. Tato metoda byla přetížena pro možnost zavolání s pouze jedním parametrem jednokanálové vyrovnávací paměti, takže výsledná dvoukanálová vyrovnávací paměť je tvořena rozkopírováním jednokanálových vyrovnávacích pamětí do obou kanálů. Tato metoda má, stejně jako ostatní metody této třídy, svou inverzní metodu, která z dvoukanálové vyrovnávací paměti učiní dvě jednokanálové. Jedná se o metodu `explodeStereo()`. Protože však každá metoda může vracet pouze jednu proměnnou a zde byly potřeba vrátit dvě (jednokanálové vyrovnávací paměti pro levý a pravý kanál), bylo toto vyřešeno uložením obou vyrovnávacích pamětí do privátních atributů *bufWordL* a *bufWordR*, které lze získat přes gettery `getBufWordL()` a `getBufWordR()`.

8.2.4 Třída SoundSource

Další funkcionalita, pro kterou byla vytvořena další třída, je zdroj zvuku. Objekt třídy `SoundSource` při inicializaci naplní svou vyrovnávací paměť zvukovými daty, které mohou být trojího typu – harmonický tón, úzkopásmový šum a soubor typu wav. Na obr. 8.5 je znázorněn UML diagram třídy `SoundSource`.



Obr. 8.5 UML diagram třídy SoundSource.

Konstruktor pro harmonický tón má pouze jeden parametr *frequency*, který udává kmitočet harmonického tónu. Jako první konstruktor nastaví privátní atribut *numChannels* udávající počet kanálů zdroje zvuku, na hodnotu 1, protože zdroj harmonického tónu v této práci není nikde zapotřebí ve stereu. Dále nastaví délku vyrovnávací paměti, která je závislá na implicitně nastavené délce zvuku (1000 ms) a implicitně nastaveném základním vzorkovacím kmitočtu (44,1 kHz). Jednotlivé vzorky harmonického tónu se počítají v cyklu dle níže uvedeného kódu, kde proměná *lenBuf* je počet vzorků ve vyrovnávací paměti.

```
for (int i = 0; i < lenBuf; i++) {
      this.bufWord[i] =
            (short) (Math.sin(i * ((double)frequency /
            (double) SoundSource.BASIC_SAMPLE_RATE) * Math.PI) *
            Sound.MAX_SHORT);
}
```

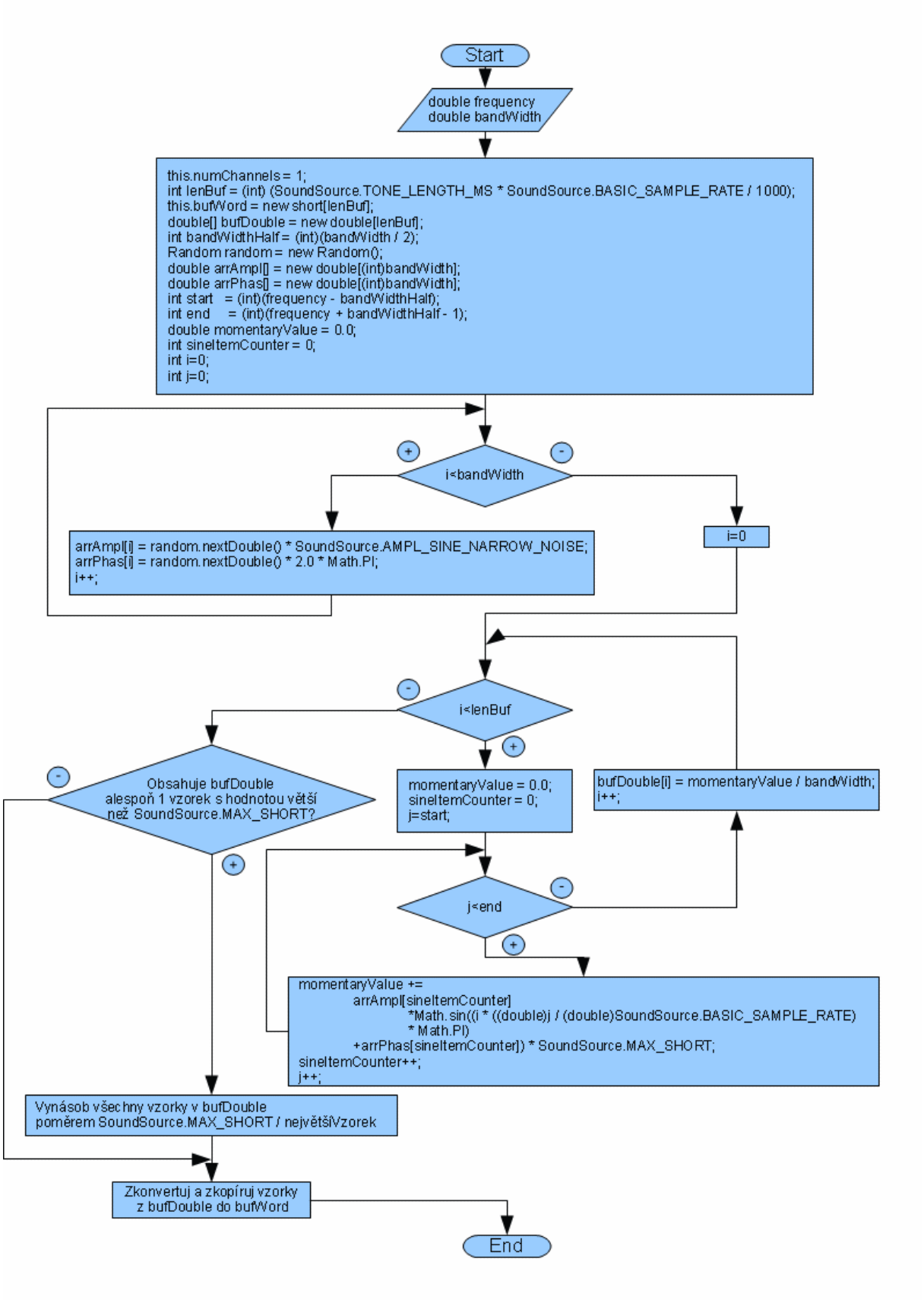
Každý vzorek je spočten jako hodnota funkce sin(*x*), kde jeho argumentem *x* je součin řídící proměnné cyklu *i* s číslem π a s podílem zadaného kmitočtu ku vzorkovacímu kmitočtu. Takto vytvořený signál má ale amplitudu v intervalu <-1; 1>, což je při rozsahu datového typu `short` nepoměrně málo a takto přehraný signál by byl sotva slyšitelný. Proto se musí každá takto vypočtená hodnota ještě vynásobit maximální dosažitelnou hodnotou typu `short`, která je uložená v konstantě *SoundSource.MAX_SHORT* = 32767, aby signál dosáhl maximální možné amplitudy bez nelineárního zkreslení.

Konstruktor pro úzkopásmový šum má již parametry dva. Prvním je střední kmitočet a druhým šířka pásma. Pokud se šířka pásma nastaví na hodnotu 0, bude v programu nahrazena hodnotou šířky pásma z barkové škály kritických pásem pomocí metody `getBarkBandWidth()` (viz. tab. 8.1).

Tab. 8.1 Barková škála kritických pásem [11]

[Bark]	$f_{dolní\text{-}mezní}$[Hz]	$f_{horní\text{-}mezní}$ [Hz]	$f_{střední}$ [Hz]	Šířka pásma[Hz]
1	0	100	50	100
2	100	200	150	100
3	200	300	250	100
4	300	400	350	100
5	400	510	450	110
6	510	630	570	120
7	630	770	700	140
8	770	920	840	150
9	920	1080	1000	160
10	1080	1270	1170	190
11	1270	1480	1370	210
12	1480	1720	1600	240
13	1720	2000	1850	280
14	2000	2320	2150	320
15	2320	2700	2500	380
16	2700	3150	2900	450
17	3150	3700	3400	550
18	3700	4400	4000	700
19	4400	5300	4800	900
20	5300	6400	5800	1100
21	6400	7700	7000	1300
22	7700	9500	8500	1800
23	9500	12000	10500	2500
24	12000	15500	13500	3500

Počet kanálů je opět nastaven na hodnotu 1 ze stejného důvodu jako u konstruktoru pro harmonický tón – zdroj úzkopásmového šumu ve dvou kanálech v této práci není potřeba. Pro vytvoření vyrovnávací paměti s úzkopásmovým šumem byla zvolena aditivní metoda. Princip spočívá v tom, že výsledný signál je tvořen součtem harmonických signálů, které se vyskytují na daném kmitočtovém pásmu s odstupem 1 Hz a které mají náhodnou amplitudu a náhodnou fázi. Na obr. 8.6 je znázorněn vývojový diagram konstruktoru pro úzkopásmový šum.



Obr. 8.6 Vývojový diagram konstruktoru pro úzkopásmový šum.

Tab. 8.2 Proměnné v konstruktoru pro úzkopásmový šum

Datový typ	**Název**	**Význam**
double	frequency	střední kmitočet
double	bandWidth	šířka pásma
double[]	bufDouble	vyrovnávací paměť typu `double`
short[]	bufWord	vyrovnávací paměť typu `short`
int	bandWidthHalf	polovina šířky pásma
Random	random	objekt pro generování náhodných čísel
double[]	arrAmpl	pole pro náhodné hodnoty amplitud harmonických složek
double[]	arrPhas	pole pro náhodné hodnoty fází harmonických složek
int	start	dolní mez spektra
int	end	horní mez spektra
double	momentaryValue	aktuálně počítaná hodnota signálu
int	sineItemCounter	počítadlo harmonických složek v průchodu cyklem

Do konstruktoru vstupují parametry *frequency* a *bandWidth*. V prvním cyklu s řídící proměnnou cyklu *i* se plní pole *arrAmpl* a *arrPhas*, jejichž počet prvků je roven šířce pásma. Tyto pole vyjadřují amplitudy a fázové posuny jednotlivých harmonických složek výsledného úzkopásmového šumu. Poté co jsou tato pole naplněna, začíná výpočet jednotlivých vzorků výstupního signálu, který se ukládá do pole *bufDouble*. Tento výpočet probíhá v dalším cyklu s řídící proměnnou *i*. Okamžitá hodnota výstupního signálu, vyjádřená proměnnou *momentaryValue*, je tvořena podílem součtu všech okamžitých hodnot harmonických tónů (vnořený cyklus s řídící proměnnou *j*) a šířkou pásma *bandWidth* – tedy aritmetickým průměrem okamžitých hodnot amplitud jednotlivých harmonických složek. Po vypočtení okamžité hodnoty se tato uloží jako další prvek pole *bufDouble* a stejným způsobem se pokračuje až do doby, kdy řídící proměnná *i* dosáhne hodnoty *lenBuf*, čili délky vyrovnávací paměti. Nakonec je potřeba zkontrolovat, zda *bufDouble* neobsahuje prvky které jsou větší, než maximální hodnota typu `short`. Pokud se zde takové hodnoty vyskytují, je třeba snížit hodnoty všech vzorků poměrem maximální hodnota typu `short` ku největší hodnotě vzorku v *bufDouble*. Poté je možné převést výslednou vyrovnávací paměť z datového typu `double` na typ `short`.

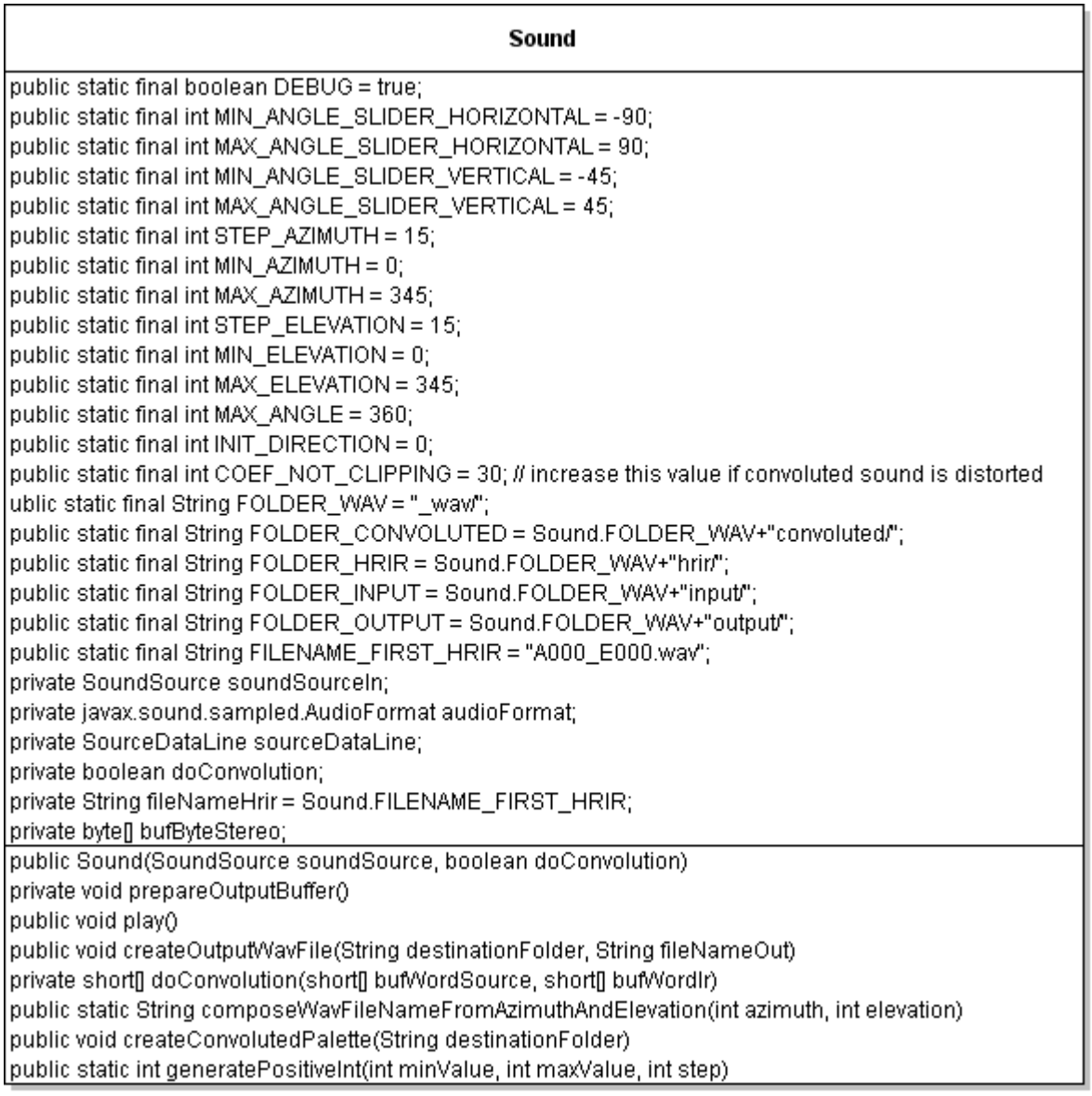
Konstruktor pro zvukový soubor typu wav má dva parametry. Prvním je *fileNamePath*, což je název souboru včetně umístění v souborovém systému, a druhým je počet očekávaných kanálů *numChannels*, protože při vytváření objektu SoundSource je potřeba vědět, kolik bude mít vstupní zvukový soubor kanálů. Pokud počet očekávaných kanálů je roven jedné, očekává se, že tento zdroj zvuku bude vzápětí upravován, a teprve potom přehrán. Pokud je ale počet očekávaných kanálů roven dvěma, bude tento zvuk již pouze přehráván, což znamená, že se přeskočí veškeré procedury pro úpravu zvukových dat,

převedou se rovnou na výstup zvukové karty a ušetří se podstatná část výpočetního času. Význam tohoto parametru bude patrný dále (viz. kap 8.3.1 a 8.3.3).

Zbývající metody jsou gettery – `getBufWord()` vrací zvuková data typu `word`, `getBufByte()` vrací táž zvuková data, ale typu `byte` a `getNumChannels()` vrací počet kanálů.

8.2.5 Třída Sound

Třída Sound má na starosti operace s připraveným zdrojem zvuku, jako je např. jeho přehrání do sluchátek, změna pozice zdroje zvuku, kterou slyší experimentátor ve sluchátkách, či vytváření nových zvukových souborů a zvukových sad. Na obr. 8.7 je znázorněn její UML diagram.

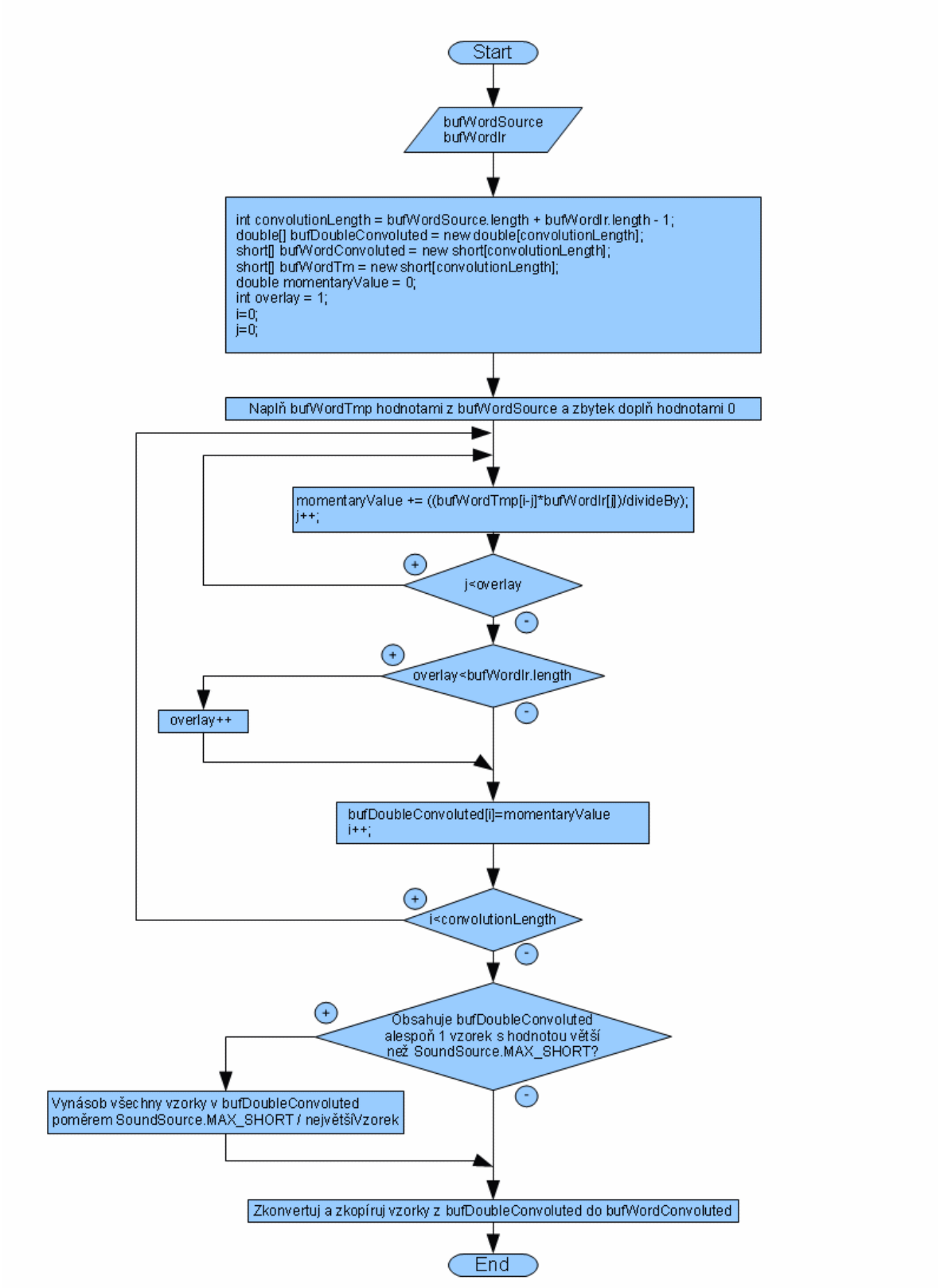


Obr. 8.7 UML diagram třídy Sound.

Konstruktor třídy Sound má dva prametry – zdroj zvuku *soundSource* a příznak *doConvolution*, který určuje, zda se bude pro daný zvuk provádět operace konvoluce či nikoliv. Hodnoty těchto parametrů jsou použity pro nastavení privátních atributů *soundSourceIn* a *doConvolution*. Dále má konstruktor za úkol iniciovat atributy *audioFormat* a *sourceDataLine*. Objekty datového typu `AudioFormat` vypovídají o zvukovém formátu zdroje, jako je např. jeho vzorkovací kmitočet, počet bitů na vzorek, počet kanálů, informace o endianitě (pořadí uložení bajtů v paměti počítače) apod. Objekty typu `SourceDataLine` potom přímo komunikují se zvukovou kartou a umožňují zápis (čtení) zvukových dat na ni (z ní) v závislosti na formátu zvukových dat předaných pomocí objeku typu `AudioFormat`.

Metoda `doConvolution()`, jak již název napovídá, provede konvoluci s předanou vyrovnávací pamětí zvukových dat a vyrovnávací pamětí impulsní odezvy. Na obr. 8.8 je zobrazeno její blokové schéma. V prvním kroku metoda převezme vstupní parametry *bufWordSource* (vyrovnávací paměť typu `word` s vstupím signálem) a *bufWordIr* (vyrovnávací typu `word` s impulsní charakteristikou). V dalším kroku iniciuje potřebné pomocné proměnné. Vzhledem k tomu, že se bude daný signál upravovat sčítáním, násobením a dělením, dojde s největší pravděpodobností k přetečení maximální hodnoty typu `short`. Proto je potřeba převést obě vyrovnávací paměti i pomocné proměnné na datový typ `double`. Proměnná *bufWordTmp* se iniciuje jako pole typu `short` s délkou *convolutionLength* (délka konvoluce), což je součet délky impulsní charakteristiky a délky vstupní vyrovnávací paměti dekrementované o jedničku. Do ní se zkopíruje obsah vstupní vyrovnávací paměti a zbytek se doplní hodnotami 0. Tímto se předejde přetečení vyrovnávací paměti v poslední fázi výpočtu konvoluce, kdy řídící proměnná cyklu *i* dosáhne hodnoty větší, než je déka *bufWordSource*. Ve chvíli, kdy jsou iniciovány všechny pomocné proměnné, zahájí se vnější cyklus s řídící proměnnou *i* a s počtem kroků *convolutionLength*. Každým průchodem tohoto cyklu dojde k výpočtu jednoho vzorku konvolvovaného signálu. K tomu je ale zapotřebí ještě vnitřní cyklus s řídící proměnnou *j*. Tento vnitřní cyklus má proměnný počet průchodů v závislosti na proměnné *overlay*. Ta nabývá hodnot z intervalu <1; *bufWordIr .length*> a vypovídá o počtu sčítanců v aktuálním kroku výpočtu konvoluce. Na začátku algoritmu je proměnná *overlay* nastavena na hodnotu 1 a s každým průchodem vnějším cyklem se inkrementuje vždy o jedničku, ale pouze do doby, kdy dosáhne délky vyrovnávací paměti impulsní charakteristiky. Na závěr opět algoritmus provede kontrolu, zda vyrovnávací paměť s výsledným signálem neobsahuje hodnoty větší, než maximální hodnota

typu `short` a pokud ano, provede omezení úrovně signálu stejně jako u konstruktoru úzkopásmového šumu (viz kap. 8.2.4) a převede na datový typ `short`.

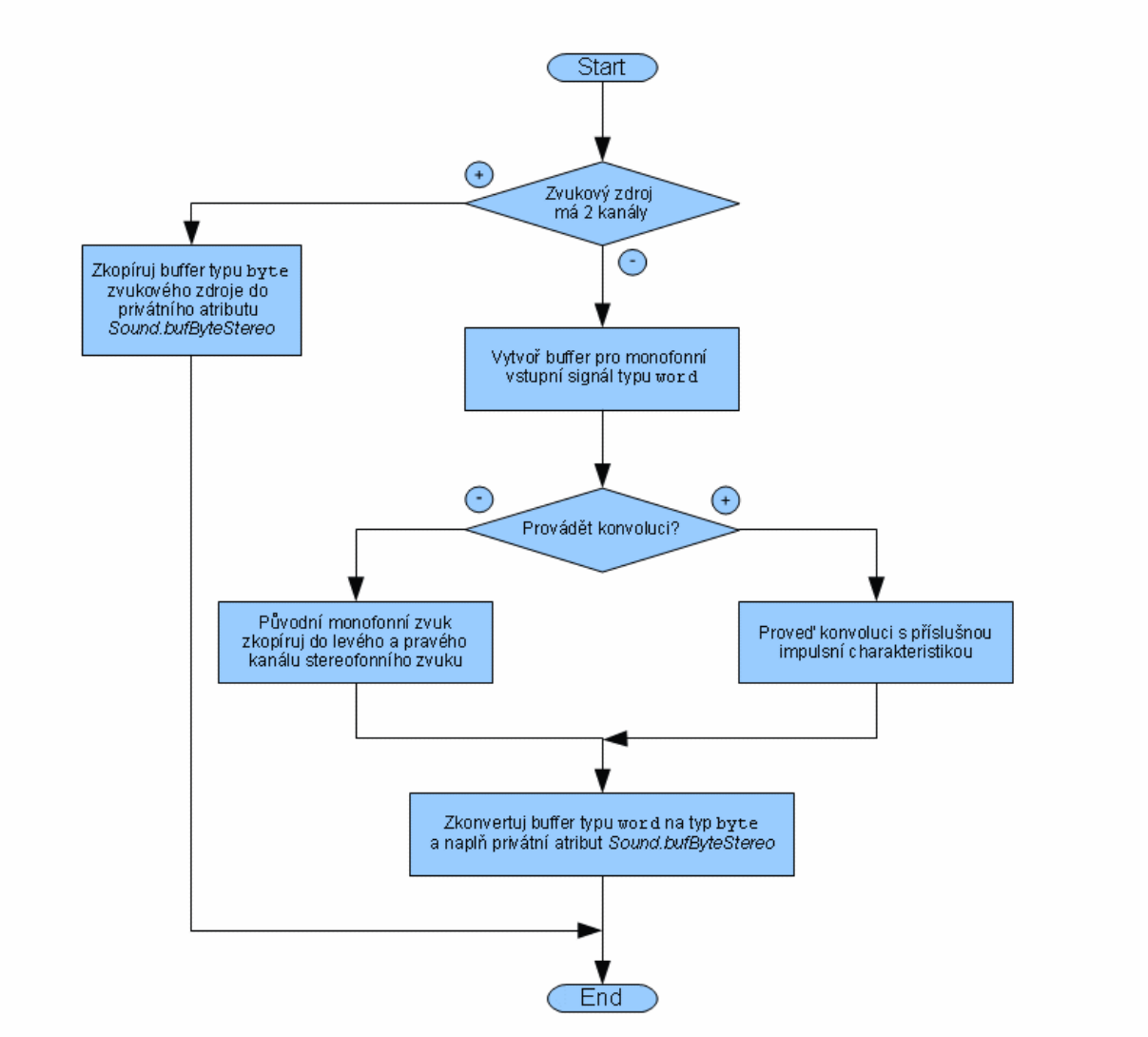


Obr. 8.8 Blokové schéma metody `Sound.doConvolution()`.

Tab. 8.3 Proměnné v metodě `Sound.doConvolution()`

Datový typ	**Název**	**Význam**
int	convolutionLength	délka konvoluce v počtu vzorků
double[]	bufDoubleConvoluted	vyrovnávací paměť pro výsledný konvolvovaný zvuk typu `double`
short[]	bufWordConvoluted	vyrovnávací paměť pro výsledný konvolvovaný zvuk typu `short`
short[]	bufWordTmp	pomocný vyrovnávací paměť pro uložení vstupní vyrovnávací paměti
double	momentaryValue	pomocná proměnná pro výpočet jednoho konvolvovaného vzorku
int	overlay	počet vzorků, které se překrývají
int	i, j	řídící proměnné cyklů

Další metoda, kterou je nutno zavolat před operacemi se zvukovými daty, je `prepareOutputBuffer()`. Jejím hlavním úkolem je naplnit privátní atribut *bufByteStereo*, zvukovými daty v závislosti na parametrech předaných v konstruktoru. Směrodatné jsou informace o počtu kanálů zvukového zdroje a dále příznak, zda se má provádět konvoluce, či nikoliv. Na obr. 8.9 je znázorněno blokové schéma. Jako první krok se provede kontrola, kolik kanálů má vstupní zdroj zvuku. Pokud má zdroj zvuku dva kanály, potom se vychází z předpokladu, že zdrojem zvuku je zvukový dvoukanálový soubor typu wav, jelikož harmonický tón i úzkopásmový šum jsou vytvářeny jako jednokanálové. Za tohoto předpokladu již lze pouze naplnit privátní atribut *bufByteStereo* daty, která se nachází ve vyrovnávací paměti typu `byte` v objektu třídy `SoundSource`, přeskočit všechny ostatní procedury a tím optimalizovat celý algoritmus, jelikož zvuk se dvěma kanály je zde určen pouze k přehrání. Pokud má ale zdroj zvuku jiný počet kanálů než dva (tzn. v tomto případě jeden – jiný počet kanálů je odchycen při vytváření objektu třídy `SoundSource` pro zvukový soubor, kdy se do parametru *numChannels* zadává očekávaný počet kanálů), je jasné, že se budou provádět operace se zvukovými daty, a proto se v následujícím kroku musí vytvořit vyrovnávací paměť typu `word`, nad kterým budou tyto operace probíhat. Dále dojde ke druhému rozhodovacímu kroku, a to jestli se má provést konvoluce. Pokud je privátní atribut *doConvolution* nastaven na hodnotu *false*, jednokanálová zvuková data se pouze rozkopírují do levého a pravého kanálu výsledného dvoukanálového zvuku. Pokud je *doConvolution* nastavena na *true*, provede se konvoluce vstupní jednokanálové vyrovnávací paměti s levým kanálem a poté i s pravým, čímž se vytvoří levý a pravý kanál výstupního dvoukanálového zvuku. Nakonec se provede konverze výstupní vyrovnávací paměti na typ `byte` a zvuková data jsou připravena k použití.



Obr. 8.9 Blokové schéma metody `Sound.prepareOutputBuffer()`.

Nejjednodušším příkladem použití zvukových dat objektu třídy `Sound` je jeho přehrání, což má na starosti metoda `play()`. Ta se v prvé řadě ujistí, zda jsou připravena zvuková data. Pokud je vyrovnávací paměť zvukových dat *bufByteStereo* prázdná, zavolá se metoda `prepareOutputBuffer()`, kterými se naplní. Tento krok je další optimalizací, která zajišťuje přeskočení přípravy zvukových dat, pakliže jsou tato již připravena. Poté dojde k použití objektu třídy `SourceDataLine`, který obsahuje metody k plnění a vyprazdňování vyrovnávací paměti se zvukovými daty. První volanou metodou je metoda `SourceDataLine.open()` která provede otevření linky specifikovaného zvukového formátu, získá potřebné systémové zdroje, a uvede ji do akceschopného stavu. Následně volaná metoda `SourceDataLine.start()` umožňuje lince zapisovat a číst vstupní a výstupní data na zvukové kartě. Metodou `SourceDataLine.write()` se zapíší zvuková data na zvukovou kartu a zvuk je přehrán. Korektním zakončením celé této operace je volání

metody `SourceDataLine.drain()`, která vyprázdní vyrovnávací paměť se zvukovými daty a dále voláním metody `SourceDataLine.close()`, která uzavře linku a uvolní systémové zdroje.

Protože výpočetní operace, zejména konvoluce, trvá řádově několik desítek sekund, bylo potřeba, zpočátku kvůli ladění (debugging) napsat metodu, která výsledný zvuk uloží do souboru typu wav. Takto bylo možné si okamžitě několikrát přehrát výsledný zvuk bez nutnosti pokaždé čekat na výsledek. K tomuto účelu byla do třídy *Sound* přidána metoda `createOutputWavFile()`, která, podobně jako metoda `play()`, nejprve zkontroluje, zda jsou připravena zvuková data, popř. je připraví, a vytvoří soubor typu wav ve specifikovaném adresáři se specifikovaným jménem. K tomu potřebuje znát typ souboru a zvukový formát, který bude stejný, jako formát impulsních charakteristik. Proto si načte první soubor s impulsní charakteristikou, přečte si z hlavičky potřebné informace a na jejich základě pak vytvoří nový zvukový soubor stejného formátu s daty, která jsou připravena v atributu *bufByteStereo*.

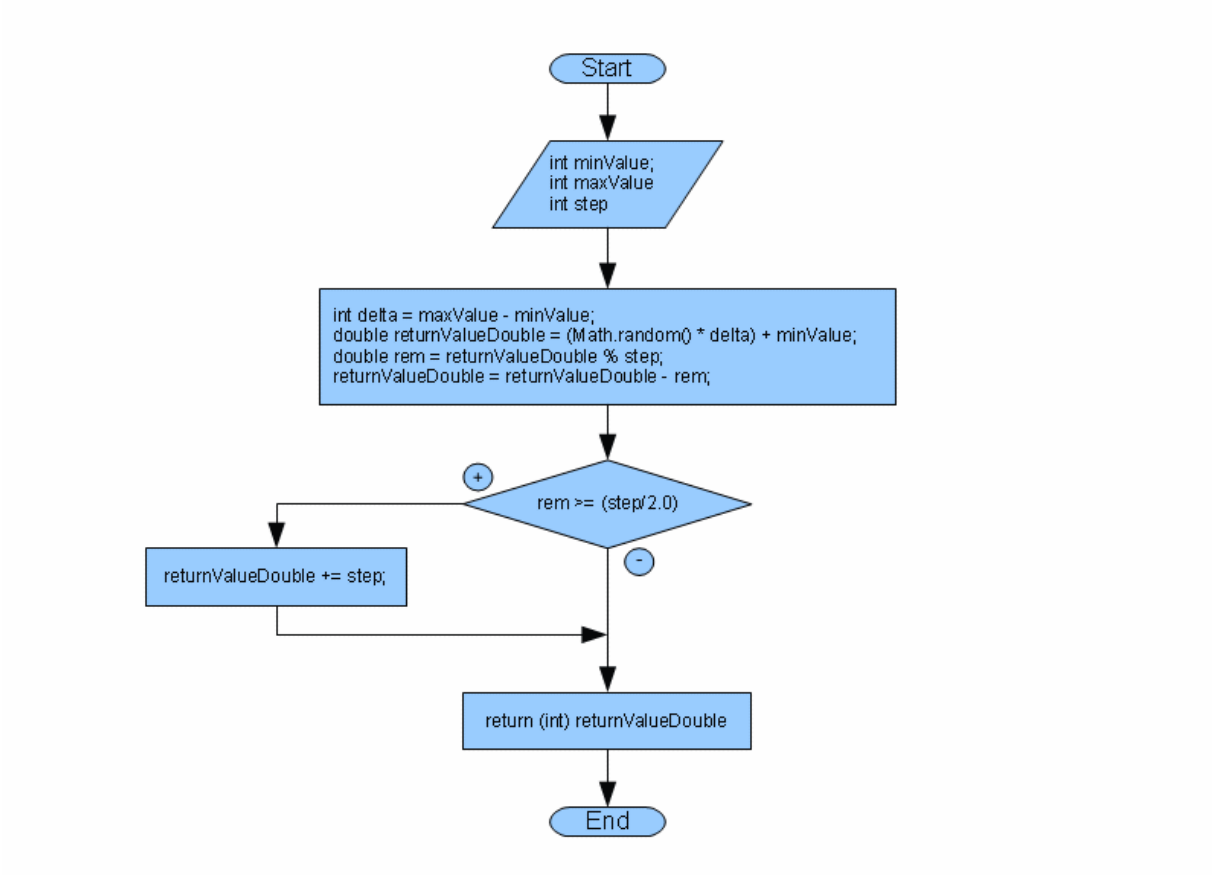
Původní myšlenka změny pozice zdroje zvuku ve sluchátkách byla taková, že se při každé změně parametrů udávajících pozici zdroje a stiskem tlačítka pro přehrání zvolí příslušná impulsní charakteristika, provede se konvoluce a zvuk se přehraje do sluchátek. Jak již bylo naznačeno výše, operace konvoluce vyžaduje příliš velký výpočetní čas na to, aby mohla být prováděna za běhu. Proto byl tento problém vyřešen tak, že pokud bude experimentátor chtít měřit na zvuku, který mu zatím aplikace nenabízí, musí si jej vytvořit. Za tímto účelem je ve třídě `Sound` metoda `createConvolutedPalette()`, která zvolenému zdroji zvuku vytvoří pro každou impulsní charakteristiku konvolvovaný dvoukanálový zvukový soubor typu wav (užitím metody `createOutputWavFile()`). Tato operace může trvat i několik desítek minut v závislosti na výkonu procesoru a zatížení operačního systému. Poté je však k dispozici celá sada zvukových souborů pro každou pozici zdroje zvuku. Změnou parametrů udávájících pozici zdroje zvuku se pak pomocí metody `composeWavFileNameFromAzimuthAndElevation()` zjistí jméno souboru, který má být přehrán (jméno souboru má tvar Aaaa_Eeee.wav, kde *aaa* je azimut ve stupních a *eee* je elevace ve stupních). Vyvolání operace přehrání je pak otázka několika desítek milisekund.

Poslední metodou třídy `Sound` je `generatePositiveInt()`, která se používá k vygenerování nového azimutu a elevace. Protože impulsní charakteristiky jsou měřeny s určitým krokem a na určitém rozsahu (např. měření elevace zcela zespod umělé hlavy přináší jisté komplikace s umístěním reproduktoru), je nutné ošetřit tuto metodu tak, aby

vracela hodnoty na specifikovaném rozsahu se specifikovaným krokem. Parametry této metody tedy jsou *minValue* (nejmenší možná hodnota na výstupu), *maxValue* (největší možná hodnota na výstupu) a *step* (krok mezi hodnotami). Pokud je potom potřeba získat náhodnou elevaci, zavolá se následující příkaz.

```
this.elevationGenerated =
      Sound.generatePositiveInt(
            Sound.MIN_ELEVATION,
            Sound.MAX_ELEVATION,
            Sound.STEP_ELEVATION);
```

Algoritmus této metody je znázorněn v blokovém schématu na obr. 8.10



Obr. 8.10 Blokové schéma metody `Sound.generatePositiveInt()`.

Tab. 8.4 Proměnné v metodě `Sound.generatePositiveInt()`

Datový typ	**Název**	**Význam**
int	minValue	minimální požadovaná hodnota
int	maxValue	maximální požadovaná hodnota
int	step	krok mezi jednotlivými hodnotami
double	delta	velikost intervalu
double	rem	zbytek po celočíselném dělení
double	returnValueDouble	hodnota na výstupu typu double

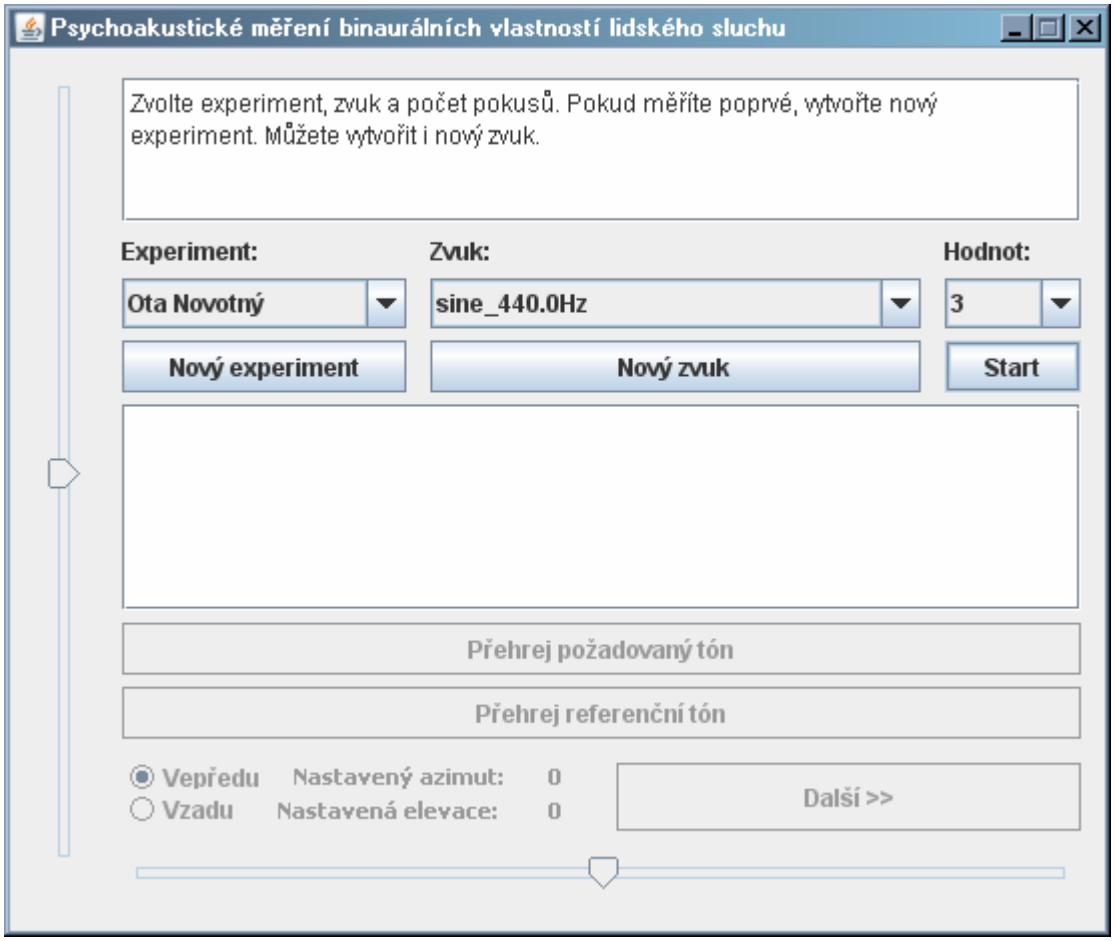
Nejprve se spočte velikost intervalu *delta*, na kterém se má číslo vygenerovat. Poté se pomocí metody `Math.random()` vygeneruje číslo z intervalu <0; 1> a vynásobí se velikostí intervalu, čímž se získá desetinné číslo z intervalu <0; *delta*>. Přičtením dolní meze *minValue* k této hodnotě se již získá desetinné číslo z požadovaného intervalu. Nyní je potřeba zjistit bližší ze dvou hodnot definované krokem (*step*). Od hodnoty *returnValueDouble* se odečte její desetinná část a ta se porovná s poloviční délkou kroku (*step*). Pokud je tato desetinná část větší, nebo rovna polovičnímu kroku, znamená to, že *returnValueDouble* je bližší vyšší hodnotě z rozsahu. Proto se v tomto případě k návratové hodnotě přičte jeden krok. Toto číslo je celé, proto je lze vrátit jako hodnotu datového typu `int`.

8.3 Formuláře

Formuláře jsou třídy, které v jazyce Java umožňují tvorbu grafických objektů, nejčastěji oken, včetně jejich ovládacích prvků. Aplikace sestává ze tří formulářů, kterými jsou hlavní formulář, ve kterém experiment probíhá, a dále dva menší formuláře, jeden pro vytvoření nového experimentu (pokusné osoby) a druhý pro vytvoření nového zvuku (nové zvukové palety).

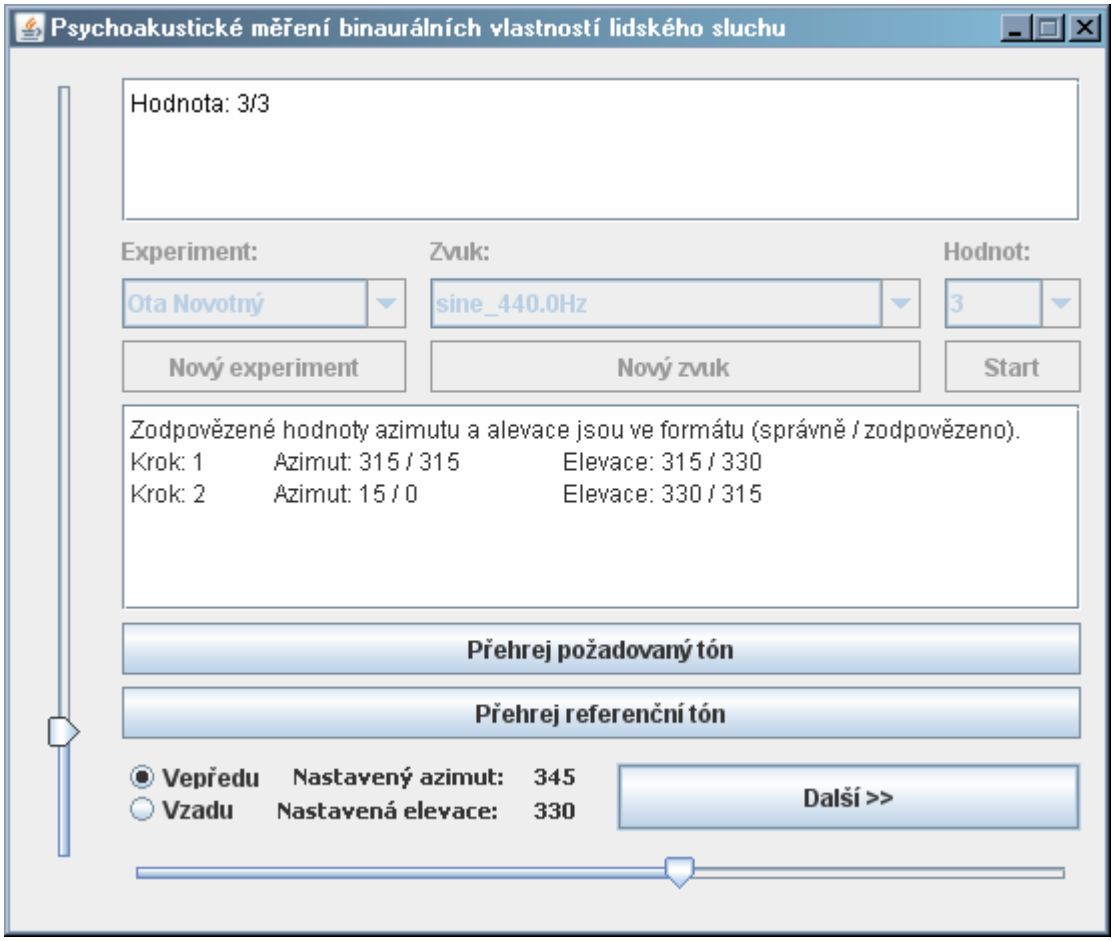
8.3.1 Hlavní formulář pro experiment

Hlavní formulář obsahuje okno se všemi ovládacími prvky, které jsou potřeba pro provedení experimentu. Po spuštění má aplikace aktivní jen některé prvky (viz obr. 8.11). Nejprve se totiž musí pomocí rolovací lišty zvolit experimentátor, který bude měřit, dále zvuk, na kterém bude měřit a nakonec počet hodnot. Vzhledem k tomu, že toto měření vyžaduje značnou míru soustředěnosti a zájmu o experiment, obsahuje rolovací lišta s počet *3*, *5* a *10* hodnot. Měřením více hodnot na jeden set může způsobit ztrátu zájmu o experiment, a proto je vhodné prokládat měření několikaminutovými přestávkami.



Obr. 8.11 Hlavní okno aplikace po spuštění.

Po stisku tlačítka *Start* se zneaktivní prvky pro nastavování parametrů experimentu a naopak se zaktivní ostatní ovládací prvky. Nyní již aplikace náhodně zvolila pozici zvuku a stiskem tlačítka *Přehrej požadovaný tón* se tento přehraje do sluchátek. Úkolem experimentátora je nastavit pozici zdroje ze kterého zvuk slyší. K tomu má k dispozici jednak přepínací tlačítko *Vepředu / Vzadu* a dále horizontální posuvník pro nastavení azimutu a vertikální posuvník pro nastavení elevace. Dle obr. 8.12 jsou nyní ovládací prvky nastaveny tak, jakoby byl zdroj vepředu, skoro dole a mírně vpravo. Přesné hodnoty nastavené pomocí těchto ovládacích prvků jsou zobrazeny ve stupních – viz *Nastavený azimut* a *Nastavená elevace*. Experimentátor má ovšem ještě k dispozici tlačítko *Přehrej referenční tón*, který mu do sluchátek přehraje tentýž zvuk, ale z pozice, kterou má aktuálně nastavenu ovládacími prvky. Takto si může oba zvuky porovnat a dodatečně upravit nastavenou pozici. Pokud si již experimentátor myslí, že nastavil správnou pozici zdroje, stiskne tlačítko *Další*. Tím se uloží jeho odpověď do databáze, aplikace vygeneruje novou náhodnou pozici zdroje a zobrazí zodpovězené a správné hodnoty (viz obr. 8.12).

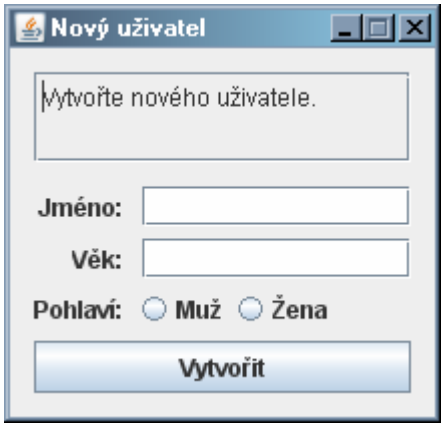


Obr. 8.12 Hlavní okno aplikace v průběhu experimentu.

Naměřením nastaveného počtu hodnot se zneaktivní všechny prvky aplikace až na tlačítko *Start*. Po kliknutí na něj se aplikace uvede do stavu, ve kterém byla po spuštění, uživatel může nastavit jiné parametry experimentu a spustit nové měření.

8.3.2 Formulář pro nového uživatele

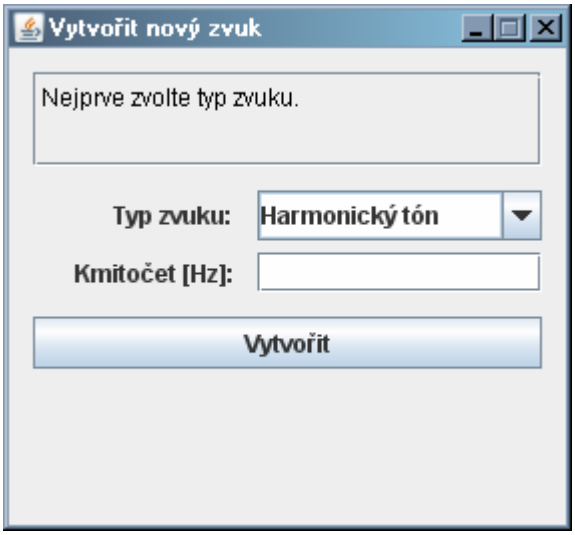
Tento jednoduchý formulář umožňuje přidat další osobu do seznamu experimentátorů (viz. obr. 8.13). Aktivuje se stiskem tlačítka *Nový experiment*. Pro jeho vytvoření je nutno zadat jméno, věk a zvolit pohlaví. Aplikace rovněž kontroluje zadané informace a v případě neplatné hodnoty vypíše červeným tučným písmem chybové hlášení. Po zavření okna se aktualizuje seznam experimentátorů v rozbalovací liště v hlavním okně.



Obr. 8.13 Formulář pro vytvoření nového uživatele.

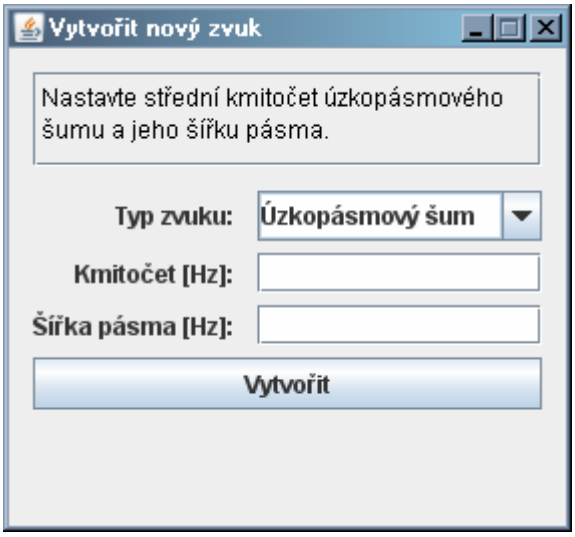
8.3.3 Formulář pro nový zvuk

Poslední formulář umožňuje vytvořit nový zvuk a aktivuje se stiskem tlačítka *Nový zvuk*. Jak již bylo popsáno výše, vytvoření nového zvuku pro měření obnáší vytvoření celé sady zvuků konvolvovaných s každou impulsní charakteristikou. Nejprve se musí zvolit typ zvuku. Po jeho zvolení se objeví příslušné textové pole specifikující vlastnosti vybraného zvuku. Harmonický tón má pouze jeden parametr, a to kmitočet v jednotkách Hz (viz. obr. 8.14).



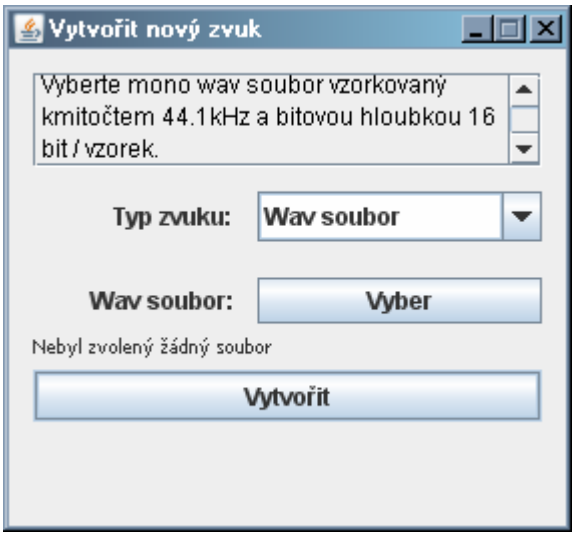
Obr. 8.14 Formulář pro vytvoření nového zvuku (harmonický tón).

Pokud je zvolen typ zvuku úzkopásmový šum, je nutné zadat jeho střední kmitočet (viz. obr. 8.15). Pokud se nezadá šířka pásma, je potom automaticky odvozena z Barkovy škály kritických pásem (viz. tab. 8.1).



Obr. 8.15 Formulář pro vytvoření nového zvuku (úzkopásmový šum).

Posledním možným zdrojem zvuku je zvukový soubor typu wav. Stiskem tlačítka *Vyber* se otevře dialog pro výběr souboru omezený pouze na adresáře a soubory s příponou wav. Tento soubor je ještě omezen podmínkou, že musí být jednokanálový, dále musí být vzorkován kmitočtem 44,1 kHz a bitová hloubka jednotlivých vzorků musí být 16 bitů na vzorek.



Obr. 8.16 Formulář pro vytvoření nového zvuku (zvukový soubor typu wav).

Po stisku tlačítka *Vytvořit* se provede validace vstupních hodnot a dále kontrola, zda výtupní adresář, který bude vzápětí vytvářen pro uložení nových zvukových souborů, existuje či nikoliv. V případě, že některá ze vstupních hodnot není validní, popř. cílový adresář (tzn. zvuk s aktuálně nastavenými parametry) již existuje, vypíše se opět červeným tučným písmem příslušné chybové hlášení. Pokud je vše v pořádku, vytvoří se v adresáři *convoluted*

(v kořenovém adresáři projektu) adresář který ve zkratce charakterizuje daný zvuk. Název adresáře je vytvořen podle následujícího formátu.

Harmonický tón	- sine_fHz,
Úzkopásmový šum	- nn_fHz_bw_BHz
Zvukový soubor	- filename.wav

kde *f* je základní kmitočet harmonického tónu, popř. střední kmitočet úzkopásmového šumu v jednotkách Hz, *B* je šířka pásma úzkopásmového šumu rovněž v jednotkách Hz a filename.wav je název zdrojového zvukového souboru.

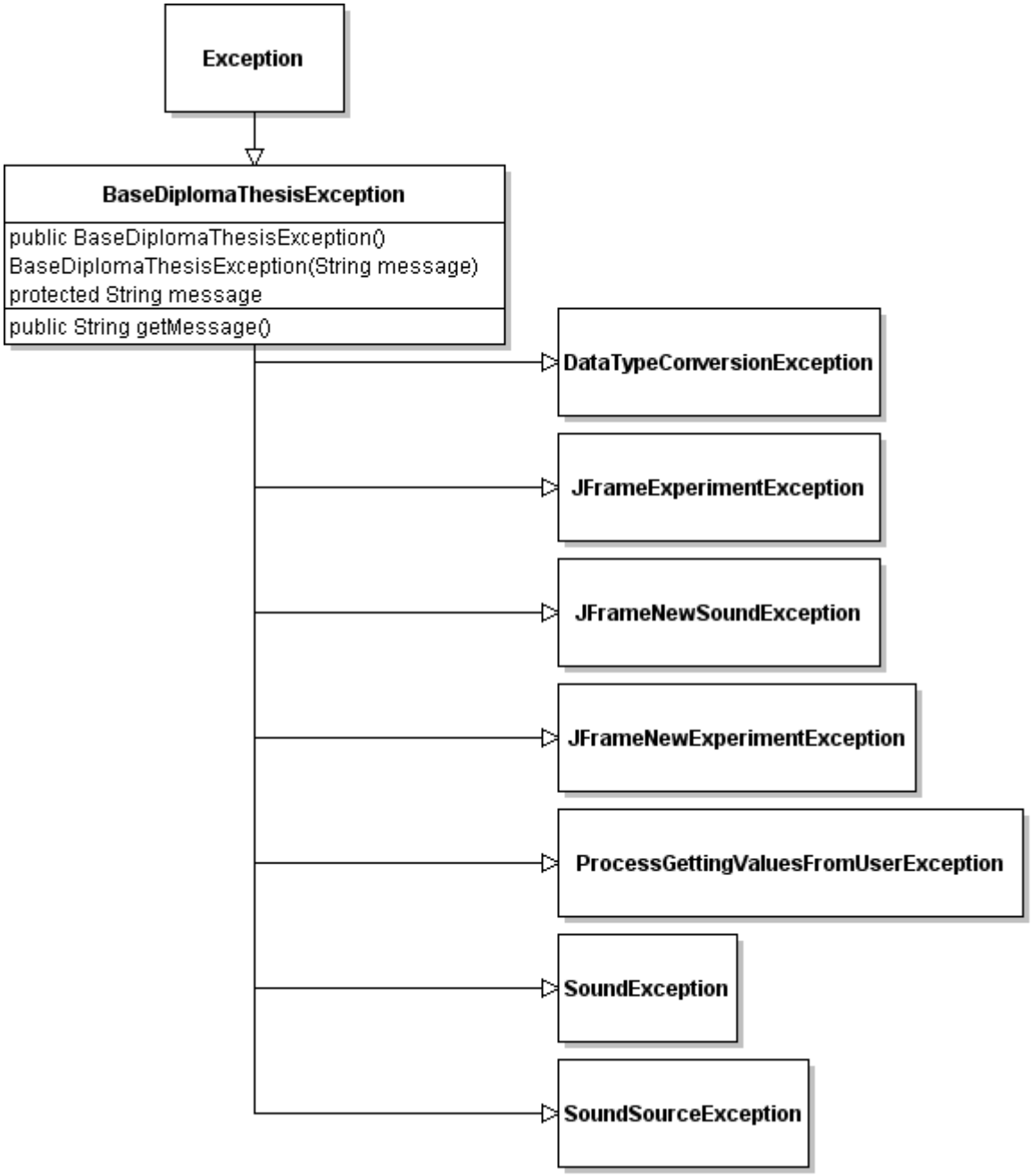
8.4 Výjimky

Na závěr je ještě nutno zmínit objekty, které se používají k ošetření chybových či neočekávaných stavů. Jedná se o tzv. výjimky (exceptions), což jsou objekty, které se vytvoří v ošetřující části kódu a v případě chyby jsou vyvolány a v konstrukci *try – catch* odchyceny. Tento mechanismus je nejlépe vidět na příkladu.

```
try {
      metoda1();
      metodaKteraVyvolavaVyjimku();
      metoda2();
} catch (Exception e) {
      // proved vypsani chyby, atd.
}
```

Vyvolá-li `metodaKteraVyvolavaVyjimku()` obecnou výjimku *e*, přeskočí se vykonání metody `metoda2()` a vykonávání kódu se přesune rovnou do části *catch*, kde s provede vypsání chyby, popř. i jiné instrukce potřebné k ošetření chybového stavu.

Každá třída této aplikace má definovánu svou vlastní třídu pro vytváření výjimek, která je děděna z rodičovské třídy `BaseDiplomaThesisException()`, která je opět děděna z obecné výjimky `Exception()`. Důvodem je přehledné, snadné a rychlé objevování chyb při ladění. Na obr. 8.17 je znázorněn UML diagram použitých výjimek.



Obr. 8.17 UML diagram výjimek.

9 Zhodnocení výsledků experimentu

Experiment jsem z časových důvodů provedl pouze sám na sobě. Z harmonických testovacích signálů byly zvoleny tóny o kmitočtech 440 Hz (a1), což je zástupce oblasti kmitočtů, kde při určování směru převládá interaurální časová diference. Na kmitočtu 2349,31 Hz (d4) je přibližná hranice, kde se mění převládající vliv interaurální časové diference na interaurální intenzitní diferenci a kmitočet 7040 Hz (a5) je zástupcem kmitočtů, u kterých při určování směru převládá interaurální intenzitní diference [7]. U tónů s kmitočty nižšími než 200 Hz jsou interaurální rozdíly minimální, takže je špatný prostorový vjem, protože vlnová délka tohoto kmitočtu odpovídá vzdálenosti ušních bubínků od sebe. Pro měření na úzkopásmovém šumu bylo proto zvoleno 15. kritické pásmo (2320 - 2700 Hz) a jako reálný zvuk bylo zvoleno drnknutí na strunu e1 na akustické kytaře.

V následujících tabulkách jsou zobrazeny hodnoty správných azimutů a elevací v konfrontaci s hodnotami, které experimentátor zvolil jako správnou odpověď. Tyto hodnoty byly pořízeny z databáze SQL dotazem. Následující SQL dotaz vybere všechny hodnoty, které změřil experimentátor *otas* na zvukovém souboru *e_string.wav*.

```
SELECT
      answer.azimuth_generated,
      answer.azimuth_answer,
      answer.elevation_generated,
      answer.elevation_answer
FROM answer
INNER JOIN sound ON sound.id_sound = answer.id_sound
INNER JOIN experiment ON experiment.id_experiment = sound.id_experiment
WHERE experiment.name = 'otas'
AND sound.source_folder = 'wav_e_string.wav'
```

Dále je pro jednotlivé hodnoty v tabulkách uvedena absolutní chyba $\Delta_{(X)}$ [12]

$$\Delta_{(X)} = X_M - X_S , \tag{7.1}$$

kde X_M je naměřená hodnota obecné veličiny X a X_S je správná hodnota obecné veličiny X. Aby bylo možné spočítat absolutní chybu, bylo nejprve nutné přepočítat jednotlivé úhly na rozsah <-180°; 180°>. Tím se předejde chybě, kdy se dvě sousední hodnoty liší o více než 180°. Hodnoty větší než 180° jsou potom brány jako záporně vzatý doplněk do úhlu 360°.

Tab. 9.1 Naměřené hodnoty na testovacím signálu harmonický tón (f = 440 Hz)

Azimut [°] (správně)	Azimut [°] (změřeno)	Δ_{azimut} [°]	Elevace [°] (správně)	Elevace [°] (změřeno)	$\Delta_{elevace}$ [°]
45	30	-15	45	15	-30
135	345	-150	15	345	-30
210	300	90	330	345	15
285	300	15	15	15	0
180	180	0	15	345	-30
150	0	-150	30	45	15
285	285	0	0	15	15
90	90	0	0	330	-30
180	180	0	30	0	-30
0	0	0	30	0	-30
330	330	0	315	345	30
15	15	0	45	0	-45
180	0	-180	15	0	-15
210	315	105	15	0	-15
180	0	-180	45	0	-45

Tab. 9.2 Naměřené hodnoty na testovacím signálu harmonický tón (f = 2349,31 Hz)

Azimut [°] (správně)	Azimut [°] (změřeno)	Δ_{azimut} [°]	Elevace [°] (správně)	Elevace [°] (změřeno)	$\Delta_{elevace}$ [°]
315	315	0	345	345	0
285	225	-60	0	345	-15
45	240	-165	30	45	15
15	210	-165	30	0	-30
75	210	-225	345	0	15
75	330	-105	30	345	-45
180	180	0	45	15	-30
60	210	-210	0	345	-15
150	270	-240	45	345	-60
315	255	-60	0	45	45
330	165	195	345	0	15
120	225	-255	45	330	-75
90	180	90	345	0	15
270	195	-75	0	30	30
45	135	90	315	330	15

Tab. 9.3 Naměřené hodnoty na testovacím signálu harmonický tón (f = 7040 Hz)

Azimut [°] (správně)	Azimut [°] (změřeno)	Δ_{azimut} [°]	Elevace [°] (správně)	Elevace [°] (změřeno)	$\Delta_{elevace}$ [°]
75	45	-30	30	330	-60
255	270	15	15	315	-60
270	270	0	315	330	15
135	315	-180	15	15	0
270	210	-60	15	330	-45
300	180	240	15	0	-15
0	0	0	330	345	15
75	315	-120	345	15	30
120	240	-240	345	30	45
300	240	-60	330	15	45
195	345	150	315	15	60
90	60	-30	15	330	-45
345	225	-120	0	330	-30
240	150	270	30	315	-75
180	195	-345	30	15	-15

Tab. 9.4 Naměřené hodnoty na testovacím signálu úzkopásmový šum (fs=2500Hz, B=380Hz)

Azimut [°] (správně)	Azimut [°] (změřeno)	Δ_{azimut} [°]	Elevace [°] (správně)	Elevace [°] (změřeno)	$\Delta_{elevace}$ [°]
45	30	-15	0	15	15
60	135	75	15	330	-45
75	180	105	0	0	0
135	345	-150	345	30	45
90	330	-120	30	45	15
255	255	0	315	315	0
330	345	15	330	30	60
195	195	0	0	30	30
120	120	0	0	345	-15
285	225	-60	330	315	-15
210	210	0	30	330	-60
90	90	0	330	30	60
195	195	0	330	330	0
300	315	15	330	315	-15
15	30	15	315	330	15

Tab. 9.5 Naměřené hodnoty na testovacím signálu drnknutí na struně e1 akustické kytary

Azimut [°] (správně)	**Azimut [°] (změřeno)**	**Δ_{azimut} [°]**	**Elevace [°] (správně)**	**Elevace [°] (změřeno)**	**$\Delta_{elevace}$ [°]**
195	195	0	0	0	0
120	120	0	0	345	-15
210	210	0	15	15	0
105	120	15	30	30	0
150	15	-135	0	345	-15
315	315	0	315	330	15
165	0	-165	15	15	0
330	345	15	330	345	15
180	180	0	345	0	15
180	0	-180	15	345	-30
150	165	15	345	330	-15
225	210	-15	330	0	30
105	75	-30	0	345	-15
315	315	0	315	345	30
165	0	-165	345	345	0

10 Závěr

V reálném světě není příliš mnoho příležitostí setkat se s harmonickými tóny, proto byla při tomto experimentu jejich lokalizace nejobtížnější a zabrala nejvíce času. Protože však sada impulsních odezev umělé hlavy byla měřena ve vzdálenosti 195 cm, což lze považovat za blízký zdroj zvuku, bylo měření na kmitočtu 440 Hz přesnější, než na zbylých dvou harmonických tónech, protože na kmitočtech v oblasti 440 Hz a u blízkých zdrojů převládá při určování směru interaurální časová diference. Měření na úzkopásmovém šumu bylo poněkud snazší, ale v porovnání s kytarovým drnknutím stále obtížné. Drnknutí na strunu e1 akustické kytary je v této práci jediným reálným zvukem a určení pozice bylo v porovnání s předchozími zvuky nesrovnatelně jednodušší, rychlejší, přesnější, ale také zábavnější. Chyby, kterých se experimentátor při měření dopustil lze přičíst především časové tísni při zacvičování a při samotném měření.

Lidský sluch je při určování pozice reálných zvuků přesnější, jelikož je na ně zvyklý z každodenního života. Dalším poznatkem při lokalizaci je fakt, že lidský sluch, především u reálných zvuků, dokáže poměrně přesně určit pozici ve smyslu vlevo – vpravo, ale stává se, že chybuje ve smyslu vepředu – vzadu. Tento druh chyby je patrný z hodnot azimutu pohybujících se okolo hodnot 0° a 180°, tedy přímo před pozorovatelem, nebo přímo za ním.

Vzhledem k tomu, že hodnoty byly naměřeny jedním subjektem a vzhledem k jejich velice nízkému počtu však nelze považovat tyto závěry za věrohodné. K vytvoření věrohodných závěrů by bylo potřeba řádově stovky až tisíce hodnot od desítek experimentátorů, což by v případě možnosti použití této aplikace na několika počítačích s jednou centrální databází nemusel být zásadní problém. Aplikaci by ale bylo nutné pro tento účel poněkud vylepšit.

Prvním vylepšením by byla možnost vytvoření uživatelské hierarchie (experimentátor a administrátor) a nutnost se do aplikace přihlašovat. Administrátor by vytvářel uživatelské účty experimentátorům, kteří by po přihlášení do aplikace neměli možnost zvolit pro měření jiného uživatele, než je on sám. Tím by se zajistilo, že naměřené hodnoty náleží skutečně tomu experimentátorovi, na kterého jsou v databázi navázány. Při vyhodnocování výsledků pak lze použít postupy, kterými je možné odhalit experimentátory vkládající záměrně chybné hodnoty s cílem poškodit výsledky experimentu, a hodnoty těchto uživatelů potom ignorovat. Aplikace by pak mohla disponovat i funkcemi zajišťujícími separaci, popř. i blokaci (znemožnění opětovného přihlášení administrátorem) těchto uživatelů – záškodníků.

Druhým návrhem pro vylepšení by byl modul zajišťující grafické zobrazení výsledků, nejlépe v 3D grafu, s různými možnostmi zobrazování a filtrace dat, např. hodnoty všech uživatelů, nebo pouze uživatelů určité věkové kategorie, popř. pouze jednoho vybraného uživatele apod.

11 Použitá literatura

[1] Melka, A. Základy experimentální psychoakustiky. Akademie múzických umění v Praze, 2005. ISBN 80-7331-043-0

[2] Schimmel, J. Prostorové a směrové slyšení. Učební text předmětu Elektroakustika, VUT v Brně, 2009.

[3] Smetana, C. a kol. Praktická elektroakustika. Státní nakladatelství technické literatury, Praha 1981.

[4] Syrový, V. Hudební akustika. Akademie múzických umění v Praze, Praha 2003. ISBN 80-7331-901-2.

[5] Merhaut, J. a kol. Příručka elektroakustiky. Státní nakladatelství technické literatury, Praha 1964.

[6] Streicher, R., Everest, F., A. The New Stereo Soundbook, 3rd ed. Audio Engineering Asociates, 2006. ISBN 978-0-9665162-1-0

[7] Blauert, J. Spatial Hearing. The MIT Press, 1997. ISBN 0-262-02413-6

[8] Smékal, Z. Signály a soustavy. Skripta VUT v Brně, 2009

[9] Smékal, Z. Číslicové filtry. Skripta VUT v Brně, 2009

[10] URL: <http://recherche.ircam.fr/equipes/salles/listen/download.html> [cit. 2010-05-18].

[11] Zwicker, E., Fastl, H. Psychoacoustics, Facts and Models, 2nd updated Ed. Springer, 1999. ISBN 3-540-65063-6

[12] Dřínovský, J. Chyby a neurčitosti měření. Učební text předmětu Radioelektronická měření, VUT v Brně, 2009

12 Seznam použitých zkratek, veličin a symbolů

FIR – Finite Impulse Response

HRIR – Head Related Impulse Response

HRTF – Head Related Transfer Function

IID – Interaural Intensity Diference

ITD – Interaural Time Diference

MLS – Maximum-Length Sequence

PCM – Pulse Code Modulation

SQL – Structured Query Language

URL – Uniform Resource Locator

UML – Unified Modeling Language

a – úhel který svírá směr příchozího zvuku s mediální rovinou

Δl – binaurální rozdíl

d – vzdálenost obou uší.

$s[n]$ – obecný diskrétní signál

$h[n]$ – impulsní charakteristika lineárního, časově invariantního systému

$w[n]$ – diskrétní signál po konvoluci

$\delta[n]$ – jednotkový impuls

X_M – naměřená hodnota obecné veličiny X

X_S – je správná hodnota obecné veličiny X

$\Delta_{(X)}$ – absolutní chyba měření obecné veličiny X

13 Seznam příloh

CD obsahující:

- tento text
- java aplikaci včetně potřebných přídavných modulů
- databázový model vytvořený v programu Toad Data Modeller
- zálohovací skript databáze včetně naměřených hodnot
- impulsové charakteristiky umělé hlavy
- sady testovacích signálů