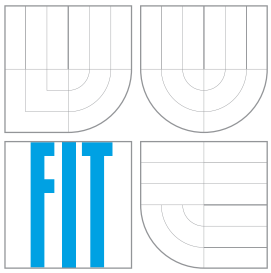
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

SIMULACE HRY NA THEREMIN POMOCÍ SENZORU LEAP MOTION

THEREMIN SIMULATION USING LEAP MOTION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

MAROŠ STAUROVSKÝ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. PAVEL NAJMAN

BRNO 2014

Abstrakt

Cílem této práce je navrhnout a implementovat aplikaci, která bude simulovat hru na hrací nástroj theremin pomocí senzoru Leap Motion za použití protokolu MIDI. Tento senzor umožňuje detekovat pohyb rukou a gest v sledované oblasti. Toho využívá výsledná aplikace na simulování stejného způsobu hraní jaký se využívá v případe thereminu. Na rozdíl od tohoto nástroje ale poskytuje možnost produkovat zvuk různých nástrojů čehož je dosáhnuto za pomocí standardu MIDI. V práci jsou pak blíže popsané vlastnosti senzoru Leap Motion a protokol MIDI.

Abstract

The goal of this thesis is the creation and implementation of application that simulates playing the theremin by using Leap Motion sensor and MIDI protocol. This sensor allows us to detect movement of hands and recognition of gestures in the monitored area. This is used in the aplication to simulate the same principle that is used for playing theremin. Unlike this instrument we are able to produce sound of different instruments which is accomplished by using MIDI. The Leap Motion sensor and MIDI protocol are detailed in the theoretical part of the thesis.

Klíčová slova

Detekce pohybu, leap motion, leap, motion, detekce, pohyb, MIDI, theremin, simulace hraní.

Keywords

Motion detection, leap motion, leap, motion, detection, motion, MIDI, theremin, play simulation.

Citace

Maroš Staurovský: Simulace hry na theremin pomocí senzoru Leap Motion, bakalářská práce, Brno, FIT VUT v Brně, 2014

Simulace hry na theremin pomocí senzoru Leap Motion

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Pavla Najmana

.....
Maroš Staurovský
20. mája 2014

Poděkování

Chcel by som sa poďakovať môjmu vedúcemu Ing. Pavlovi Najmanovi za zaujímavú tému, za jeho cenné pripomienky, čas a odbornú pomoc. Ďalej by som sa chcel poďakovať mojej priateľke a rodičom za podporu.

Obsah

Kapitola 1

Úvod

Už v prvých vedecko-fantastických filmoch sme mohli vidieť prístroje, ktoré fungovali a vykonávali príkazy bez toho, aby sa ich človek musel dotknúť. Vďaka pokroku v technológiách je tento spôsob ovládania zariadení možný. To nám otvára dvere ku mnohým, veľmi zaujímavým možnostiam. Od jednoduchého prezerania obrázkov, či prezentácií až po ovládanie chirurgických nástrojov či robotických ramien. Táto práca sa zameriava na použitie pohybového senzoru Leap Motion na simuláciu hry na hudobnom nástroji theremin. V nasledujúcich kapitolách si priblížime rôzne metódy detekcie pohybu a ich interpretácie 2. V tejto kapitole si taktiež uvedieme asi najznámejší komerčný detektor pohybu – MS Kinect a jeho vlastnosti.

V nasledujúcej kapitole 3 si popíšeme hudobný nástroj Theremin, princíp na akom funguje a prečo je vhodný na simuláciu senzorom Leap Motion.

Ďalej sa budeme zaoberať štandardom MIDI, ktorý budeme používať na tvorbu hudby 4. Uvedieme si špecifiká tohto štandardu, možnosti využitia a oblasti, kde sa využíva.

Pokračovať budeme návrhom 5 a implementáciou 6. V týchto kapitolách si popíšeme návrh aplikácie, použité prostriedky a knižnice. Rovnako si tu ukážeme ukážku rozhrania programu a uvedieme riešenie najväčších problémov, ktoré vznikli pri návrhu aplikácie.

Na záver 7 si zhrnieme získané poznatky a zamyslíme sa nad ďalšími možnosťami do budúcna.

Kapitola 2

Pohybové senzory a detekcia pohybu

V tejto kapitole sa zameráme na princípy detekcie pohybu a možnosti ich využitia. Taktiež si priblížime rôzne zariadenia, ktoré sa používajú.

2.1 Detekcia pohybu

Detekcia pohybu je proces zisťovania zmeny pozície objektu vzhľadom na jeho okolie alebo naopak, zmena okolia vzhľadom ku objektu [17]. Detekovať môžeme využitím mechanických alebo elektrických metód.

Tieto metódy môžu byť založené na:

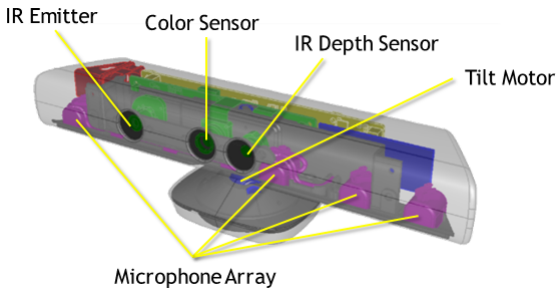
- Detekcii infračerveného svetla
- Optickej detekcii
- Detekcii zvuku
- iné

Detekcia pohybu má veľmi široké uplatnenie. Najčastejšie sa s ňou stretneme pri automatických osvetľovacích alebo zabezpečovacích systémoch, kde plní úlohu rozsvecovania svetiel, či spúšťania nahrávania kamery. Tieto systémy sú relatívne jednoduché a majú jednoznačný a nemenný účel. Okrem toho sa s detekciou môžme stretnúť aj v doprave, výrobe či hraní hier. Uplatnenie nájde v predikcii dopravnej situácie, počasia a v iných oblastiach. Nás bude zaujímať využitie detekcie pohybu v súvislosti s počítačmi. V tejto oblasti našlo využitie zatiaľ len pár prístrojov. V nasledujúcich sekciách si priblížime dva z nich. Prvým bude Kinect od spoločnosti Microsoft, druhým bude senzor Leap Motion, pre ktorý je určená vyvíjaná aplikácia.

2.2 Kinect

Kinect je produkt spoločnosti Microsoft, pôvodne určený pre použitie s hernou konzolou Xbox360. V roku 2011 boli oficiálne vydané ovládače a SDK pre operačný systém Windows 7 pod názvom *Kinect for Windows* [13]. Toto SDK je možné využiť na tvorbu aplikácií pre Kinect v programovacích jazykoch C++/CLI, C# alebo Visual Basic .NET.

Ako môžeme vidieť na nasledujúcom obrázku 2.1, Kinect obsahuje dve kamery, ktoré slúžia na snímanie priestoru a rozpoznávanie postáv.



Obr. 2.1: Senzor Kinect a jeho konštrukcia [10]

Jedna z kamier je RGB kamera, ktorá umožňuje farebné snímanie scény v rozlíšení až 1280x960 pixelov. Ďalšou súčasťou je infračervený emitor a hĺbkový monochromatický CMOS senzor. Senzor slúži na zachytenie odrazeného svetla, ktoré vydáva emitor. Odrazené lúče svetla sú prevedené na hĺbkové informácie, čím získame vzdialenosť medzi senzorom a objektom, ktorý zachytáva.

Okrem týchto dvoch kamier Kinect obsahuje sústavu štyroch mikrofónov, ktoré umožňujú určiť polohu zdroja zvuku a rovnako aj smer zvukovej vlny. Vďaka týmto mikrofónom je možné Kinect ovládať aj pomocou hlasových povelov. Ďalšou súčasťou senzoru je troj-osý akcelerometer, ktorý môže byť použitý na určenie aktuálnej orientácie Kinectu [10].

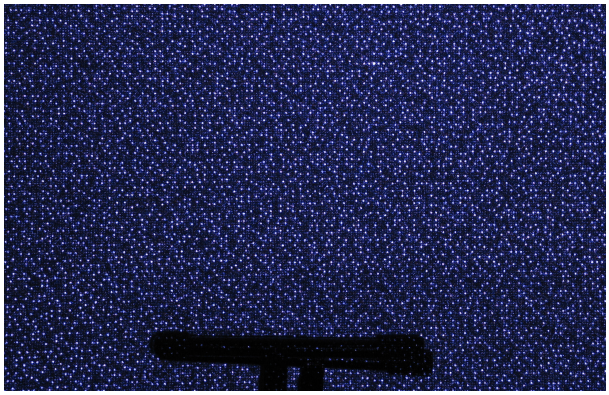
2.2.1 Princíp fungovania senzoru Kinect

Keďže ide o proprietárny senzor, neexistuje k nemu detailná dokumentácia, ktorá by uvádzala presný princíp detekcie pohybu a algoritmy, ktoré sú za to zodpovedné. Dokážeme však popísať všeobecné princípy, s ktorými senzor pracuje.

Keďže technológie, ktoré stoja za týmto senzorom, resp. časť technológií, vyvinula spoločnosť PrimeSense, môžeme podľa podaných patentov[5][15][4] zistiť aspoň približne na akom princípe je postavená detekcia objektov a teda aj gest pomocou Kinectu. Ďalším zdrojom informácií je samozrejme spätné inžinierstvo[1]. Nech už využijeme akýkoľvek zdroj dostaneme sa k tomu, že Kinect využíva infračervený emitor (laser) na premietnutie presne daného vzoru škvŕn[2] do oblasti, na ktorú je nasmerovaný. Tento vzor je vytvorený pomocou rozptyľovača (difúzora), cez ktorý prechádza laserový lúč. Na obrázku 2.2 môžeme vidieť ako tento vzor vyzerá. Táto oblasť je snímaná CMOS senzorom, ktorý má IR filter a teda umožňuje zachytávanie odrazených infračervených lúčov. Výsledný vzorec, ktorý je prijatý senzorom sa porovná so vzorom pôvodným a na základe porovnania detekuje jednotlivé objekty a ich pozíciu v obraze. Tvar a veľkosť týchto škvŕn, ktoré tvoria vzorec, sa mení na základe vzdialenosti od senzora. Kinect používa tri vzdialenostné zóny, od ktorých sa odvíja presnosť snímania pohybu. Prvá zóna poskytuje vysokú hĺbkovú presnosť pre objekty vo vzdialenosti od 0,8m do približne 1,2m. Druhá zóna poskytuje priemernú hĺbkovú presnosť vo vzdialenosti približne 1,2m až 2,0m. Posledná, tretia, zóna poskytuje najnižšiu presnosť vo vzdialenosti od 2,0m do približne 3,5m.

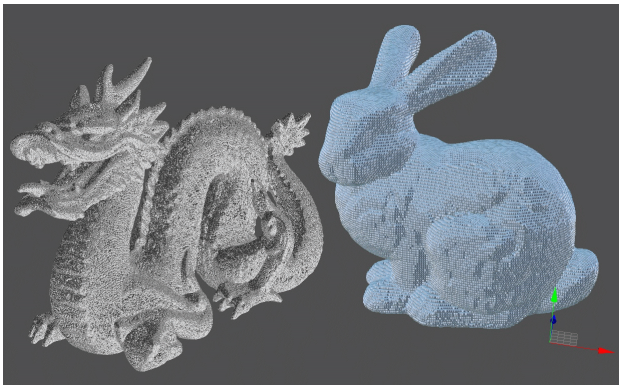
Tento vzor, ktorý Kinect používa by sme mohli označiť aj ako mračno bodov[3], ktoré

[1] reverse engineering
[2] z anglického slova speckles
[3] z anglického Point cloud



Obr. 2.2: Vzor premietaný laserom[1]

sa využívajú najmä v medicíne, animácii či na tvorbu 3D modelov. Ide o množinu bodov v 3D priestore, kde každý takýto bod nesie informácie o svojej polohe. Na tvorbu sa zvyčajne používajú 3D skenery. Na nasledujúcom obrázku 2.3 môžeme vidieť príklady objektov popísaných pomocou mračna bodov.



Obr. 2.3: Ukážka mračna bodov

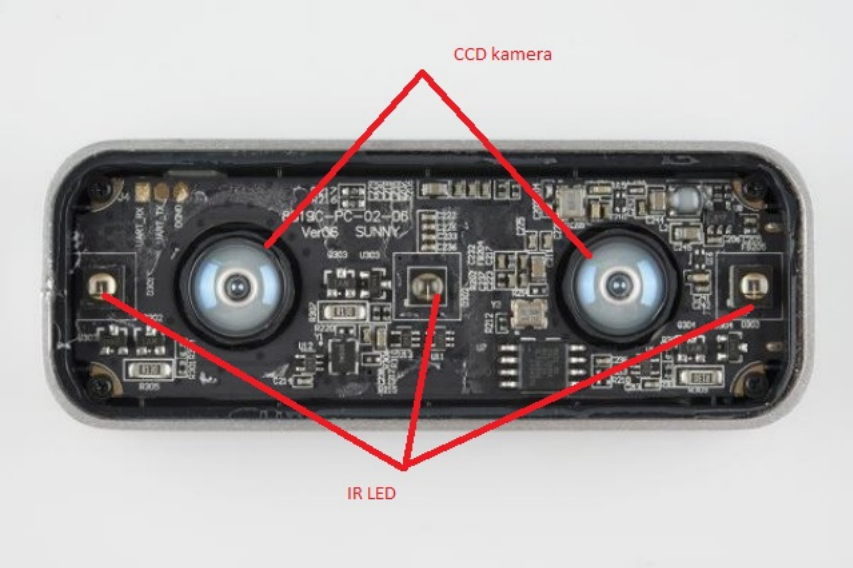
2.3 Leap Motion

Senzor Leap Motion je periférne zariadenie, ktoré umožňuje užívateľom pracovať so softvérovým vybavením počítača za použitia rúk a rôznych gest [16]. V nasledujúcej tabuľke 2.1 si môžeme pozrieť technické špecifikácie tohto senzora, ktoré uvádza na svojich stránkach spoločnosť Leap Motion.

Samotný senzor pozostáva z dvoch CCD kamier a troch infračervených LED 2.4. Tieto diódy vytvárajú nad senzorom 3D vzor, ktorý umožňuje získať hĺbkové informácie. Vďaka tomu vieme získať nielen informácie o pozícii ruky a jej pohybe, ale aj určiť jej výškovú vzdialenosť od senzora. Tento 3D vzor má približne polguľový tvar s polomerom 500mm. Celkový objem sledovaného priestoru je teda 500x500x500mm^3, pričom presnosť senzoru je približne 0,01mm [12].

Leap Motion	Špecifikácia
Výška	0,5 palca
Šírka	1,2 palca
Hĺbka	3 palce
Váha	0,1 libry
Pribalené káble	24 palcový a 60 palcový USB 2.0 (microUSB 3.0 konektory)
Minimálne sys. požiadavky	Win 7 alebo 8, alebo Mac OS X 10.7 Lion
	AMD Phenom™II alebo Intel©Core™i3, i5, i7 procesor
	2GB RAM
	USB 2.0 port
	Internetové pripojenie
Záručná doba	1 rok

Tabuľka 2.1: Technické špecifikácie senzoru Leap Motion [7]



Obr. 2.4: Senzor Leap Motion a jeho konštrukcia [8]

2.3.1 Princíp fungovania senzoru Leap Motion

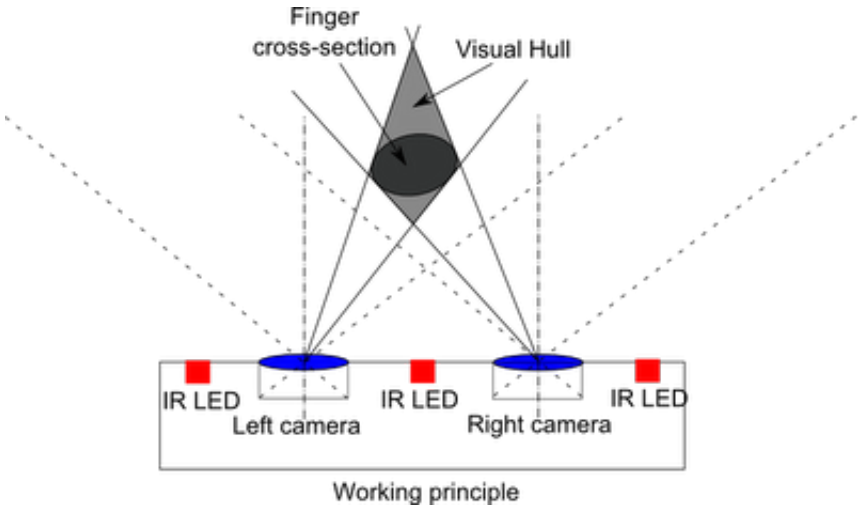
Pretože ide o jedinečný senzor vzhľadom na svoju veľkosť, cenu a schopnosti, tvorcovia senzoru neuvoľnili a pravdepodobne ani neuvoľnia väčšie množstvo informácií, ktoré by odhalili presný princíp a algoritmy, ktoré stoja za detekciou pohybu pomocou tohto senzora. Jedno je však isté, zdá sa až neuveriteľné, že takýto malý senzor, ktorý je napájaný iba pomocou USB portu, dokáže snímať danú plochu s takou presnosťou a zároveň pracovať rýchlosťou okolo 100 až 200 fps[4]. Z počiatku existoval názor, že Leap Motion je len zmenšený Kinect. To sa však rýchlo vyvrátilo nielen malou veľkosťou, cenou a napájaním, ale aj tým, že tvorcovia senzoru veľakrát dementovali použitie mračien bodov či iných snímok ako takých. Rovnako uviedli, že nedochádza k počítaniu žiadnych hĺbkových obrázkov, (máp) nakoľko je Leap Motion postavený na unikátnom matematickom modeli[3]. Z tohto dôvodu

[4]Frames per second

môžeme predpokladať, že drvivú väčšinu práce pri detekcii objektov a ich pohybu vykonáva počítač a na samotnom senzore dochádza len k relatívne malému počtu výpočtov. Existujú však rôzne teórie, ktoré sa snažia vysvetliť fungovanie senzoru Leap Motion. Jednu takúto teóriu si tu priblížime.

Keďže už vieme, že senzor nevytvára žiadne snímky mračna bodov ani nedochádza k počítaniu celkových hĺbkových máp, musíme sa zamyslieť nad inými možnosťami detekcie objektov a ich pohybu, ktoré by nám poskytli ak nie rovnakú tak aspoň podobnú rýchlosť a presnosť aká je uvádzaná pre tento senzor.

Jednou z týchto možností je metóda visual hull reconstrucion[6], čo môžeme voľne preložiť ako vizuálna rekonštrukcia telesa a vkladanie elíps (elliptical fitting). Skombinovanie týchto dvoch metód nám poskytne vysokú rýchlosť sledovania objektov a vysokú presnosť. Na obrázku 2.5 môžeme vidieť zakreslenie tohto princípu pri použití senzora Leap Motion.



Obr. 2.5: Možný princíp detekcie prstu[12]

Zároveň môžeme vidieť oblasti, ktoré snímajú jednotlivé kamery a objekt, ktorý do týchto oblastí zasahuje. Kamery zachytia odrazené infračervené lúče a jednoducho dokážu odhadnúť približnú veľkosť objektu. Do oblasti[5], ktorú získame vieme jednoducho vložiť jednu elipsu, ktorá bude buď vertikálne alebo horizontálne a bude nám slúžiť ako približný model objektu. Jedna z týchto elíps môže byť odstránená pomocou obmedzenia pomeru medzi jednotlivými osami polygónu. Z princípu tohto senzoru ako snímača pohybu rúk, prstov či predmetov uchopených v ruke a medzi prstami sa zdá prirodzené využitie vertikálnej elipsy pred horizontálnou elipsou. Prst, pero alebo ruku môžme vnímať ako kolekciu takýchto elíps, ktoré nasledujú za sebou.

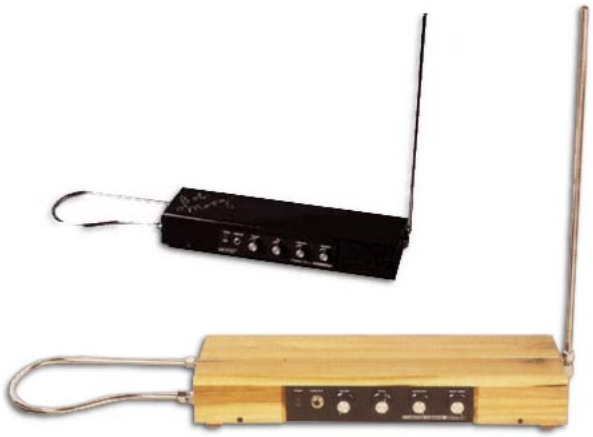
Využitie siluet objektov, vizuálnej rekonštrukcie telesa a vkladanie elíps je oveľa jednoduchšie než vytváranie mračien bodov a teda aj omnoho rýchlejšie a vďaka tomu vieme dosiahnuť aj oveľa presnejšie detekovanie pohybu a pozície objektov v sledovanom priestore.

[5] visual hull polygon

Kapitola 3

Theremin

Theremin 3.1, taktiež známy ako teremin alebo tereminvox je elektrický hudobný nástroj, ktorý vynašiel v roku 1919[1] Lev Termen (anglicky Léon Theremin). Theremin je prvý hudobný nástroj, na ktorý sa hrá bez toho, aby sa ho hráč akokoľvek dotýkal.



Obr. 3.1: Theremin

3.1 Princíp thereminu

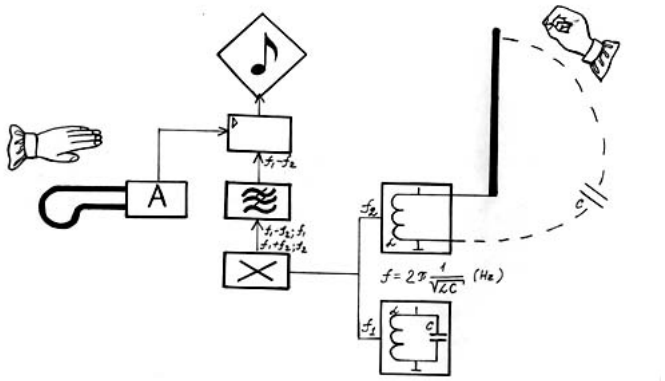
Theremin sa skladá z dvoch kovových antén, oscilátorov, cievky a kondenzátora 3.2, ktoré sú uložené v drevenom púzdre.

Hráč potom pohybuje rukami v blízkosti týchto dvoch antén, čím kontroluje hlasitosť a výšku výsledného tónu.

Na zmenu výšky tónu sa využíva princíp heterodynu, kedy zmiešaním dvoch signálov získame tretí. V prípade thereminu ide o signály, ktoré získavame z dvoch oscilátorov. Výsledkom takéhoto zmiešavania môžu byť rôzne výstupné signály. Najdôležitejšie sú tie, ktoré vzniknú sčítaním a odčítaním signálov na vstupe [2].

Pohybom ruky v blízkosti antény hráč upravuje frekvenciu LC oscilátora. V týchto oscilátoroch je frekvencia určená hodnotou induktora (L) a kondenzátora (C). Keď pripojíme

[1] niektoré zdroje uvádzajú rok 1928



Obr. 3.2: Bloková schéma thereminu

anténu ku kondenzátoru, bude sa táto správať ako ďalší kondenzátor a tak bude mať dopad na frekvenciu. Kapacitancia sa bude meniť na základe pohybu ruky v blízkosti antény, čo zapríčiní zmenu frekvencie oscilátora. Takéto zmeny sú však veľmi malé, menej ako 1%, preto je nutné použiť oscilátor s vyššími frekvenciami, napr. 500 kHz. Oscilátor fungujúci na frekvencii 500 kHz nám poskytne zmenu až do 5kHz za predpokladu maximálnej zmeny frekvencie vo výške 1%.

Keď zmiešame signál z takto zapojeného oscilátora, ktorý osciluje na frekvencii 500 kHz až 505 kHz, so signálom z oscilátora s pevnou frekvenciou 500 kHz získame rozdiel 0 až 5 kHz.

Takto získame nielen počuteľné tóny, ale aj nástroj, ktorý je schopný tieto tóny vydávať v rozsahu až piatich oktáv.

Na zmenu hlasitosti vydávaného tónu sa využíva ďalší oscilátor a anténa. Oscilátor opäť pracuje vo vysokých frekvenciách, napr. 200 kHz. Anténa je pripojená ku kondenzátoru, rovnako ako pri obvode, ktorý riadi výšku tónu. Pohybom ruky v blízkosti antény sa teda mení hlasitosť výsledného tónu. Výsledný signál je poslaný do LC obvodu, ktorý je nastavený na rovnakú frekvenciu, v tomto prípade na 200 kHz. Takýto obvod slúži na vyrušenie frekvencií iných ako 200 kHz. To znamená, že čím viac je zmenená frekvencia oscilátora, tým slabší signál získame. Výsledný signál je potom usmernený a použitý na ovládanie hlasitosti hraného tónu.

Následne je výsledný signál zosilnený v zosilovači a prehraný pomocou reproduktora.

3.2 Senzitivita

Ako už bolo spomenuté, theremin môže vydávať tóny v rôznych rozsahoch. Senzitivita v prípade thereminu je myslená ako zmena vo výške hraného tónu v závislosti na pozícii ruky hráča. Rovnaký princíp sa samozrejme vzťahuje aj ku zmene hlasitosti.

V priebehu rokov bolo mnoho návrhov thereminu s rôznou senzitivitou. Pri popisovaní senzitivity si musíme uvedomiť, že zmenu vo výške tónu spôsobuje zmena kapacitancie medzi rukou a výškovou anténou. Taktiež je dôležité určiť mieru zmeny výšky tónu či miery senzitivity. Ako ideálna jednotka sa zdá byť kilohertz na pikofarad ($\frac{kHz}{pF}$). Inak povedané o koľko kilohertzov sa zmení výška pri zmene jedného pikofarada v kapacitancii medzi rukou a anténou [11].

Aby sme lepšie pochopili, čo vplýva na zmenu frekvencie a určuje senzitivitu nástroja, musíme sa zamerať na základné stavebné prvky thereminu. Takmer všetky návrhy pracujú s LC rezonančnými obvodmi ako prvkami, ktoré určujú výslednú frekvenciu. Tieto obvody tvoria základ oscilátorov použitých vo väčšine týchto nástrojov, zvyšok obvodov poskytuje kladnú spätnú väzbu potrebnú na udržanie oscilácie.

Takýto LC obvod rezonuje na frekvencii, na ktorej sa induktancia cievky $X_L = 2\pi FL$ rovná kapacitancii kondenzátora $X_C = \frac{1}{2\pi FC}$.

Riešením týchto dvoch rovníc pre frekvenciu F je potom nasledujúca rovnica:

$$F = \frac{1}{2\pi\sqrt{LC}}$$

Pre praktické použitie je potom vhodné túto upraviť na používané jednotky mikrohenry a pikofarad. Výsledná rovnica bude vyzerať nasledovne:

$$F = \frac{159160}{\sqrt{LC}}$$

kde frekvencia F je v kilohertzoch (kHz), indukcia L je v mikrohenry (μH) a kapacita C je v pikofaradoch (pF).

Hodnota kapacity C je úplná hodnota kapacity všetkých zapojených prvkov, teda pevných aj meniteľných kondenzátorov, kapacitancia obvodu, parazitná kapacitancia antény a najdôležitejšia kapacitancia, a to kapacitancia medzi rukou a anténou. Najväčšiu časť tejto celkovej kapacity thereminu bude obvykle tvoriť kapacita kondenzátorov, pevný kondenzátor s kapacitou v rozmedzí od 100 do 1000 pF, ktorý je väčšinou zapojený paralelne s meniteľným kondenzátorom určením na ladenie. Parazitická kapacitancia antény je obvykle od 20 do 50 pF. Táto kapacitancia závisí od tvaru a veľkosti antény. Kapacitancia obvodu je zvyčajne minimálna v jednotkách pF. Kapacitancia medzi rukou a anténou býva najnižšia zo všetkých, kde jej hodnota nadobúda hodnoty od 0 pF vo veľkej vzdialenosti od antény až po jednotky pikofaradov veľmi blízko antény.

Iný spôsob ako zapísať vyššie spomenutú rovnicu na výpočet výslednej frekvencie je nasledovný:

$$F = 159160 * C^{-1/2} * L^{-1/2}$$

Keď si zoberieme deriváciu podľa C získame priamu reprezentáciu senzitivity thereminu v kilohertzoch na pikofarad:

$$\frac{dF}{dC} = -(79580 * L^{-1/2}) * C^{-3/2} = \frac{-79580}{\sqrt{L}} * \frac{1}{\sqrt{C^3}}$$

Z tejto rovnice teda vieme povedať, že senzitivita je nepriamo úmerná odmocnine z tretej mocniny celkovej kapacitancie.

Takýmto spôsobom môžeme jednoducho vypočítať senzitivitu rôznych návrhov thereminu.

Ako môžeme vidieť v tabuľke, každý návrh thereminu má odlišnú senzitivitu. Od veľmi nízkej (Clara Rockmore) až po veľmi vysokú (Theremax s minimálnou induktanciou). Dva uvedené návrhy s meniteľnou induktanciou (SWTP a Theremax) nám potvrdzujú vyššie uvedenú závislosť na induktancii tak, ako sme predpokladali podľa čiastkovej derivácie dF/dL. Pri týchto návrhoch môžeme experimentovať s rôznou senzitivitou podľa toho, ako nastavíme cievky. Podobne nám návrh Douga Forbsa umožňuje širokú škálu nastavenia kondenzátora, z ktoréhu sú v tabuľke uvedené dva príklady (100 pF a 150 pF).

Popis návrhu	Induktancia L μH	Kapacitancia C pF	Priemerná F kHz	Senzitivita kHz/pF
Clara Rockmore	1165	750	170,266	-0,114
SWTP (TECI) max L	300	410	453,803	-0,553
SWTP (TECI) min L	150	410	641,775	-0,783
Theremax max L	350	120	776,597	-3,236
Theremax min L	180	120	1082,913	-4,512
Doug Forbes C = 150	500	150	581,446	-1,938
Doug Forbes C = 100	500	100	711,763	-3,559

Tabuľka 3.1: Príklady senzitivity

Z týchto poznatkov dochádzame k záveru, že zvyšovaním základnej frekvencie znižovaním induktancie, kapacitancie alebo oboch zvýšime aj senzitivitu. Navyše pre danú frekvenciu vieme zvýšiť senzitivitu zvyšovaním induktancie L a znižovaním kapacitancie C.

Pri senzitivite je však potrebné sa zamyslieť nad tým, či je výhodnejšia vyššia alebo nižšia senzitivita. Jednoznačná odpoveď neexistuje. Čo sa však dá jednoznačne povedať je to, že staršie návrhy mali veľmi nízku senzitivitu v porovnaní s modernými návrhmi. To znamená, že na zmenu výšky bolo potrebné vykonať relatívne veľké pohyby rukou. To by sa mohlo zdať ako negatívum, avšak pri tak náročnom nástroji akým theremin nesporne je, je nižšia citlivosť skôr plusom nakoľko je potom hra na ňom jednoduchšia v porovnaní s modernými návrhmi thereminu.

3.3 Theremin a Leap Motion

Ako už bolo uvedené, senzor Leap Motion sleduje pohyb rúk a podľa toho sa správa ovládaná aplikácia. Toto sa dá využiť práve na simuláciu hrania na theremine až sa môže zdať, že pôvodný účel tohto senzora bol práve tento.

Kvôli elektrickému princípu thereminu nevieme jeho výsledný zvuk meniť tak, aby znel ako iné nástroje. To však pri Leap Motion neplatí a tak môžeme použiť štýl hrania na theremin a zároveň hrať zvuk piána, trianglu, perkusných nástrojov a podobne. V tomto ohľade nám senzor poskytuje možnosť využiť voľnosť pohybu rúk a zároveň nie sme obmedzení pevne daným zvukom thereminu. Naskytá sa nám možnosť hrať na bicie nástroje bez nutnosti ich vlastnenia či zapojenia ich elektronickej varianty alebo hrať na trúbku a nemusieť pri tom robiť nič iné ako pohybovať prstom nad senzorom.

Kapitola 4

MIDI

Každý z nás sa už so štandardom MIDI[1] vo svojom živote stretol, aj keď o tom nemusel vedieť. Tento štandard sa totiž používa napr. v klávesových syntetizátoroch ako na nasledujúcom obrázku 4.1, ale aj v iných zariadeniach, ktoré slúžia na syntézu[2] zvuku.

História týchto nástrojov začala na začiatku 20. storočia, kedy Thaddeus Cahill zostrojil prvý elektromechanický hudobný nástroj Telharmonium. V priebehu nasledujúcich rokov vznikali rôzne ďalšie nástroje ako napríklad theremin3. Prvý skutočne úspešný syntetizátor predstavil v roku 1964 Robert Moog a označil tak základ syntetizátorového hudobného žánru. Priekopníkom v oblasti syntetizátorov bol potom syntetizátor Prophet-5 firmy Sequential Circuits z roku 1978, ktorý riadil mikroprocesor a umožňoval ukladať a načítavať všetky potrebné parametre. V roku 1981 potom Yamaha uviedla na trh model GS-1, ktorý využíval FM syntézu.

Obr. 4.1: Syntetizér Roland SH-201

Pred príchodom MIDI bolo vydávanie zvuku zároveň z dvoch syntetizátorov možné buď tak, že hráč hral na oboch syntetizátoroch súčasne, čo je značne nepohodlné alebo nakúpil viac syntetizátorov od jedného výrobcu. Tieto potom vzájomne prepojil pomocou chráneného protokolu či pomocou sekvencéru rovnakej firmy. Toto bolo nutné z toho dôvodu, že všetci výrobcovia vyvíjali vlastné sekvencéry, ktoré boli chránené patentmi a teda nebolo možné prepájať zariadenia iných firiem medzi sebou.

V roku 1982 sa na výstave NAMM[3] stretli zástupcovia dvoch firiem a to Roland Corp. z Japonska a Sequential Circiuts z USA. Stretli sa z jednoduchého dôvodu, a to aby našli spôsob, akým by mohli rôzne sekvencéry medzi sebou komunikovať. Ich názory na to, ako túto víziu uskutočniť sa však líšili. Jedna strana chcela vyvinúť špičkový systém, ktorý by

[1] musical instrument digital interface – digitálne rozhranie pre hudobné nástroje
[2] Toto označenie použil prvýkrat Thaddeus Cahill v roku 1896.
[3] National Association of Music Merchants = Národná asociácia hudobných obchodníkov

však nebol dostupný masám ľudí, druhá chcela vyvinúť systém presne opačný, lacný a široko dostupný. Výsledkom ich stretnutia bol protokol UMI[4].

Do tohto projektu sa zapojili aj ďalšie spoločnosti ako Yamaha alebo Oberheim a po mnohých testoch a ladení sa v roku 1983 tento projekt ukončil vydaním nového štandardu MIDI [14].

V priebehu následujúcich rokov sa štandard rozšíril aj na osobné počítače (PC), ktoré poskytovali jednoducho programovateľné prostredie a najmä grafické rozhranie na ľahké ovládanie softvérových sekvencérov, ktoré rýchlo predčili tie hardvérové. Nové aktualizácie a špecifikácie dodávala výrobcom novo založená spoločnosť MMA[5]. Keďže o ukladaní MIDI dát vtedy nikto neuvažoval, bola MMA nútená v dobe, keď už PC mali dostatok pamäte, vydať aktualizáciu štandardu, v ktorej predstavila súborové formáty Standard MIDI (SMF), Downloadable Sounds (DLS) a eXtensible Music Format (XMF). Ďalším vylepšením bolo zavedenie štandardu General MIDI (GM), ktoré zaručovalo zvukovú kvalitu prehrávaných súborov MIDI.

4.1 Základ MIDI

- **Protokol**

  Keďže bolo MIDI pôvodne určené pre klávesové nástroje, mnoho správ z ktorých pozostáva MIDI, bolo z týchto nástrojov prevzaných.

- **Note On, Note Off**

  Note On sú udalosti, ktoré označujú začiatok hrania tónu, napr. stlačenie klávesy na piáne či brnknutie na strunu. Note Off označuje koniec hrania daného tónu. Správa MIDI však nenesie zvukové dáta, ale iba popisuje udalosť, ktorá viedla k vytvoreniu daného zvuku. Aby sme vedeli, ktorá nota bola zahraná, má každá priradené unikátne číslo. Rovnakým spôsobom zaznamenávame aj to, ako tvrdo alebo mäkko bol tón zahraný.

- **Prenos MIDI informácií**

  Informácie v MIDI sú prenášané sériovo a nie paralelne, a je dôležité na to brať zreteľ. Výber seriového prenosu nad paralelným je daný historicky. V dobe vzniku štandardu mal paralelný prenos dát určité nevýhody, ktoré prevážili jeho výhody. Išlo predovšetkým o nákladnosť, ktorá by spôsobila nedostupnosť pre bežných používateľov. Ďalej káble, konektory a zásuvky boli komplikované a náchylné na poškodenie. Naproti tomu sériové rozhranie bolo lacnejšie a spoľahlivejšie, čo mu otvorilo cestu na masový trh. Navyše jeho prenosové vlastnosti vtedy bohato postačovali výrobcom aj užívateľom. V záujme udržania kompatibility sa spôsob prenosu dát do dnešnej doby nezmenil.

  MIDI posiela dáta rýchlosťou 31 250 bps[6]. Toto označenie sa nazýva **baud rate**. Táto rýchlosť bola vybraná pretože je celým deliteľom 1MHz, čo je frekvencia, na ktorej pracovali prvé mikroprocesory[9]. Každý bajt MIDI správy zaberá 10 bitov. Z týchto 10 bitov zaberá samotná MIDI informácia 8 bitov a 2 bity (start bit a stop bit) sú použité na označenie začiatku a konca každého bajtu a tvoria tak rámcové ohraničenie

[4] Universal Musical Intrument
[5] MIDI Manufacturer's Association
[6] bits per second

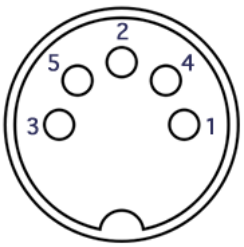
správy. Prvý bit označuje, či ide o stavový bajt alebo o datový bajt, za týmto prvým bitom nasleduje 7 bitov samotnej informácie.

Z toho vieme odvodiť, že MIDI pošle za sekundu približne 3906 bajtov dát. Ak si to porovnáme s dátovým prenosom 176 400 bajtov potrebnou na prenos digitálneho zvuku v kvalite audio-CD, MIDI sa môže zdať pomalé. Keďže však MIDI neprenáša audiodáta, nepotrebuje tak vysoké prenosové rýchlosti a plne mu postačuje sériový prenos. V skutočnosti je možné s MIDI prehrávať až 500 nôt za sekundu a aj komplikovaný akord znie tak ako má a všetky noty sú zahrané zároveň.

- **Hardvér a konektory**

Informácie MIDI putujú káblami, konektormi na koncoch kábla a zásuvkami, do ktorých sú konektory zasunuté. Dĺžka káblu môže byť až 15 metrov. Je treba mať na pamäti, že čím dlhší je kábel tým dlhšie bude trvať prenos dát z bodu A do bodu B. Ďalší problém, na ktorý je potrebné myslieť, je slabnutie signálu so vzrastajúcou vzdialenosťou, ktorú musí prejsť. Toto slabnutie energie signálu spôsobuje deformácie signálu prípadne jeho výpadky. Tento problém sa rieši zapojením signálových zosilovačov.

Samotný MIDI konektor je 5 kolíkový DIN jack 4.2. Na oboch stranách kábla je rovnaký konektor a to samec. V prípade, že chceme kábel predĺžiť, je potrebné použiť špeciálny kábel s konektormi DIN typu samec/samička.



Obr. 4.2: MIDI konektor

Nie všetky kolíky sú však aktívne. Tabuľka 4.1 popisuje funkcie jednotlivých kolíkov.

Číslo kolíku	Popis
1	Nepoužíva sa
2	Používa sa ako tienenie
3	Nepoužíva sa
4	Slúži na príjem dát
5	Slúži na posielanie dát

Tabuľka 4.1: Popis kolíkov MIDI konektora

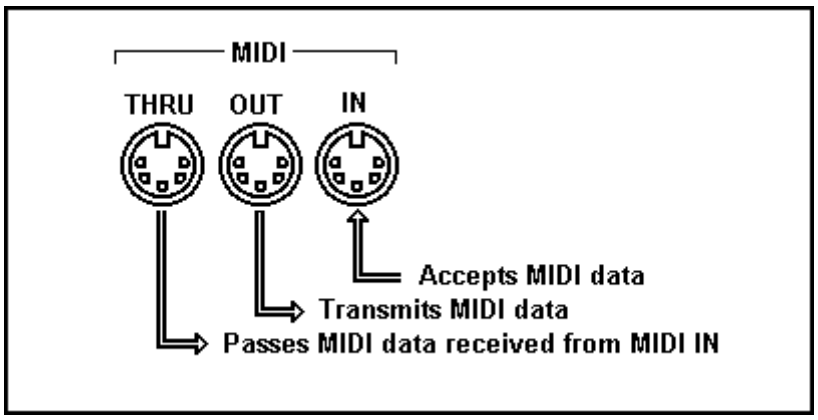
- **Zásuvky MIDI**

Zásuvky[7] na nástrojoch, MIDI merge boxoch (patch bay) alebo na rozhraniach MIDI sú samičou verziou konektorov MIDI. Klávesy (ale aj iné nástroje) majú väčšinou

[7] Z anglického sockets

tri zásuvky 4.3. Ako vidíme na obrázku, tieto sú označené ako In, Out a Thru. Posledná menovaná sa väčšinou nenachádza na počítačových rozhraniach alebo na merge boxoch. Dôvod prečo to tak je si uvedieme nižšie.

Informácie MIDI prúdia jedným smerom a nezáleží teda na tom, ktorý koniec zapojíme do nástroja a ktorý do počítačového rozhrania či do merge boxu. Je to dané tým, že zásuvka na nástroji je vedená ako výstup a zásuvka na rozhraní je vedená ako vstup.



Obr. 4.3: MIDI zásuvka

MIDI Out

MIDI Out slúži ako výstup pre MIDI. Čo je posielané cez túto zásuvku, sú správy, ktoré nástroj či zariadenie vytvára. Je dôležité pochopiť, že z tejto zásuvky vychádza len výstupný signál. Z tohto dôvodu ak chceme prepojiť medzi sebou viac ako jedno zariadenie, je potrebné použiť zásuvku MIDI Thru. Táto zásuvka funguje tak, že akúkoľvek informáciu, ktorá dorazí do zásuvky MIDI In presmeruje na MIDI Thru.

MIDI Out neposiela žiadne audiodáta, ale správy MIDI, ktoré iné zariadenie príjme svojou zásuvkou MIDI In a interpretuje ich. V zásade teda platí, že výstupy MIDI Out by mali byť prepojené so vstupmi MIDI In.

MIDI In

MIDI In slúži ako vstup pre MIDI. Dáta, ktoré sú prijaté touto zásuvkou, sú spracované hardvérovým alebo softvérovým procesorom. Ak si prajeme použiť tieto dáta v inom zariadení MIDI, sú tieto poslané ďalej cez MIDI Thru. Dáta MIDI v podstate hovoria napr. zvukovej karte aby prehrala notu určitou silou, na danom kanáli a za použitia daného zvuku uloženého v pamäti. Takto syntetizátor produkuje zvuk cez audiovýstup. Na samotnom audiovýstupe potom môžeme zaznamenávať samotné audiodáta, ale nie správy MIDI.

Zásuvka MIDI In môže prijímať dáta ako z MIDI Out tak aj z MIDI Thru.

MIDI Thru

MIDI Thru, ako sme si už spomenuli vyššie, funguje tak, že informácie, ktoré prídu do zásuvky MIDI In sú presmerované na túto zásuvku. Prístroj však na túto zásuvku nezasiela dáta, ktoré sám vytvára. Takto sa dajú posielať dáta z jedného zariadenia na viac zariadení prostredníctvom tejto zásuvky. Takéto preposielanie informácií sa nazýva reťazenie zariadení.

- **Reťazenie zariadení**

  Reťazenie zariadení[8] umožňuje odovzdávať informácie jedného riadiaceho zariadenia (Master) ostatným zariadeniam (Slave). Môžme to využiť napríklad ak chceme dáta vysielané počítačom odovzdať viacerým externým zariadeniam bez použitia viacportového rozhrania MIDI alebo bez merge boxu. Reťazenie je síce efektívny spôsob ako prepojiť viac zariadení bez nutnosti použiť špeciálny hardvér má však aj svoje nevýhody.

  Ako sme si už vraveli, MIDI využíva sériový prenos a to znamená, že čím dlhšie cestuje informácia káblami a rôznymi zariadeniami, tým dlhšie jej trvá kým doputuje tam kam má. Zároveň využívanie zásuvky MIDI Thru zaťažuje do určitej miery procesory jednotlivých zariadení. To znamená, že ak zapojíme za seba viacero zariadení, tak dôjde k oneskorovaniu signálu. Preto je základným pravidlom reťazenia neprepájať viac ako tri zariadenia. V prípade, že potrebujeme prepojiť viac zariadení je nutné použiť viacportové rozhranie alebo merge box. Obe zariadenia si bližšie popíšeme v sekcii 4.3. V reťazci má každé zariadenie priradený vlastný kanál a teda môže prehrávať iný zvuk.

- **Formáty súborov MIDI**

  Aby bolo možné skladby opätovne prehrávať či upravovať, boli na ukladanie MIDI správ zavedené súborové formáty ako napr. Standard MIDI File (SMF). Existuje však mnoho rôznych formátov, ktoré sa odvíjajú od aplikácie, ktorá bola použitá na uloženie skladby. Je to z toho dôvodu, že jednotlivé aplikácie okrem MIDI informácií ukladajú do súboru aj rôzne nastavenia programu. My sa budeme bližšie zaoberať len formátom SMF.

  Tento formát v sebe ukladá všetky nutné informácie MIDI a pre každú udalosť má priradené časové razítko (time stamp), podľa ktorého sekvencér určí, kedy danú udalosť prehrať.

  Výhoda MIDI spočíva okrem iného v tom, že výsledný MIDI súbor zaberá málo miesta. Je to z toho dôvodu, že pri zázname sa neukladá samotný zvuk, ale iba jednotlivé udalosti, ktoré vedú k vzniku tohto zvuku. Takýto súbor potom funguje podobne ako návod alebo recept, ako použitím zvukov uložených v pamäti vytvoriť výslednú skladbu. Preto MIDI súbory zaberajú niekoľkonásobne menej miesta na disku ako audio súbory.

  Takýto spôsob záznamu má však aj svoje nevýhody, a to hlavne tú, že výsledný zvuk záznamu sa bude líšiť v závislosti od zariadenia, na ktorom je prehrávaný, na rozdiel od klasických audio súborov, ktoré budu znieť na všetkých zariadeniach rovnako.

  Existujú tri typy súborového formátu SMF, ktoré popisuje nasledujúca tabuľka.

Typ MIDI	Počet stôp	Obsahuje všetky správy MIDI	Počet skladieb
0	1	áno	1
1	1 na kanál	áno	1
2	1 na kanál	áno	neobmedzene

Tabuľka 4.2: Vysvetlenie rôznych typov SMF

[8] Daisy Chaining Devices

Typ 0 obsahuje len jednu skladbu a všetky kanály sú spojené do jednej stopy. Typ 1 obsahuje rovnako ako typ 0 len jednu skladbu, avšak každý kanál je uložený na vlastnej stope. Typ 2 môže obsahovať neobmedzené množstvo skladieb, kde si každá skladba nesie svoje informácie o tempe.

Ukladanie do formátu SMF je vhodné vtedy, keď potrebujeme takýto súbor prenášať medzi rôznymi počítačmi, ktoré nemusia mať rovnaký operačný systém, nakoľko drvivá väčšina vývojárov podporuje tento formát.

- **Hierarchia MIDI – kanály a porty**

  Pri používaní MIDI je veľmi pravdepodobné, že hráč bude chcieť využiť viac ako jeden hudobný nástroj a na to slúžia kanály a porty MIDI.

  Kanál MIDI slúži na izolovanie informácií pre rôzne nástroje a teda pri použití viacerých nástrojov vieme zabezpečiť, aby nástroj prehral či inak spracoval len informácie preň určené a ostatné informácie, ktoré sú určené pre iné nástroje, jednoducho ignoroval na základe čísla kanálu.

  Port MIDI alebo zbernica MIDI sa skladá zo zásuviek MIDI In, Out a Thru. Väčšina zariadení má jeden port, ale môžeme sa stretnúť aj s takými, ktoré majú portov viac. Každý takýto port dokáže pracovať s až šestnástimi kanálmi. Môžeme teda využiť až 16 nástrojov, ktoré môžu hrať naraz. V prípade, že zariadenie má viac ako jeden port MIDI, získame ďalších 16 kanálov na každom porte. Napríklad na zariadení, ktoré má 2 porty MIDI možme využiť až 32 kanálov. Viac portov však môže znamenať aj to, že prístroj iba posiela rovnaké informácie na viac výstupov. Na zistenie, či ide o multiportové alebo multivýstupové zariadenie je nutné nazrieť do dokumentácie daného zariadenia. Rozdiel medzi týmito dvoma zariadeniami je veľmi podstatný. Prvé zariadenie má viacero portov, pričom na každý port dokáže posielať 16 kanálov zároveň. Takže ak by malo 3 porty MIDI, malo by 48 kanálov. Druhé zariadenie má jeden port s viacerými, napr. tromi, výstupmi, na ktoré posiela rovnakých 16 kanálov. V oboch prípadoch majú teda zariadenia rovnaký počet výstupov avšak líšia sa v počte používaných kanálov.

  Používanie multiportových zariadení má jednu veľkú výhodu. Pretože MIDI využíva sériový prenos, sú všetky dáta prenášané postupne, čo môže pri veľkom počte použitých nástrojov spôsobiť zahltenie ciest. Tento jav spôsobí zlyhanie dátového toku a môže sa stať, že niektoré nástroje neobdržia korektné informácie, čo má za následok výpadky či oneskorenie počas prehrávania hudby. Tomuto sa dá efektívne zabrániť práve používaním viacerých portov MIDI.

  Ako bolo spomenuté, vždy máme prístup minimálne k šestnástim kanálom. Na každom z týchto kanálov môžeme mať priradený práve jeden zvuk. Teda napríklad kanál 1 má priradený zvuk piána, kanál 2 zvuk kontrabasu, kanál 3 zvuk píšťaly atď. Je samozrejmé možné mať priradený rovnaký zvuk na viacerých kanáloch. K jednému kanálu môžeme teda mať priradený len jeden zvuk, ale v prípade potreby je možné tento zvuk jednoducho zmeniť na iný. Robí sa to rovnako ako zmena noty, stačí na ovládacom paneli vybrať iný zvuk. Rovnako ako pri zmene nôt budú na túto zmenu reagovať ďalšie zariadenia v reťazci.

- **Multitimbrálne nástroje**

  Dnešné zariadenia, ktoré pracujú s MIDI sú zvačša multitimbrálne, čo znamená, že dokážu prijímať a prehrávať informácie naraz zo šestnástich rôznych kanálov MIDI.

Každý kanál, ktorý zariadenie prijíma, je možné nastaviť na iný zvuk. Multitimbrálny nástroj dokáže oddeliť jednotlivé správy MIDI a odoslať ich správnemu programu, ktorý ho potom prehrá. Na týchto nástrojoch je možné zapnúť alebo vypnúť príslušný kanál MIDI za účelom lepšej kontroly toho, ktorý kanál informáciu spracuje a ktorý nie. Keď je kanál vybraný, správa MIDI pre tento kanál bude prehraná za použitia priradeného zvuku.

4.2 MIDI správy a General MIDI

Channel Voice – správy tohto typu tvoria jadro MIDI. Drvivá väčšina toho, čo MIDI počas hrania posiela, sú tieto správy. Popíšeme si teraz všetky správy typu Channel Voice, ktoré sú pre túto prácu podstatné.

- **Note on**

  Správa Note On sa, ako už bolo povedané, vysiela keď je stlačená klávesa a teda chceme zahrať určitý tón. Táto správa obsahuje dve informácie a to číslo zahranej noty a sílu (velocity), ktorou bola nota zahraná. Obe uvedené informácie môžu nadobúdať hodnoty od 0 do 127. Čísla jednotlivých nôt si uvedieme neskôr v implementácii. V prípade sily akou bola nota zahraná je to jednoduché, čím vyššie číslo, tým silnejšie bola nota zahraná. Inak povedané, vieme určiť, či bola nota zahraná forte, mezzo forte alebo pianissimo a podobne. Ak je hodnota velocity rovná nule, je to rovnaké akoby nota hraná nebola.

  Obvykle sa ako hodnota velocity používa mezzo forte, ktoré reprezentuje velocity s hodnotou 64 až 79. Je to dané tým, že staršie modely kláves, ktoré používali MIDI, prenášali iba túto úroveň velocity. Moderné klávesy sú schopné prenášať všetky úrovne. To, ako hodnota velocity ovplyvňuje výsledný zvuk je dané aj samotným zvukom. Vyššia hodnota velocity môže znamenať čistejší zvuk, inokedy hlasnejší či sa môže výsledný zvuk zdať úplne iný.

- **Note off**

  Táto správa sa posiela vždy pri uvoľnení stlačenia klávesy. Väčšina zariadení používa v týchto prípadoch správu Note On s velocity 0. Note Off je možné použiť s hodnotou Release Velocity, ktorá určuje akou rýchlosťou bola klávesa pustená a ovplyvňuje postupné doznievanie zvuku. Posielanie tejto hodnoty však musí dané zariadenie podporovať.

- **Zmena programu**

  V dnešnej dobe sú zvukové moduly vybavené pamäťou, ktorá v sebe uchovává rôzne programy (zvuky). Názvy zvukov sa samozrejme môžu líšiť od výrobcu k výrobcovi. Pretože sa k zvukom pristupuje pomocou čísla, môžeme aktuálny zvuk meniť pomocou správy Program Change. Pomocou tejto funkcie máme prístup k až 128 zvukom. Pri poslaní tejto správy zareaguje zariadenie nastavené na určitý kanál tak, že zmení pre tento kanál zvuk.

  Pre zaistenie kompatibility skladieb medzi rôznymi zariadeniami bolo nutné zaviesť určitý štandard pri číslovaní jednotlivých programov. Bez takéhoto štandardu by mohlo dôjsť k situácii keď na jednom zariadení by bolo nastavené piáno a na druhom bicie pričom číslo programu by bolo rovnaké. Tento štandard, ktorý zjednoťuje číselné označenie programov, sa nazýva General MIDI (GM).

- **Zmena výšky tónu**

  Zmena výšky tónu alebo Pitch Bend Change umožňuje efekt, kedy jedna nota plynulo prechádza do noty nasledujúcej. Ľudské ucho je veľmi citlivé na zmeny vo výške tónu a preto má Pitch Bend rozlíšenie až 16 384 krokov. Toto sa potom ďalej delí na dve polovice, ktoré sa skladajú z 8192 krokov. Rozsah Pitch Bend je teda -8192 do +8192, kde 0 je pôvodná hodnota pitch. Špecifikácia GM udáva, že funkcia Pitch Bend by sa mala používať maximálne v rozmedzí +- 2 polkroky.

Control Change – na moderných, prípadne starších drahších, klávesách nájdeme okrem kláves aj množstvo jazdcov, koliesok, pedálov, ... Tieto sa používajú na reguláciu hodnoty rôznych efektov či vlastností hranej skladby alebo jednotlivých nôt. Rovnako ako klávesy aj tieto zariadenia posielajú správy MIDI, ktoré môžeme zaznamenať a reprodukovať. Pretože množstvo týchto správ je veľmi veľké, uvedieme si len tie, ktoré nás v rámci tejto práce zaujímajú.

- **Dĺžka trvania portamenta**

  Portamento Time alebo Dĺžka trvania portamenta udáva, za aký čas sa výška jednej noty premení na výšku nasledujúcej noty.

- **Hlasitosť kanálu**

  Channel Volume alebo Hlasitosť kanálu slúži pri multitimbrálnych zariadeniach na ovládanie úrovne hlasitosti jednotlivých kanálov, prípadne na ovládanie spoločnej úrovne.

General MIDI – ako už bolo spomenuté, za účelom zjednotenia číselných označení programov MIDI vznikol štandard General MIDI. Na to, aby zariadenie získalo certifikát General MIDI musí spĺňať požiadavky General MIDI System Level 1 a to okamžite po nákupe zariadenia. Dodatočné opravy nie sú prípustné. Ak budeme používať rôzne zariadenia, ktoré sú kompatibilné s GM, môžeme na nich prehrávať rovnakú skladbu, ktorá bude znieť rovnako na všetkých týchto zariadeniach. Takto zaznamenaná skladba sa môže ukladať vo formáte Standard MIDI a ďalej šíriť. Keďže GM upravuje viacero aspektov kompatibility MIDI zariadení, uvedieme si iba tie, ktoré zohrávajú podstatnú úlohu v tejto práci a výslednej aplikácii.

- **Patche General MIDI**

  Jednou zo základných vlastností GM je usporiadaný zoznam zvukov, ktoré musí výrobca do svojho zariadenia nahrať aby spĺňal štandard. Kategórie týchto zvukov vidíme v následujúcej tabuľke 4.3.

4.3 MIDI zariadenia

S protokolom MIDI pracuje široká škála zariadení od klávesových nástrojov, gitár až po rôzne nástroje, ktoré sú určené výlučne na prácu s týmto štandardom ako napríklad MIDI merge boxy alebo sekvencéry. V súvislosti s MIDI môžeme hovoriť aj o počítačoch ako o MIDI zariadeniach, nakoľko dnešné počítače podporujú tento štandard a so správnym softvérom dokážu skladby vo formáte MIDI nielen prehrávať, ale aj upravovať či vytvárať.

- **Merge box**

Číslo programu	Nástroj	Číslo programu	Nástroj
1-8	Piano	65-72	Jazýčkové nástroje (Reed)
9-16	Chromatické Perkusie	73-80	Píšťalové nástroje
17-24	Orgán	81-88	Synth Lead
25-32	Gitara	89-96	Synth Pad
33-40	Basa	97-104	Synth efekty
41-48	Sláčikové nástroje	105-112	Etnické nástroje
49-56	Súbor (Ensemle)	113-120	Perkusie
57-64	Mosadzné nástroje (Brass)	121-128	Zvukové efekty

Tabuľka 4.3: Kategórie hudobných nástrojov podľa General MIDI

Merge box je zariadenie, ktoré obsahuje viacero MIDI vstupov a výstupov. Tieto zásuvky môžu reprezentovať buď jeden port alebo môže každá dvojica MIDI In a MIDI Out reprezentovať samostatný port, čím sa z merge boxu stane multiportové zariadenie, resp. multiportový box MIDI. Tieto zariadenia sa používajú, okrem iného, najmä na smerovanie správ MIDI (MIDI routing). Pomocou tohto smerovania vieme určovať prepojenie vstupov MIDI In s výstupmi MIDI Out. To znamená, že informácia, ktorá vstupuje cez vstup označený ako MIDI In 1 nemusí vychádzať zo zariadenia cez výstup označený ako MIDI Out 1, ale môže vychádzať cez výstup MIDI Out 3 alebo 5. Zároveň je možné posielať vstupujúcu informáciu na viac ako jeden výstup, takže prijímaný signál bude poslaný na viacero zariadení, ktoré obdržia tú istú informáciu, ktorá prišla jedným vstupom.

Okrem smerovania sa potom merge boxy využívajú aj na filtráciu určitých správ MIDI, ukladanie rôznych nastavení a podobne.

- **Multiportové rozhranie PC**

  V prípade, keď používame počítač na tvorbu alebo prácu s MIDI máme väčšinou k dispozícii jeden port MIDI, teda jeden vstup a jeden výstup, ktorý nám dáva prístup k 16 kanálom. Ak nám však tento počet nepostačuje, je nutné siahnuť po multiportovom rozhraní. Tieto rozhrania je možné k počítaču pripojiť prostredníctvom USB alebo sériového portu počítača. Prostredníctvom tohto rozhrania teda získame viac MIDI portov, to znamená viac vstupov a výstupov. Získame tak aj viac kanálov a to priamo úmerne počtu portov, takže napríklad rozhranie s tromi portami nám umožní využívať až 48 kanálov zároveň.

- **Sekvencér**

  Sekvencér je nástroj, ktorý umožňuje zaznamenávať udalosti MIDI a potom ich prehrať rovnakým spôsobom. Takto zaznamenané udalosti je možné potom upravovať, prehrávať inou rýchlosťou, pracovať s kanálmi, vytvárať kópie udalostí a vkladať ich podľa potreby, prevádzať záznam MIDI do notového zápisu a mnoho ďalšieho. V dnešnej dobe sa stretneme ako s hardvérovými tak so softvérovými sekvencérmi.

Kapitola 5

Návrh

Cieľom tejto práce je navrhnúť a implementovať program, ktorý bude využívať senzor Leap Motion a protokol MIDI na simuláciu hry na hudobný nástroj theremin. Pri návrhu bolo potrebné vziať do úvahy nielen technické špecifikácie senzoru ako napr. veľkosť snímanej plochy, ale aj špecifikácie protokolu MIDI a samozrejme zohľadniť princíp fungovania thereminu.

Hlavným problémom bolo premietnuť zmenu výšky tónu tak, ako k nej dochádza pri theremine do výslednej aplikácie za použitia MIDI. Toto sa dá dosiahnuť len čiastočne, nakoľko táto zmena sa pri theremine deje postupne zmenou frekvencie, avšak pri použití štandardu MIDI sa výška tónu mení iba tak, že sa daný tón resp. nota zahrá. Tento rozdiel spôsobuje to, že aj keď je prechod medzi notami plynulý, okamih kedy je nota zahraná je výraznejší než prechod z jednej noty do druhej pri theremine.

Zároveň bolo dôležité uvedomiť si a náležite pracovať so súradnicovým systémom, ktorý používa senzor Leap Motion, pretože na najdôležitejšej ose x je možné dosiahnuť ako plusové tak aj mínusové hodnoty, ktoré však MIDI nepodporuje. Taktiež súradnicový rozsah je väčší než rozsah nôt, ktoré môžeme zahrať a preto bolo potrebné zvoliť vhodný pomer zmeny v pozícií k posunu medzi notami, aby sme boli schopní zahrať čo najväčšiu škálu tónov.

V tomto bode sa opať dostávame k thereminu a jeho senzitivite, ktorú sme si spomínali v kapitole 3, presnejšie v časti 3.2. Rovnako ako musíme brať do úvahy túto vlastnosť pri samotnom hudobnom nástroji, musíme ju zvážiť aj pri návrhu toho, aký pohyb bude korešpondovať so zmenou v hranej note. Príliš vysoká senzitivita by nám bránila v pohodlnom hraní, nakoľko by sa aj zmena v polohe prsta len o niekoľko milimetrov prejavila veľkou zmenou v hraných notách. Na opačnú stranu príliš nízka senzitivita by mala za následok malý notový rozsah, ktorý by nás opäť mohol zbytočne obmedzovať tým, že veľký pohyb prsta by mal len malý efekt na zmenu nôt.

Na implementáciu návrhu som sa rozhodol pre jazyk Java. Okrem zjavných a dobre známych výhod tohto jazyka ako je prenositeľnosť som si ho vybral aj z toho dôvodu, že je naň k dispozícii Leap Motion SDK[1], ktoré je jedinou možnosťou ako vyvíjať aplikácie pre tento senzor. Toto SDK si bližšie popíšeme v časti 5.1 kde si priblížime spôsob akým vieme pristupovať k informáciám, ktoré nám Leap Motion poskytuje. Ďalšou výhodou javy je aj to, že jej základnou súčasťou je aj knižnica protokolu MIDI, ktorá poskytuje všetky potrebné triedy a metódy a teda je zaručená prenositeľnosť a to, že výsledný zvuk bude na všetkých zariadeniach rovnako prehraný. Túto knižnicu si bližšie popíšeme v časti 5.2. V tejto časti si taktiež stručne popíšeme použité triedy a ich metódy, pomocou ktorých

[1] Software Development Kit

generujeme zvuk a jeho hlasitosť na základe informácii od senzoru.

5.1 Leap Motion SDK

Ako už bolo povedané, jediná možnosť ako vyvíjať aplikácie pre senzor Leap Motion, je za využitia SDK, ktoré je vývojárom k dispozícií a je postupne aktualizované o nové metódy, dokumentáciu alebo rôzne príklady. Súčasťou SDK sú teda príklady jednoduchých aplikácií, dokumentácia a knižnice potrebné pre niekoľko programovacích jazykov ako Java alebo C#.

Základným stavebným kameňom pri tvorbe aplikácii na senzor Leap Motion sú triedy *Listener* a *Controller*. Objekt triedy Controller, ako už názov napovedá, slúži ako prostriedok, cez ktorý prúdia informácie zo senzora do systému. Objekt triedy Listener slúži na zachytávanie týchto informácií a ich následné spracovanie. Programátor má potom prístup ku všetkým informáciám, ktoré sú na vytvorenie programu potrebné.

Ide najmä o metódu *onFrame*, ktorá poskytuje informácie z najnovšej snímky ako počet rúk, prstov alebo nástrojov. Je dôležité uvedomiť si, že nedochádza k prenosu grafických snímok, ale iba informácií o tom, aké objekty sa v snímke snímanej senzorom vyskytujú, aká je ich poloha a vzájomná závislosť. Pod závislosťou si môžeme predstaviť napríklad príslušnosť prstov k danej ruke. Jednotlivé informácie potom získavame prostredníctvom nasledujúcich tried:

- *HandList* a *Hand*

  Trieda HandList nám poskytuje zoznam všetkých zaznamenaných rúk. Z tohto zoznamu vieme získať informácie o počte rúk, o tom, ktorá ruka je najviac vpravo alebo vľavo či o tom, ktorá ruka je najviac vpredu.

  Trieda Hand nám poskytuje informácie o vybranej ruke, jej pozícii, rýchlosti, počte prstov priradených k danej ruke a množstvu ďalších vlastností a metód, ktoré nám umožňujú definovať správanie v závislosti na dátach, ktoré získame od senzoru.

  Ďalšími informáciami, ktoré v rámci tejto triedy vieme získať okrem údajov o jej rýchlosti pohybu, sú informácie o jej natočení, zdvihu a podobne.

  Taktiež vieme získať polomer teoretickej gule, ktorá by sa do ruky zmestila pri rôznej polohe prstov. Toto sa dá využiť napríklad na ovládanie 3D programov, kde potrebujeme uchopiť model a otáčať ním prípadne ho presúvať na inú pozíciu. Rovnako by sa to dalo využiť v hrách s rožšírenou realitou[2], kde by sme na obrazovke videli predmety a pomocou senzora by sme s nimi dokázali interagovať bez nutnosti použiť klasické periférie.

- *FigerList* a *Finger*

  Tieto triedy obsahujú metódy, prostredníctvom ktorých vieme zistiť počet prstov, vieme získavať jednotlivé prsty a zadeľovať im špecifické úlohy, vyberať prst, ktorý je najľavejší alebo najpravejši a podobne. Vďaka tomu vieme napríklad vypočítať priemernú pozíciu pre všetky prsty buď jednolivo pre každý prst alebo spoločnú priemernú pozíciu všetkých prstov.

  Taktiež vieme získať presný smer, ktorým prst ukazuje, jeho rýchlosť, či drží nejaký objekt, ktorý sa dá definovať ako nástroj, napríklad pero alebo štetec. Rovnako máme prístup k informáciám o tom, ako dlho je daný prst zaznamenávaný, čo je dôležité ak

[2] augmented reality

chceme k prstom pristupovať pomocou ich ID. Keďže sa môže stať, že senzor prestane na pár snímkov registrovať daný prst, ku ktorému sa my cez ID snažíme pristupovať, priradí mu nové ID, ktoré nemusí byť zhodné s tým predošlým a teda by sme dostali neplatný objekt. Vďaka metóde *finger.isValid()* dokážeme zistiť, či používame platné ID a v prípade, keď je dané ID neplatné, vieme si vďaka metóde *finger.timeVisible()* zistiť, ktorý prst bol zaznamenaný ako posledný a používať nové ID.

- *GestureList* a *Gesture*

  Trieda GestureList nám poskytuje zoznam všetkých gest, ktoré boli zaznamenané senzorom. Jedná sa o gestá, ktoré boli začaté alebo pokračujú v danej snímke. Vieme tak teda rozoznať presne dané gesto a môžeme ho pomocou jeho ID presne sledovať v snímkach nasledujúcich.

  Trieda Gesture potom slúži na samotné rozoznávanie gest. Medzi gestá, ktoré Leap Motion natívne podporuje, patrí napríklad mávnutie ruky ponad senzor[3], pichnutie prstom do vzduchu[4], ktoré sa môže interpretovať ako ťuknutie na obrazovku podobne ako pri dotykových obrazovkách, opísanie kruhu vo vzduchu[5] a ďalšie intuitívne a jednoduché pohyby ruky či prstov. Okrem rozpoznávania gest vieme touto triedou určiť, ktorá ruka dané gesto vykonáva, rovnako ako to, ktoré prsty sa na geste podieľajú.

  Je však dôležité si uvedomiť, že na to, aby boli gestá zaznamenávané, je potrebné tieto gestá jednotlivo povoliť pomocou metódy *controller.enableGesture(názov gesta)*. Inak žiadne gesto nebude rozpoznané ani ohlásené na spracovanie.

  Ďalej je treba vziať do úvahy dĺžku vykonávania gesta. Nakoľko máme možnosť získavať gestá aj pomocou ich ID, je možné, že takto vybrané gesto nebude v danej snímke a metóda *frame.gesture(ID)* nám vráti neplatný objekt čo bude viesť k chybe vykonávania programu. Je preto dôležité zamyslieť sa nad tým, či je výhodné vyberať gestá pomocou ID alebo iným spôsobom a ak už budeme využívať ID jednotlivých gest, tak potom musíme vykonať kontrolu tohoto ID a jeho platnosť v súvislosti s aktuálnym snímkom.

  Do viacnímkových gest patrí *CircleGesture* a *SwipeGesture*, ktoré majú tri fázy vykonávania a to start, update a stop.

  Do jednosnímkových alebo diskrétnych gest patrí *ScreenTapGesture* a *KeyTapGesture*. Tieto gestá majú vždy iba stop fázu.

5.2 MIDI knižnica pre Javu

Na prácu s protokolom MIDI existuje v Jave natívna knižnica, ktorá poskytuje triedy a metódy potrebné pre prácu s týmto štandardom.

Medzi základné piliere tejto knižnice patrí rozhranie *Synthesizer* (ďalej ako syntetizátor) a rozhranie *Sequencer* (ďalej ako sekvencér). S rozhraním sekvencér sme v rámci tejto práce nerobili a preto si ju uvedieme na záver a v krátkosti si ju priblížime. Z toho vyplýva, že sme pracovali výlučne s rozhraním syntetizátor.

[3] SwipeGesture
[4] ScreenTapGesture
[5] CircleGesture

Toto rozhranie slúži na generovanie zvuku v reálnom čase, tak ako sú noty hrané. V tomto je toto rozhranie rozdielne od rozhrania sekvencér, ktoré slúži na reprodukciu už zahraných nôt, ktoré boli týmto rozhraním zaznamenané.

Generovanie zvuku nastáva vtedy, keď kanál, ktorý prísluší k danému rozhraniu obdrží správu NoteOn. Po prijatí takejto správy je vygenerovaný zvuk podľa zadaných parametrov použitím nástroja z nahranej zvukovej banky[6]. Tento zvuk môže byť zvuk tradičného nástroja ako piáno, gitara alebo saxofón ale môže ísť aj o zvukový efekt či úplne vymyslený zvuk, ktorý sa inak ako počítačovo či elektronicky vygenerovať nedá.

Ďalej si bližšie popíšeme všetky použité a dôležité metódy, triedy a rozhrania tohto rozhrania:

- *rozhranie MidiChannel*

  Toto rozhranie slúži ako reprezentácia jedného kanála MIDI. Syntetizátor obsahuje kolekciu týchto kanálov, zvyčajne 16 ako to určuje štandard MIDI. Každý takýto kanál je nastavený na určitý nástroj a spracováva správy preň určené. To znamená, že ak kanálu 1 priradíme zvuk piána a použijeme metódu *NoteOn*, bude táto správa spracovaná kanálom 1, ktorý buď pomocou syntetizátora alebo sekvencéru prehraje zadanú notu. Noty, ktoré je možné prehrať, zadávame ako celočíselnú hodnotu správy NoteOn a to v rozmedzí 0 až 127, ako môžeme vidieť v tabuľke 5.1. Okrem čísla noty zadávame pri použití metódy NoteOn aj hodnotu velocity, ktorá je taktiež celočíselná hodnota a udáva akou silou by bola nota na reálnom nástroji zahraná. Táto hodnota má na výsledný zvuk veľký vplyv ako sme si už popísali v kapitole 4.

Oktáva	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
-1	0	1	2	3	4	5	6	7	8	9	10	11
0	12	13	14	15	16	17	18	19	20	21	22	23
1	24	25	26	27	28	29	30	31	32	33	34	35
2	36	37	38	39	40	41	42	43	44	45	46	47
3	48	49	50	51	52	53	54	55	56	57	58	59
4	60	61	62	63	64	65	66	67	68	69	70	71
5	72	73	74	75	76	77	78	79	80	81	82	83
6	84	85	86	87	88	89	90	91	92	93	94	95
7	96	97	98	99	100	101	102	103	104	105	106	107
8	108	109	110	111	112	113	114	115	116	117	118	119
9	120	121	122	123	124	125	126	127				

Tabuľka 5.1: Čísla MIDI nôt

Z ostatných metód tohto rozhrania je veľmi dôležitou metóda *NoteOff*, ktorá má na starosti vypnutie hrania danej noty. Táto metóda má ako jediný parameter čislo noty, ktorá sa má prestať hrať a funguje rovnako, ako keď hráč prestane hrať notu na normálnom nástroji.

Ďalšou takouto metódou je metóda *controlChange*, ktorá reaguje na zmenu ovládacích prvkov ako je napr. zmena hlasitosti kanála a podobne. Týchto ovládacích prvkov je veľmi veľa a preto si ich tu nebudeme spomínať. Táto metóda má dva parametre, kde prvý parameter je číslo prvku, ktorého hodnota sa zmenila a druhý parameter

[6] soundbank

je nová hodnota ovládacieho prvku, takže napr. zmena hlasitosti kanála sa vykoná zavolaním metódy controlChange(7,80), kde 7 je číslo ovládača hlasitosti a 80 je jeho nová hodnota.

Metóda *programChange* slúži na zmenu programu daného kanála. To znamená, že ak chcem zmeniť nástroj či zvuk, ktorý tento kanál má priradený, použijeme túto metódu. Jej jediným parametrom je číslo programu, ktorý chceme kanálu priradiť.

- *rozhranie Soundbank*

  Toto rozhranie obsahuje zoznam inštrumentov, ktoré môžu byť nahrané do syntetizátora. Je dôležité si uvedomiť rozdiel medzi Java Sound Soundbank a MIDI bank. MIDI povoľuje 16383 zvukových bánk, z ktorých každá môže obsahovať až 128 inštrumentov. Na druhej strane Soundbank môže obsahovať 16383x128 inštrumentov, pretože inštrumenty v Soundbank sú indexované pomocou čisla programu MIDI a čisla banky MIDI. Principiálne môžeme Soundbank považovať za kolekciu MIDI bánk.

- *trieda Instrument*

  Táto trieda slúži na simulovanie zvuku určitého hudobného nástroja alebo na dosiahnutie iného druhu zvuku či efektu. Inštrumenty sú zvyčajne uložené v kolekcíach, ktoré sa nazývajú zvukové banky. Skôr ako môžeme daný inštrument použiť, musíme ho nahrať do syntetizátora a priradiť jednému alebo viacerým kanálom. Toto dosiahneme použitím metódy programChange s číslom príslušného inštrumentu.

Sekvencér Ako sme si už povedali, na záver si v krátkosti popíšeme ešte rozhranie sekvencéru.

Sekvencér je softvérové alebo hardvérové zariadenie, ktoré slúži na prehrávanie sekvencií MIDI. Takáto sekvencia obsahuje zoznam dát MIDI s časovým razítkom tak, ako ich je možné čítať napríklad zo štandardného súboru MIDI.

Rozhranie Sequencer obsahuje metódy, ktoré podporujú nasledujúce základné MIDI správy:

- Získanie sekvencie zo súborov MIDI
- Pustenie a zastavenie prehrávania
- Úpravu dát pridávaním alebo mazaním MIDI správ
- Zmenu tempa prehrávania
- ...

Kapitola 6

Implementácia

AKO už bolo spomenuté vyššie, prvým väčším problémom, ktorý bolo potrebné vyriešiť, bola zmena výšky tónu. Nakoľko MIDI podporuje 127 nôt, ktoré udávajú výšku tónu a Leap Motion má rozsah -500 až +500 bodov na x-ovej osi, bolo potrebné zvoliť pomer, ktorý nám umožní využiť čo najväčšiu časť osi a zároveň nám umožní použiť čo najväčší rozsah nôt.

Ako prvá varianta bol použítý pomerový faktor 1:7, teda zmena o jednu notu na 7 jednotiek posunu na osi x. Tento pomer bol určený veľmi jednoducho a to tak, že sme plnému rozsahu osi priradili 127 nôt, čím sme sa dostali k číslu 7,874 bodu na notu. S predpokladom toho, že okrajové časti detekovanej plochy budú horšie snímané sme teda radšej zaokrúhlili nadol. Ukázalo sa , že tento predpoklad bol správny, a kvalita detekcie je v okrajových častiach detekovanej plochy horšia, a teda niekedy nebol prst detekovaný a nebolo možné zahrať plné spektrum nôt. Ako druhá alternatíva bol použitý pomer 1:5 čím sme síce dostali notový rozsah, ktorý prekračuje ten, ktorý je podporovaný štandardom MIDI, ale získali sme tak všetky noty, ktoré je možné zahrať a to bez výrazne zhoršenej miery detekcie.

Tento výber pomeru simuluje senzitivitu samotného thereminu. Rovnako ako pri theremine rozhoduje veľkosť induktancie a kapacitancie, tak pri tejto aplikácii rozhoduje pomer medzi posunom na osi a zmenou noty. Preto, ako už aj bolo spomínané, bolo potrebné nielen priradiť zmenu noty, zmene pozície na x-ovej osi ale aj zvoliť vhodnú senzitivitu samotnej aplikácie. Príliš vysoká senzitivita (napr. pomer 1:1 alebo 1:2) by nám zabránila jednak využiť celý osový rozsah a jednak by udržanie jednej noty bolo prakticky nemožné, nakoľko aj maličká zmena v pozícii by znamenala zmenu noty.

Ďalším problémom bolo samotné prehrávanie noty. Keďže sa prehranie noty spúšťa pomocou správy NoteOn a ukončuje pomocou správy NoteOff, je nutné aby medzi týmito dvoma správami bol aspoň minimálny časový rozdiel. Tu je dôležité sa zamyslieť nad tým, aký časový úsek ešte možno označiť za krátky, ale zároveň dostatočne dlhý na to, aby táto nota znela prirodzene. Skúšaním rôznych veľkostí pauzy medzi týmito správami som postupne došiel k názoru, že najvýhodnejšia pauza medzi správami je 200 až 300 milisekúnd. Počas testovania som používal hodnoty od 100 do 1000 milisekúnd, pri ktorých som sledoval nielen výsledný súzvuk, ale aj schopnosť aplikácie reagovať na zmenu pozície prsta a teda aj zmenu noty. Pri vysokých hodnotách bol reakčný čas aplikácie príliš nízky, pri nízkych hodnotách znel výsledný zvuk príliš neprirodzene, priam rušivo. Vybranie stredne krátkej doby sa ukázalo ako asi najschodnejšie riešenie, kedy aplikácia má veľmi dobrý reakčný čas a výsledný zvuk neruší pri používaní.

Týmto síce dosiahneme dobre znejúce tóny, ale stále bude dochádzať k prerušeniu medzi jednotlivými tónmi, ktoré je dané používaním správ NoteOff. Vo výsledku by sme teda

dostali síce aplikáciu, ktorá vydáva tóny na základe pozície ruky, ale tieto tóny by na seba nenadväzovali a boli by hrané samostatne s malou odmlkou medzi sebou. Tento problém sa dá jednoducho vyriešiť pomocou vláken[1]. Na začiatku aplikácie si pustíme dve vlákna, ktoré sa budú striedať v hraní tak, že v jednu chvíľu hrá len jedno vlákno a druhé čaká. Takto dostaneme plynulé hranie nôt bez odmlky.

Pri implementácií sme sa museli zamyslieť okrem iného aj nad tým, ako bude výsledná aplikácia vyzerať a hlavne ako sa bude ovládať. Mohlo by sa zdať, že je jednoduchšie využiť senzor a jeho možnosti. Treba však zobrať do úvahy to, že už ho využívame na hranie a teda použiť ho aj na ovládanie samotnej aplikácie by mohlo byť minimálne nepraktické. A to z toho dôvodu, že počas hrania by sa mohlo jednoducho stať, že nechcene zvolíme nejakú položku z menu alebo naopak, chceli by sme prejsť do menu a učiniť nejakú zmenu a zároveň by sme stále hrali.

Ďalším dôvodom prečo je výhodnejšie v tomto prípade použiť klasické ovládanie myšou je vo veľkosti okna aplikácie a vo veľkosti samotných tlačidiel menu. Pre bežného používateľa je jednoduchšie sa zorientovať vo fungovaní aplikácie, ktorá funguje obdobným spôsobom ako väčšina ostatných, ktoré používa a preto si aj rýchlejšie zvykne na novú aplikáciu.

Ak by sme chceli využiť na ovládanie senzor, museli by sme buď použiť veľké okno aplikácie s veľkými tlačidlami, alebo potom by sme museli zaviesť špeciálny spôsob ovládania, ktorý by nemusel každému vyhovovať a nie je zaručené, že by bol intuitívny a užívateľ by si naň musel dlho zvykať.

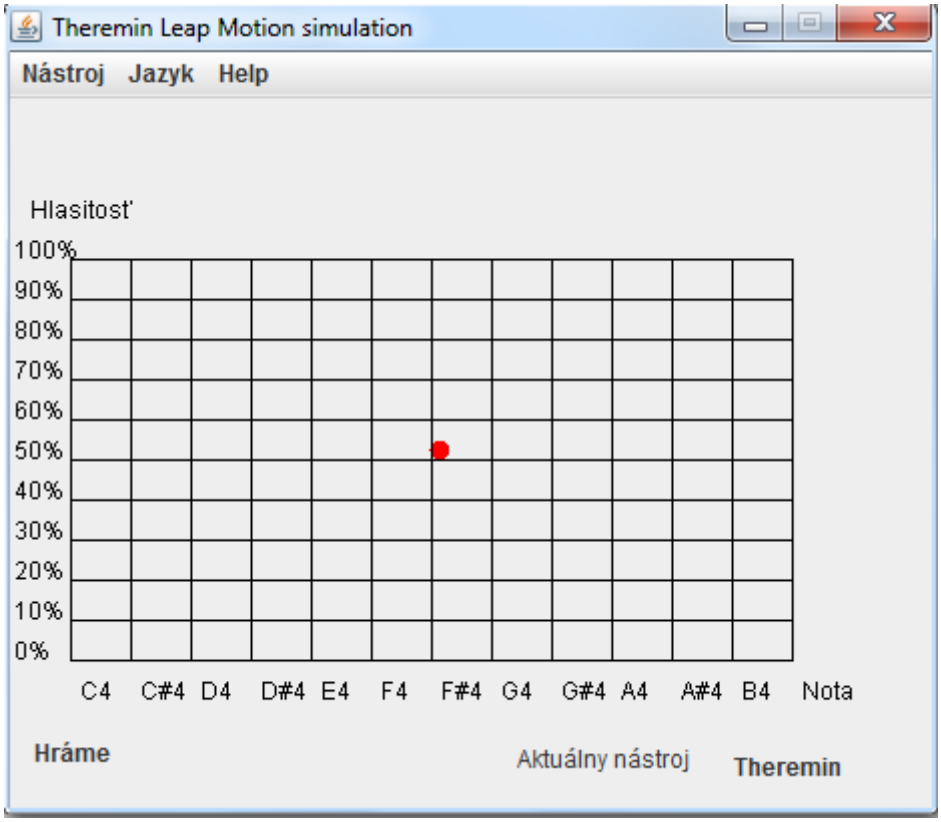
Napriek tomu, že pri hre na theremin sa používajú obe ruky, výsledná aplikácia využíva iba jednu ruku a prsty k nej priradené. Je to z toho dôvodu, aby sa obmedzili výmeny rúk pri detekovaní. Týmto rozumieme výpadok v registrácii jednej ruky, kedy by mohlo dôjsť k situácii, že sa funkcie rúk vymenia a ruka, ktorá dovtedy ovládala výšku tónu, by ovládala hlasitosť a naopak, ruka ktorá dovtedy ovládala hlasitosť by ovádala výšku tónu. Zároveň by to obmedzilo škálu hrateľných tónov ak by sme chceli zamedziť práve spomenutým výpadkom detekcie rúk.

Prioritne pre tieto dôvody je ovládanie aplikácie prenechané tradičnej periférii – myši. Na obrázku 6.1 vidíme aplikáciu s pripojeným senzorom, ktorý detekuje pohyb ruky a dochádza ku generovaniu zvuku. Ako si môžeme všimnúť, rozhranie aplikácie tvorí niekoľko prvkov, ktoré uľahčujú používateľovi orientáciu v tom, aký nástroj práve používa a či naň práve hraje. Taktiež má k dispozícii mriežku, ktorá ilustruje notu a oktávu, v ktorej je aktuálne hraná nota.

Na vizualizácu práve hranej noty slúži červený bod, ktorý sa v závislosti od pohybu po x-ovej a y-ovej osi pohybuje po tejto mriežke. Na x-ovej osi máme vynesené všetky noty, ktoré podporuje štandard MIDI spolu s ich oktávou. Táto oktáva sa mení v závislosti na polohe ruky nad senzorom, a to tak, že ak dosiahne hráč hornú alebo dolnú hranicu oktávy, nota C alebo B, presunie sa bod tak, aby korešpondoval s nasledujúcou notou, ktorá sa bude hrať. To znamená, že ak hráč aktuálne hrá notu C3 a presunie sa rukou doľava, čím zahrá nižšiu notu, oktáva nôt sa prepíše z 3 na 2 a bod sa presunie na pozíciu noty B2. Rovnakým spôsobom je potom riešená aj zmena oktávy a pozície bodu pri zmene na vyššiu oktávu. Teda pri hraní noty B3 sa po posunutí ruky doprava prepíše hodnota oktávy z 3 na 4 a bod sa presunie na notu C4. Takto nám stačí mať len relatívne malú mriežku, na ktorej sme schopní vizualizovať hranie všetkých nôt v plnom rozsahu -1 až 9 oktáv, tak ako to špecifikuje MIDI.

Na to, aby sme korektne vedeli zmeniť oktávu podľa pozície na x-ovej osi bolo potrebné

[1] Threads



Obr. 6.1: Screenshot aplikácie počas hrania

zohľadniť pomer, ktorý sme si predstavili skôr, a ktorý ovplyvňuje to, ako sa premietne detekovaný pohyb po tejto osi na zmenu noty. K tomu slúži nasledujúci vzorec:

$$(x * 360)/65) + 50$$

V tomto vzorci hodnota x predstavuje pozíciu na x-ovej osi, 360 je šírkový rozmer mriežky a 65 je počet bodov posunu, ktoré pripadajú na jednu oktávu. Posledné číslo 50 je výsledkom testovania aplikácie, kedy v dôsledku celočíselného delenia dochádzalo k prepisu oktáv už pri hraničných notách, resp. pozíciách. S takto upravnou pozíciou dochádza v drvivej väčšine prípadov k správnej zmene oktáv až pri prekročení hraničnej pozície a zmene v notách. Presun bodu z jedného okraja mriežky na druhý je potom už len otázka pričítania alebo odčítania šírky mriežky. Teda ak mám ľavú hranicu, tak pripočítam k pozícii 360, čím sa dostanem k pravému okraju a naopak, ak mám pravú hranicu mriežky, tak od pozície odpočítam 360 a dostanem bod na ľavý okraj mriežky. Týmto spôsobom vieme plynulo prechádzať z jednej oktávy do druhej.

Na premietnutie zmeny hlasitosti slúži os y. Na zmenu hlasitosti slúži pozícia na y-ovej osi senzora a je adekvátne premietnutá do pozície bodu na mriežke.

K dispozícii má používateľ rovnako možnosť pomocou roletového menu zmeniť nástroj a jazyk aplikácie. Pomocou prvého menu si môže vybrať jeden zo štyroch nástrojov. Momentálne sú to tieto nástroje:

- Piano
- Gitara
- Flauta
- Theremin

Z jazykov je potom k dispozícii slovenčina, čeština a angličtina. Pri zmene jazyka je táto zmena okamžite vykonaná a všetky nápisy aplikácie sú v danom jazyku.

Pomocou roletového menu Help si môže zobraziť používateľ nápovedu, kde je popísaný spôsob ako túto aplikáciu používať.

Aby nebolo nutné nastavovať požadovaný nástroj a jazyk pri každom spustení aplikácie, sú tieto voľby ukladané v konfiguračnom súbore *conf.properties*. Tento súbor je načítaný pri každom spustení programu a nastaví jazyk a nástroj podľa uložených hodnôt. Rovnako sa tam zapíšu všetky zmeny v týchto dvoch nastaveniach na základe výberu z roletového menu.

Kapitola 7

Záver

Cieľom tejto práce bolo vytvorenie aplikácie, ktorá bude simulovať hru na hudobný nástroj theremin s použitím protokolu MIDI. Na dosiahnutie tohto cieľa bolo nutné sa nielen zoznámiť so senzorom Leap Motion, ale aj bližšie spoznať samotný simulovaný hudobný nástroj theremin a najmä protokol MIDI, ktorý aplikácia využíva.

Narozdiel od štandardu MIDI, ktorý je veľmi dobre zdokumentovaný, je senzor Leap Motion náročnejší na pochopenie svojho základného princípu, na ktorom pracuje. Preto je aj v práci vysvetlený menej presne než protokol MIDI, ale to je spôsobené všeobecným nedostatkom informácií o presnom princípe jeho fungovania.

Hudobný nástroj theremin bližšie nepopisuje veľké množstvo literatúry a tak je človek, ktorý sa o tento prístroj zaujíma, rovnako ako pri Leap Motione odkázaný najmä na rôzne internetové zdroje, niekedy aj pochybnej kvality.

Napriek týmto úskaliam boli všetky body zadania splnené ako je vidieť nielen v teoretickej časti tejto práce, ale aj samotná aplikácia, ktorá využíva senzor Leap Motion a protokol MIDI na simuláciu hry na theremine je plne funkčná a poskytuje použiteľné rozhranie. Toto rozhranie je možné potom ďalej rozširovať.

Leap Motion má pred sebou podľa môjho skromného názoru veľmi dobré vyhliadky do budúcna, aj keď ho čaká ešte dlhá cesta za masovým využitím. To však, predpokladám, nájde nieen ako periféria PC, ale aj v robotike, medicíne či priemysle. Vďaka tomu, že ešte stále ide o novú technológiu, aj keď prvé testovacie prototypy Leap Motionu boli uvoľnené pre vývojárov už okolo roku 2012, je vývoj tohto zariadenia stále veľmi zaujímavý a denne vznikajú rôzne aplikácie, ktoré sú k dispozícii cez ich obchod Airspace[1] alebo cez aplikáciu rovnakého mena.

Ako som už povedal, očakávam, že Leap Motion si časom nájde cestu do mnohých odvetví a predovšetkým rôznych zariadení; niektorí výrobcovia notebookov už ohlásili implementovanie tohto senzora do svojich prémiových zariadení. Preto by som do budúcna videl ako príležitosť na rozšírenie tejto aplikácie zvýšiť počet nástrojov, na ktoré by sa dalo hrať. Nielen theremin, ale aj gitara či bicie nástroje by sa dali simulovať za použitia protokolu MIDI a Leap Motionu. Bolo by výhodné a zaujímave pre užívateľov keby mali možnosť zapojiť nielen prst, ale aj obe ruky zároveň, kde by každá ruka mala svoj presne daný účel ako napríklad držanie akordu ľavou rukou a hranie na struny pravou rukou.

Ďalším pravdepodobne vítaným rozšírením by bola možnosť ukladať hrané skladby vo formáte MIDI za účelom neskoršieho opätovného prehrávania či editácie..

[1] https://airspace.leapmotion.com/

Literatúra

[1] admin: Kinect Hacking 103: Looking at Kinect IR Patterns. 2010.
URL http://www.futurepicture.org/?p=116

[2] BAARS, M.: The theremin - How it works [online]. 2004-12-29 [cit. 2014-04-10].
URL http://www.thereminvox.com/article/view/134/1/2.html

[3] Edwin: Is it possible to get raw point cloud data? [online]. 2014-02-12 [cit. 2014-04-30].
URL https://support.leapmotion.com/entries/40337273-Is-it-possible-to-get-raw-point-cloud-data-

[4] Evgeny SPEKTOR, D. R., Zafrir MOR: Integrated processor for 3D mapping. 2010, patent application number: 20100007717.
URL http://www.faqs.org/patents/app/20100007717

[5] FREEDMAN, B.; SHPUNT, A.; MACHLINE, M.; aj.: Depth mapping using projected patterns. 2010, patent application number: 20100118123.
URL http://www.faqs.org/patents/app/20100118123

[6] LAURENTINI, A.: The Visual Hull Concept for Silhouette-Based Image Understanding. *IEEE Trans. Pattern Anal. Mach. Intell.*, ročník 16, č. 2, 1994: s. 150–162, ISSN 0162-8828.
URL http://dx.doi.org/10.1109/34.273735

[7] Leap Motion, I.: Leap Motion product [online]. 2014, [cit. 2014-04-30].
URL https://www.leapmotion.com/product

[8] LIMER, E.: Leap Motion Teardown: Magic Made Simply [online]. 2013-06-28 [cit. 2014-04-30].
URL http://www.gizmodo.co.uk/2013/06/leap-motion-teardown-magic-made-simply

[9] MANNING, P.: *Electronic and Computer Music*. Oxford University Press, Oxford, 1985, rev. 2004, ISBN 0-19-517085-7, [cit. 2013-12-15].

[10] Microsoft, C.: Kinect for Windows Sensor Components and Specifications [online]. [cit. 2014-04-20].
URL http://msdn.microsoft.com/en-us/library/jj131033.aspx

[11] NACHBAUR, F.: On theremin sensitivity [online]. 1997-09, rev. 2000-10 [cit. 2014-04-11].
URL http://www.dogstar.dantimax.dk/theremin/thersens.htm

[12] NGUYEN, C.: Working principle of Leap Motion device [online]. 2013-01-06 [cit. 2014-04-30].
URL https://sites.google.com/site/chuongvnguyen/blog/workingprincipleofleapmotiondevice

[13] ORLAND, K.: Microsoft Announces Windows Kinect SDK For Spring Release [online]. 2011, [cit. 2014-04-20].
URL http://www.gamasutra.com/view/news/33136/Microsoft_Announces_Windows_Kinect_SDK_For_Spring_Release.php

[14] Robert GUÉRIN: *Velká kniha MIDI*. Computer Press, Brno, 2004, ISBN 80-7226-985-2, [cit. 2013-12-10].

[15] SHPUNT, A.: Depth mapping using multi-beam illumination. 2010, patent application number: 20100020078.
URL http://www.faqs.org/patents/app/20100020078

[16] SPIEGELMOCK, M.: *Leap Motion Development Essentials*. Packt Publishing, Birmingham, 2013, ISBN 978-1-84969-772-9, [cit. 2014-04-30].

[17] Wikipedia: Motion detection [online]. 2013, [cit. 2013-12-15].
URL http://en.wikipedia.org/wiki/Motion_detection

Dodatok A

Obsah CD

- Text práce vo formáte PDF a zdrojové súbory pre LaTeX
- Zdrojový kód programu v jazyku Java
- Spustiteľný .jar archív