TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Možnosti tance robota Pepper

Bakalářská práce

Studijní program: B0613A140005 Informační technologie
Studijní obor: Inteligentní systémy

Autor práce: **Mariya Kanash**
Vedoucí práce: Ing. Miroslav Holada, Ph.D.
Ústav informačních technologií a elektroniky

Liberec 2022

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Zadání bakalářské práce

Možnosti tance robota Pepper

Jméno a příjmení: **Mariya Kanash**
Osobní číslo: M19000021
Studijní program: B0613A140005 Informační technologie
Specializace: Inteligentní systémy
Zadávající katedra: Ústav informačních technologií a elektroniky
Akademický rok: **2021/2022**

Zásady pro vypracování:

1. Seznamte se s humanoidním robotem Pepper na pracovišti školitele. Proveďte rešerši stávajícího stavu tanečních realizací tohoto robota.
2. Naprogramujte software, který umožní, aby robot předvedl různé taneční kreace včetně hudebního doprovodu.
3. Zaměřte se na synchronizaci pohybu a hudby. Navrhněte vhodný algorytmus detektce rytmu a ověřte jeho spolehlivost a robustnost.
4. Při návrhu dále zohledněte možnost snadné modifikovatelnosti ukázkové úlohy. Ověřte funkcionalitu na několika tanečních stylech.
5. V závěru diskutujte výhody a nevýhody vlastní realizace a zhodnoďte celkově taneční možnosti robotu Pepper.

Rozsah grafických prací:	dle potřeby dokumentace
Rozsah pracovní zprávy:	30-40 stran
Forma zpracování práce:	tištěná/elektronická
Jazyk práce:	Čeština

Seznam odborné literatury:

[1] VANER, Pavel : M15000123. Spolupráce robotů NAO: NAO Robots collaboration. Liberec: Technická univerzita v Liberci, 2018. Bakalářské práce. Technická univerzita v Liberci.

[2] EICHLER, Miroslav : M15000089. Využití dostupných senzorů robota Nao pro detekci objektů a mapování okolí: Use available Nao robots' sensors to detect objects and map of its surrounding. Liberec: Technická univerzita v Liberci, 2018. Bakalářské práce. Technická univerzita v Liberci.

[3] https://www.softbankrobotics.com

[4] NENCHEV, Dragomir N. a Atsushi KONNO. Humanoid Robots. 1. Berlin, SRN: Elsevier – Health Sciences Division, 2016. ISBN 9780128045602.

Vedoucí práce:	Ing. Miroslav Holada, Ph.D. Ústav informačních technologií a elektroniky
Datum zadání práce:	12. října 2021
Předpokládaný termín odevzdání:	16. května 2022

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
děkan

prof. Ing. Ondřej Novák, CSc.
vedoucí ústavu

V Liberci dne 19. října 2021

Prohlášení

Prohlašuji, že svou bakalářskou práci jsem vypracovala samostatně jako původní dílo s použitím uvedené literatury a na základě konzultací s vedoucím mé bakalářské práce a konzultantem.

Jsem si vědoma toho, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu Technické univerzity v Liberci.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědoma povinnosti informovat o této skutečnosti Technickou univerzitu v Liberci; v tomto případě má Technická univerzita v Liberci právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Současně čestně prohlašuji, že text elektronické podoby práce vložený do IS/STAG se shoduje s textem tištěné podoby práce.

Beru na vědomí, že má bakalářská práce bude zveřejněna Technickou univerzitou v Liberci v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů.

Jsem si vědoma následků, které podle zákona o vysokých školách mohou vyplývat z porušení tohoto prohlášení.

13. května 2022 Mariya Kanash

Poděkování

Ráda bych poděkovala Ing. Miroslavu Holadovi Ph.D. za cenné rady a připomínky ohledně této bakalářské práce, a zároveň bych ráda poděkovala všem, kteří přispěli ke vzniku tohoto díla.

Možnosti tance robota Pepper

Abstrakt

Tato bakalářská práce se zabývá především výzkumem možností pohybu robota Pepper, tzn. jaké možnosti pohybu nabízí jednotlivé části robota. Výstupem této práce je naanimované taneční vystoupení robota do rytmu hudby, kterou sám přehrává. V rámci práce byla vytvořena aplikace, která běží na samotném robotovi, jenž umožňuje jednoduché ovládání tanečního vystoupení robota. Tato aplikace umožňuje vytvoření různých tanečních sekvencí pro ukázku tanečních možností robota. Všechny pohyby jsou v synchronizaci s přehrávanou hudbou.

Klíčová slova: Robot, animace, BPM, Android Studio, Animation Editor

Abstract

The main goal of this work is to research the capabilities of the robot Pepper, how good is it at moving and controlling all the different parts of its body, meaning its head, arms, torso etc. The result of this work is the animated performance of the robot playing music and dancing to its rythm. An application running on the robot itself was created to make it easier to control the robots performance. This application supports the creation of different dance sequences to showcase the robots capabilities. All the dance movements are in synchronization with the music playing.

Keywords: Robot, animation, BPM, Android Studio, Animation Editor

Obsah

Seznam obrázků

Seznam zkratek

TUL	Technická univerzita v Liberci
FM	Fakulta mechatroniky, informatiky a mezioborových studií Technické univerzity v Liberci
FFT	Fast Fourier transform, Rychlá Fourierova transformace
IDFT	Inverse Discrete Fourier Transform, Zpětná diskrétní Fourierova transformace
OS	Operační systém
SDK	Software Development Kit, Sada vývojových nástrojů
AI	Artificial Intelligence, Umělá inteligence
BPM	Beats per minute, Údery za minutu
XML	Extensible Markup Language, Rozšiřitelný značkovací jazyk
DWT	Discrete wavelet transform, Diskrétní vlnková transformace
JSON	JavaScript Object Notation, JavaScriptový objektový zápis
UI	User interface, Uživatelské rozhraní
ID	Identifikátor
FPS	Frames per second, Snímková frekvence

Úvod

Robotika je dnes již nedílnou součástí lidského života, roboty využíváme například ve výrobě, ve zdravotnictví, v zábavním průmyslu a jinde. Ale pro interakci s lidmi v obchodě nebo v nemocnici se roboti zatím příliš nehodí. Tato práce se ale snaží zlepšit možnosti interakce robota s lidmi. K tomu je využit robot Pepper, což je humanoidní robot, který byl vytvořen za účelem právě interakce s lidmi. Slovo humanoidní je odvozené od anglického slova human – člověk.

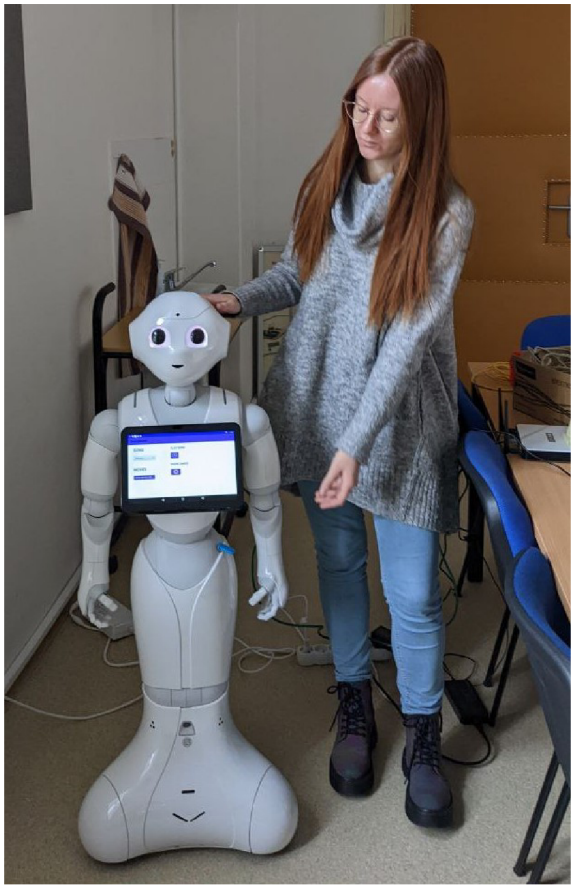
Robot Pepper již má zkušenosti s prací v obchodě mezi lidmi. Např. ve Skotsku robot Pepper pracoval jako prodavač, ale odpracoval pouze jeden týden, protože kvůli šumu v okolí nebyl schopný reagovat na otázky zákazníků [1]. Zbývá tak ještě hodně práce a rozvoje v oboru robotiky na to, aby humanoidní roboti byli schopní interagovat s lidmi na dostatečné úrovni. Tato práce přispívá v tomto rozvoji snahou rozchodit robota Pepper, výzkumem jeho tanečních schopností, a následným využitím těchto schopností v praxi za účelem zlepšení interakce mezi člověkem a robotem.

Tato práce se soustřeďuje na tvorbu tanečních animací, protože tanec je jedním ze způsobu interakce robota s lidmi. Tanec je totiž středem zájmu v oblasti sociologie a umění. Tanec je možné definovat jako několik navazujících pohybů různými částmi těla robota. Jednotlivé části robota je možné otočit v různých směrech a s různou rychlostí. Spojením pohybů je možné sledovat, jak robot přepíná mezi jednotlivými pohyby, a jak toto spojení ovlivňuje vedlejší pohyby. Nutnou součástí tance je ale hudební doprovod. Z řady navazujících pohybů doprovázených hudbou vzniká skutečný tanec. Do takové aktivity se mohou zapojit i lidé. Zkoumáním robota a člověka při společném tanci je možné přijít na poznatky, které mohou přispět ke zlepšení komunikace mezi člověkem a robotem.

Tato práce je zaměřená též na zpracování hudby ve smyslu analýzy tempa jako parametru skladby jelikož je žádoucí, aby robot tancoval do rytmu hudby. Díky této analýze by měl být robot schopný lépe synchronizovat taneční pohyby se hudbou. To robotovi napomůže předvádět tanec podobně tomu, jak to předvádí člověk.

1 Popis robota Pepper

Při programování aplikace byl použit robot Pepper, což je robot od společnosti SoftBank Robotics, který je zaměřený na komunikaci s lidmi (obr. 1.1). Pro tyto účely je robot vybavený tabletem, kamerami, které umožňují detekci lidí a rozpoznání jejich emocí [2]. Robot umí mluvit a rozpoznávat lidskou řeč. Pro komunikaci podporuje různé lidské jazyky. Je vysoký 121 cm, má příjemně vypadající obličej a hlas, má dvě ruce a dokáže sledovat polohu člověka. Díky omezení pohybů v blízkosti překážek, včetně člověka, se pohybuje s ohledem na detekovaného člověka a tím pádem mu není schopný ublížit.



Obrázek 1.1: Humanoidní robot Pepper a člověk

1.1 Popis vybavení robota Pepper

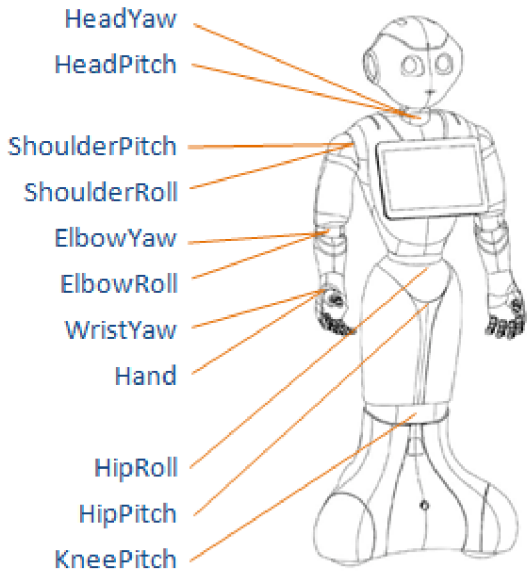
Robot Pepper má následující parametry: výška je 121 cm, šířka je 48 cm, délka je 42,5 cm. Rozpětí paží je 119,7 cm, při upažení má robot rozpětí 64,8 cm, při vzpažení dosahuje robot výšky 135,5 cm. Robot má hlavu, hruď a spodní část. Tyto části obsahují následující prvky:

- hlava obsahuje:
  - reproduktory s obou stran hlavy – ve stejném místě, kde se nachází uši u člověka
  - kamery
    * 2D kamery v čele a puse
    * dvě 2D kamery v očích, které dohromady tvoří 3D obrázek
    * 3D kamera je udělána z stereo kamer za očima
  - 4 mikrofony navrchu hlavy
  - 2 klouby v místě krku
  - hmatové senzory navrchu hlavy
- hruď obsahuje:
  - tablet s OS Android
  - 2 ruce, každá ruka má:
    * 6 kloubů v podobných místech, jako má člověk, v rameni, v loktu a v zápěstí
    * hmatový senzor nahoře na zápěstí
  - tlačítko na zapínání/vypínání robota se nachází pod tabletem
- spodní část není rozdělena na nohy, ale obsahuje 3 klouby: v místě spojení s hrudí a uprostřed spodní části, v místě kolen

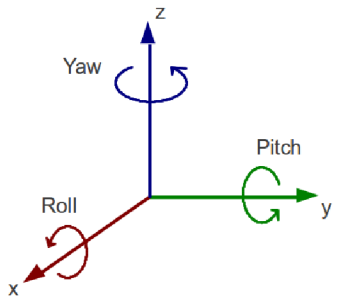
1.2 Možnosti pohybování robota Pepper

Pohybování robota je umožněno klouby, díky kterým je robot schopen pohybu v podobných mezích jako člověk. Na obr. 1.2 jsou ukázané polohy kloubů. Klouby jsou rozdělené podle směru pohybu – Yaw, Pitch a Roll (obr. 1.3). V standardní poloze Roll je otočením kolem osy X, Pitch kolem osy Y a Yaw kolem osy Z. Umístění os pohybu vůči robotovi je na obr. 1.4. Stupně otáčení jednotlivých kloubů, pokud je základní poloha robota jako na obr. 1.1:

- hlava je schopná se otáčet doleva či doprava o 120° (HeadYaw), dopředu o 25,5°, dozadu o 40,5° (HeadPitch)

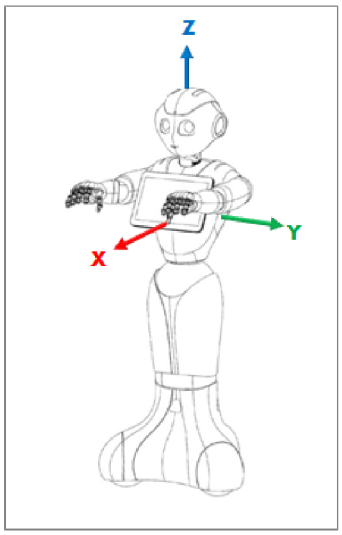


Obrázek 1.2: Klouby robota Pepper [3]



Obrázek 1.3: Osy otáčení a popis příslušných kloubů [3]

- hruď se otáčí doleva či doprava o 29,5° (HipRoll), dopředu a dozadu o 59,5° (HipPitch)
- ruce:
  - v místě spojení s hrudí je robot schopný zvednout ruku do úhlu 89,5°, k tělu je schopný přiblížit ruku o 0,5° (ShoulderRoll)
  - v místě loktu může ohnout ruku až o 89,5°, v opačném směru pohne rukou o 0,5° (ElbowRoll)
  - kolem své osy: v oblasti loktu otočí ruku až o 119,5° od těla a k tělu (ElbowYaw), v zápěstí o 104,5° od těla k tělu (WirstYaw)
- spodní část:
  - uprostřed (v oblasti kolen) je robot schopný se pohnout o 29,5° (KneePitch)

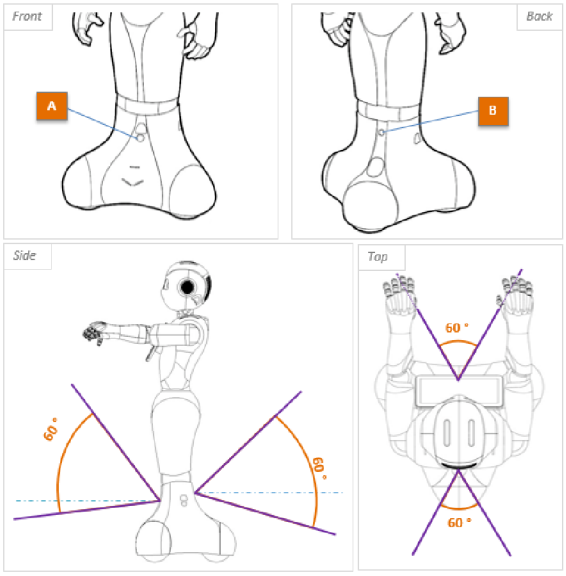


Obrázek 1.4: Osy otáčení vzhledem k robotovi [4]

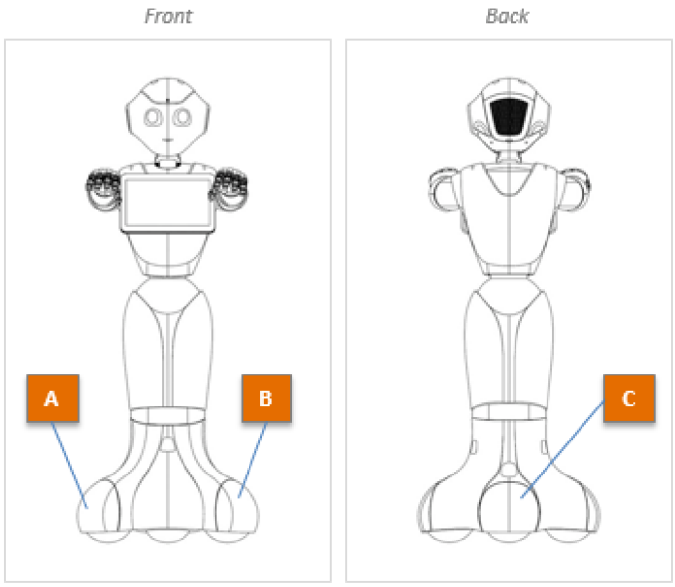
Pro pohybování po místnosti je robot ve spodní části vybavený třemi gumovými rotujícími kolečky ve formě koulí.

Při provádění pohybů je robot Pepper schopný pomocí senzorů odhadnout vzdálenost mezi ním a překážkami v jeho okolí. Senzor detekuje překážku na vzdálenosti 0,3 - 5 m a v „kuželu“ 60°. Senzory se nachází vepředu a vzadu robota v jeho spodní části (obr. 1.5). V případě detekce překážky se pohyb může provést jenom částečně nebo se neprovede vůbec.

Pro bezpečný pohyb po místnosti robot má 3 nárazníky (bumpers, obr. 1.6).



Obrázek 1.5: Senzory robota Pepper [5]



Obrázek 1.6: Nárazníky robota Pepper [6]

1.3 Programovací prostředky robota Pepper

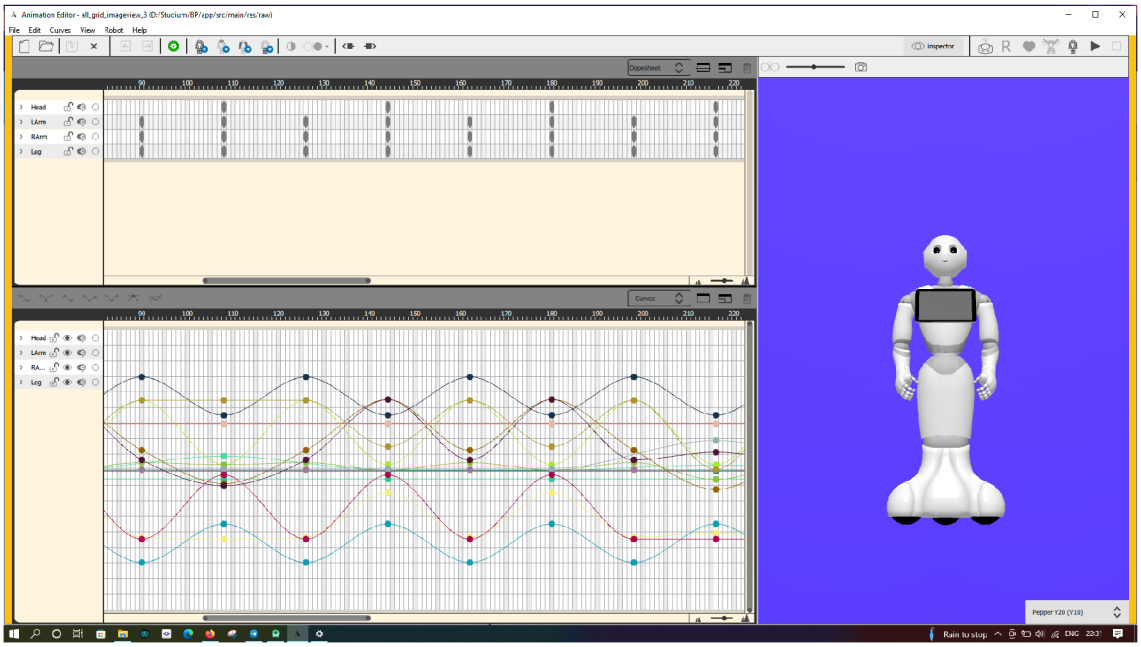
Robot se dá programovat několika způsoby. Jedním z nich je tablet na hrudi robota s operačním systémem Android. Na tabletu se dá pustit Android aplikace pro ovládání robota. Na vytvoření aplikace bylo použito vývojové prostředí Android Studio na bázi jiného vývojového prostředí – IntelliJ IDEA. V Android studiu se používá programovací jazyk Java nebo Kotlin. Pro práci s robotem Pepper se využívá Pepper SDK, což je plugin, který obsahuje grafické nástroje a Java knihovnu - QiSDK. Jiný způsob – je ovládán přes Python.

Knihovna QiSDK implementuje metody pro komunikaci s robotem [7]. QiSDK registruje či odregistruje nějakou aktivitu, protože v Android Studiu se pracuje s aktivitami – třídou typu Activity. Ukázka metod v Javě na registraci/odregistraci aktivity (**this** – je daná aktivita):

```
// Unregister the RobotLifecycleCallbacks for this Activity.
QiSDK.unregister(this, this);
// Register the RobotLifecycleCallbacks for this Activity.
QiSDK.register(this, this);
```

Pustit aplikaci je možné nejenom na reálném robotu, ale i v emulaci. Je možné pustit emulaci jak tabletu, tak i robota, tzv. Robot View (obr. 4.10). Podrobnější vysvětlení v sekci 4.4.

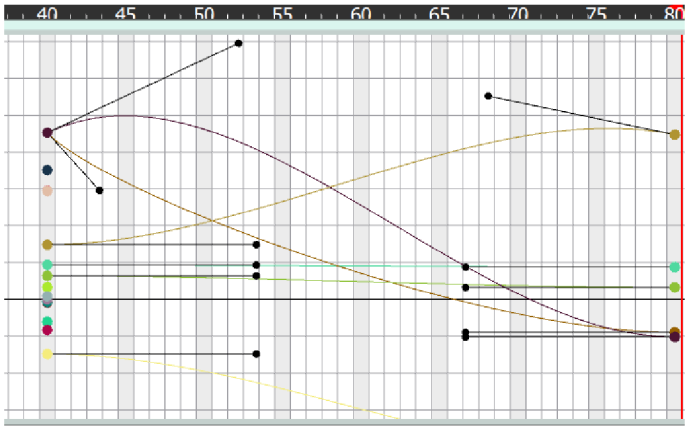
1.3.1 Prostředí na tvorbu animace pro robota Pepper Animation Editor



Obrázek 1.7: Hlavní okno Animation Editor

Animation Editor je nástroj v Android Studiu, který je součástí knihovny QiSDK. Tento nástroj obsahuje časovou osu, která je rozdělena na snímky. Na určitý snímek, kde se vyskytuje úder (beat) zjištěný podle algoritmu, se může přidávat požadovaný pohyb [8]. V Animation Editoru vlevo jsou skupiny kloubů (Head, LArm, RArm, Leg), ze kterých se vybírají potřebné klouby a nastavují se na potřebný snímek. Na časové ose se pohyb značí 2 způsoby: pomocí kroužků (key) (horní část obr. 1.7) anebo teček, spojených křivkami (spodní část obr. 1.7). V key režimu není vidět změna parametrů nastavení kloubů, jsou tam jenom klouby, které se používají v tomto frame (snímku). Křivky spojují pohyby mezi sebou a vyskytují se v několika režimech, bezier, bezier automatic, linear, bezier smooth a bezier symmetrical. To bude mít vliv na rychlost změny parametrů kloubů, tj. na provedení robotem pohybu kloubem. Kromě bezier automatic při výběru režimu je možné nastavovat sklon křivky pomocí tečen (tangents)(obr. 1.8). Pomocí tečen je možné upravovat sklon křivky libovolným způsobem. Interpolace je buď bezier, nebo lineární, to znamená přenos nebo modelování křivky pohybu pomocí bezier křivky, která se používá, aby pohyb byl plynulý. Lineární interpolace používá lineární „proklady“, tj. méně plynulé křivky, spíše přímky, ze kterých se skládají větší přímky. Na pravé straně Animation Editoru se nachází Robot 3D View (vpravo obr. 1.7), kde je emulace robota. Tam je možné přehrávat animaci, nastavovat požadovánou polohu končetin robota, startovní pozici robota, nebo měnit model robota (animační skript je možné přehrát na jiných robotech, např. Romeo či Nao od společnosti SoftBank Robotics).

Hotová animace se ukládá do souboru ve formátu XML (rozšiřitelný značkovací



Obrázek 1.8: Křivky s zobrazenými tečnami (černé čáry)

jazyk) [9]. XML je podobný formátu HTML, ale bez předdefinovaných tagů. Každý tag se definuje podle přání autora. Tag začíná „<“ a končí „/>“. V rámci těchto závorek se zapisuji potřebné parametry přes rovnítko. V případě souboru s animací existují následující tagy:

- Animation – tag, kde je informace o celé animaci. Obsahuje parametr „editor“, tj. typ editoru, a jeho verze - „typeVersion“ a „fps“.
- ActuatorCurve – obsahuje informaci o kloubech. Uvnitř toho tagu se zobrazují tagy s názvem „Key“, tj. jednotlivé pohyby. Obsahuje parametry „fps“ a název kloubu – „actuator“.
- Key – informace o jednom pohybu, má parametry „value“, tj. parametr polohy kloubu, a „frame“, tj. čas výskytu pohybu. Také obsahuje tag „Tangent“.
- Tangent – je uvnitř „Key“, obsahuje parametry „side“ orientace left/right – na levou/pravou stranu, „abscissaParam“ poloha tečny na ose x, ordinateParam poloha tečny na ose y, „editor:interpType“ – typ interpolace, typ práce s křivkami – typ interpolace (bezier, linear atd.).

Příklad tagu s parametry tečny v XML souboru:

```
<Tangent side="right" abscissaParam="13.3333333"
ordinateParam="0"
editor:interpType="bezier_auto"/>
```

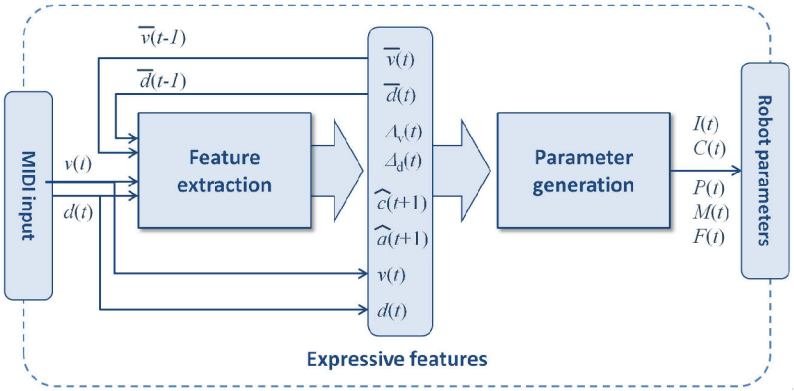
2 Využití tanečních možností robota Pepper v praxi

Vývojáři, kteří programují nebo nějakým jiným způsobem využívají robota Pepper, připravují hotové šablony animace a využívají je, když se například na robota obrátí nějaký člověk, mluví na něj a robot pak odpovídá, anebo něco ukazuje a zároveň přehrává potřebnou animaci. Na Youtube při hledání tančícího robota Pepper se většinou zobrazují videa, kde se lidé chtěli pobavit, jenom vyzkoušet robota a podívat se na jeho schopnosti, anebo videa z výstav, nákupních center, obchodů, kde robot prezentuje společnost, produkt, nebo je určen pro zábavu.

Zajímavým využitím tanečních schopností robota Pepper bylo vystoupení 20 robotů Pepper na stadiónu společně s roboty od Boston Dynamics [10]. Roboti hráli roli roztleskávaček a pohyby jednotlivých robotů byly mezi sebou synchronizovány.

Robot se taky používal v experimentu, kde robota Pepper umístili v nákupním centru [11]. Tím se zkoumala interakce lidí s robotem, jeho popularita (jak moc pozornosti robot získá) a reakce lidí na robota (obr. 2.1). Autoři měli jako cíl nasbírat informace od uživatelů pro vývoj aplikací zaměřených na komunikaci s robotem. Bylo zjištěno, že lidé začínali tančit s robotem, tleskali mu po provedení tance anebo zopakovali po robotovi jeho pohyby. Autoři došli k závěru, že robot by měl poskytovat zábavu pro děti a užitečné informace například o umístění obchodů v nákupním centru.

Obrázek 2.1: Komunikace robota a člověka v nákupním centru [12]



Obrázek 2.2: „AI“ systém spolupráce robota a člověka [14]

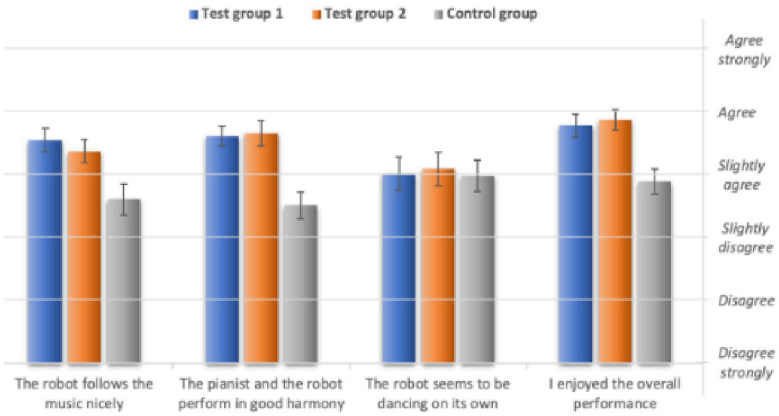
V roce 2020 byl zkoumaný vliv na uživatele komunikujícího s robotem a způsob této komunikace [13]. Uživatelé používali aplikaci jenom na tabletu, nebo na tabletu s robotem, kde robot předváděl různé animace. Aplikace byla kvízem, kde účastníci experimentu zodpovídali na otázky a pak hodnotili své pocity z použití aplikace dvěma zmíněnými způsoby. Robota s animací účastníci hodnotili jak nejzajímavějšího a nejvíc je bavícího, ale méně pragmatického z hlediska použití aplikace. Také se ukázalo, že robot svou animací rušil účastníky a odpoutal jejich pozornost od kvízu. Otázkou synchronizace hudby při vytváření animace se nikdo nezabýval seriózně: buď animaci a hudbu synchronizovali ručně, nebo se přehrávaly příliš krátké animace, které nepotřebovaly hudební doprovod.

V článku [14] se zkoušela komunikace člověka a robota přes umělou inteligenci (AI). Člověk hrál na piano kompozici, která měla určitou tonalitu, rychlost apod. AI následně určila parametry skladby a poskytla informaci robotovi. Robot na základě obdržené informace měl předvést taneční pohyby, které by byly synchronizovány s hudbou. Na generování tance byla použitá sada základních pohybů, které využívají tanečníci baletu. Ze základních pohybů se vytvářely dílčí pohyby.

Robot obdržel následující parametry: intensity *I(t)* – intenzita, complexity *C(t)* – složitost, pattern *P(t)* – seznam ze sedmi pohybových vzorů, uspořádaných podle rostoucí intenzity, muting *M(t)* – tonalita hudby, mollová nebo durová a fill *F(t)* nebo „doplnění“ dalšího pohybu k již vytvořeným. To se provádělo v systému ve fázi Parameter generation (obr. 2.2). Výše uvedené parametry se generovaly pomocí sady proměnných skladby, jako: rychlost skladby $v(t)$, hustota rytmu $d(t)$, průměrná rychlost v předchozím taktu $\bar{v}(t)$, změna rychlosti mezi předchozím a následujícím taktem $\Delta v(t)$, a předpověď intenzity hudby v následujícím taktu $\hat{c}(t{+}1)$. Tento krok se uskutečnil ve fázi Feature extraction (obr. 2.2). Celkový „AI“ systém je rekurentní, tzn., že tento systém na vstupu požívá i výstup z předchozího kroku, aby zlepšil odhad a výpočet parametrů skladby.

Na základě přidaných parametrů a náhodné veličiny se spočítá pravděpodobnost volby pohybu ze seznamu.

Autoři kladou důraz na přípravu pohybů tak, aby byly vzájemně vyměnitelné, a při jakékoliv jejich kombinaci vypadal vygenerovaný tanec vhodně, přirozeně, a aby se pohyby k sobě hodily a měnily se plynule. Také bylo důležité, aby jednotlivé



Obrázek 2.3: Dotazník ohledně spolupráce robota a člověka [14]

pohyby byly od sebe odlišitelné, tzn., aby se při změně tonality skladby generovaly jiné pohyby.

150 lidí bylo rozděleno do tří skupin a bylo jim ukázáno několik videí s robotem a hudebníkem. Následně pak lidé měli vyplnit dotazník. Dotazník obsahoval 4 skupiny otázek (obr. 2.3): 1.„The robot follows the music nicely“ – „Robot sleduje hudbu dobře“, 2. „The pianist and the robot perform in good harmony“ – „Klavírista a robot vystupovali v souladu mezi sebou“, 3.„The robot seems to be dancing on its own“ – „Zdá se, že robot tančí sám o sobě“, 4.„I enjoyed the overall performance“ – „Vystoupení se mi líbilo“. První dvě testovací skupiny kromě poslední skoro se všemi tvrzeními ohledně spolupráce robota a hudebníka, odpovídali stejně, např. na první otázku odpověděli, že s těmito tvrzeními souhlasí. Jelikož skupiny koukaly na videa, která se lišila jenom hudebníkem který hrál na piano, znamenalo to, že navržený systém funguje nezávislé na pianistovi. Lidé vnímali dobré spolupráce robota a člověka. Všechny 3 skupiny zodpověděly stejně na třetí tvrzení, a to že spíše souhlasí.

Vytvořený AI systém byl vyzkoušen také na vystoupení před 250 lidmi. Ukázalo se, že publikum ocenilo vystoupení pozitivně a reagovalo na to, jako na něco neobvyklého. Ve videu [15] klavírista občas měnil tempo melodie, a bylo vidět, že robot měnil pohyby od pomalejších (baletních), kde melodie je pomalá, do rychlejších a intenzivnějších, kde je melodie rychlejší. Během vystoupení robot nakláněl hlavu podle rytmu v hudbě.

Humanoidní robot se používal v terapii, například v práci, která se zabývala terapií pacientů s demencí [16]. Robot komunikuje s pacienty během aktivních cvičení ve formě tance. Pohyby jsou adaptované tak, aby bylo možné provádět stejné pohyby i pacientům na invalidním vozíku a aby pacienti s demencí byli schopni je provést bez potíží. Intenzita provedení jednotlivých pohybů se upravuje pomocí monitoringu srdečního tepu. Je stanovená horní hranice tepu pacientů. Pokud dojde k jejímu překročení, robot musí snížit intenzitu prováděných pohybů. Robot nejdřív demonstruje pohyb, který pak pacienti mají zopakovat. Kromě demonstrace pohybů, robot Pepper také komunikuje s uživatelem pomocí řeči, např. robot říká „nezpomaluj se“

nebo „hýbej pravou rukou víc“ apod.

Na měření tepu se využívá snímač, který komunikuje s počítačem Raspberry Pi Zero W. Počítač se připojuje k robotovi přes wi-fi a odesílá data. Cílem práce bylo vyzkoušet různé pohyby podle intenzity a zjistit pro každý z cviků průměrné zvýšení tepů za vteřinu. Nalezenou informaci měl pak robot Pepper použít pro kontrolu provedení tréninku, tj. zvolit další pohyb, buď více nebo méně intenzivní.

V jiném článku [17] práce také probíhala s lidmi s demencí, ale na pacientech byly vyzkoušeny 2 programy „trénování“ pomocí tabletu a pomocí robota. Robot Pepper inicioval dialog buď v blízkosti osoby, nebo měl trénink nastavený na určitý čas. Robot motivoval člověka použít při tréninku tablet a provedl samotný trénink s použitím řeči, gest a tance. Informace, která se týkala tréninku, se zobrazovala na jeho tabletu, přičemž robot motivoval k použití jeho různých funkcí. Také samotný uživatel/pacient mohl začít trénink kdykoliv. Během experimentu účastníci dostávali otázky a odpovědi a zúčastnili se diskusí. Také se mohli zúčastnit průzkumů před, po a během experimentu.

Taneční pohyby robota Pepper byly použity při zkoumání interakce mezi člověkem a robotem a důvěry mezi nimi během zapamatování řeckých písmen v podobě tanečních pohybů robota [18]. Uživatel měl možnost studovat písmena buď pomocí tabletu, nebo zopakovat pohyb za robotem. Na identifikaci toho, jestli se člověk pohyboval či nikoliv, bylo použito zařízení Microsoft Kinect. Následně byly informace o stavu člověka posílány robotovi. Studování písmen bylo prováděno dvěma způsoby, které se lišily tím, že když uživatel chtěl přejít k dalšímu pohybu, na tabletu by stiskl tlačítko „další“, a při opakování pohybu by uživatel pro pokračování musel zopakovat poslední pohyb.

Na konci průzkumu byli účastníci dotázáni na jejich sympatii a důvěru k robotovi. Ve výsledcích průzkumu se ukázalo, že typ interakce má velký vliv na vznik a existenci důvěry mezi člověkem a robotem. Pokud uživatel měl na začátku experimentu nízkou důvěru k robotovi, při zahájení studia pomocí zopakování pohybů získal větší důvěru k robotovi, než při komunikaci přes displej. Autoři článku ale zmiňují, že to platí jenom pro účastníky s nižší důvěrou, a to může být spojené s tím, že tito lidé velmi podceňovali robota, ale pak zjistili, že robot dokáže udělat víc a tím pádem rychle stoupala jejich důvěra. Výsledky nezaznamenaly žádný rozdíl mezi skupinami ohledně sympatie k robotu Pepper, a to podle autorů může být způsobené tím, že forma interakce s Pepprem nebyla dostatečná na zvýšení sympatie k robotu.

3 Analýza hudby a stanovení tempa

Pro synchronizaci hudby s vygenerovanou animací je nutné stanovit parametr, který by tuto synchronizaci měl vyjádřit. K tomu je vhodný parametr BPM (údery za minutu) [19]. BPM určuje délku trvání tónů. Za jeden beat můžeme považovat jednu čtvrťovou notu. Jelikož celý takt může obsahovat až 4 čtvrťové noty, tak BPM udává počet těchto not za minutu. Např. když je BPM rovno 120 BPM, tak na každou vteřinu je to 120/60 = 2 beaty za vteřinu. Na každý beat bude navazován jeden pohyb robota. Hodnoty BPM se v klasické hudbě rozdělují na jednotlivé skupiny: larghissimo – je velmi pomalu (24 BPM and níž), larghetto – spíše pomalu (60–66 BPM), až presto – velmi rychle (168–200 BPM).

3.1 Algoritmus MIT Media Lab to Matlab

Jedná se o algoritmus vytvořený na Rice University v Americe [20]. Algoritmus reaguje na náhlé změny hodnot, tzv. skoky, ke kterým dojde při zpracování a postupné upravě signálu. Smyslem algoritmu je, že signál je rozdělen na několik skupin podle různých frekvencí. Následně se každý rozdělený signál zpracovává zvlášť. Pro každý rozdělený signál se spočítá IDFT, po čemž je signál vyhlazen, aby se zbavil šumu, a sledují se změny určitých hodnot. U zpracovaných signálů se monitoruje hodnota energie. Na výstupu algoritmus pro každou možnou hodnotu BPM spočte energii, a nakonec se z těchto hodnot zvolí maximální pro určení BPM.

FFT, neboli Rychlá Fourierova transformace [21], je rychlejší způsob výpočtu DFT – Diskrétní Fourierovy transformace. DFT se vyjadřuje z Fourierových řad pro diskrétní signály, tj. pro nespojité signály. Pro převedení spojitého signálu do diskrétní podoby je vyžadováno provedení vzorkování spojitého signálu do jednotlivých nespojitých vzorků. Proto je potřeba vzorkovací frekvence – obvykle se značí F_s. Vzorkovací frekvence musí být větší nebo rovna polovině maximální frekvence v signálu. Spojitý signál je spojitý v čase (nebo prostoru) a hodnotách, přičemž číslicový (či diskrétní) je omezený počtem hodnot reprezentujících signál a stanovuje se v jednotlivých hodnotách času [21]. DFT se používá pro výpočet spektra signálu. DFT lze vyjádřit následovně:

$$X[k] = \frac{1}{N} \sum_{n=0}^{N-1} x[n] exp(-j2\pi nk/N) \tag{3.1}$$

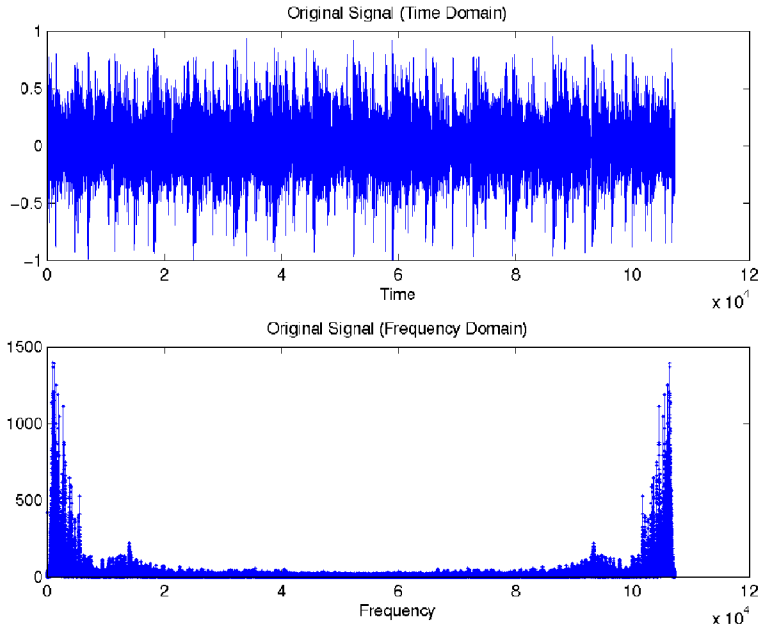
Kde N je počet vzorků vstupního signálu, k je perioda.

Opakem DFT je IDFT, neboli inverzní diskrétní Fourierova transformace, která převadí signál zpět ze spektra do časového prostoru [21], tím pádem je možné vzorec 3.1 přepsat jako:

$$x[k] = \frac{1}{N} \sum_{n=0}^{N-1} X[n] exp(-j2\pi nk/N) \tag{3.2}$$

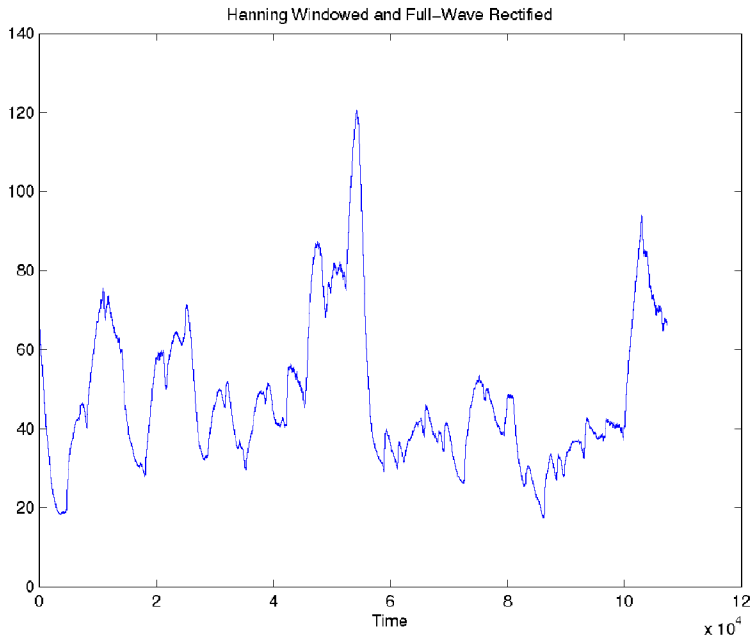
Podrobnější popis následujících kroků algoritmu:

- filtrace (obr. 3.1) – signál se rozděluje do jednotlivých skupin, každá skupina představuje frekvenci různých hudebních nástrojů. Signál se dělí na 6 skupin podle frekvence, tyto skupiny jsou 0-200 Hz, 200-400 Hz, 400-800 Hz, 800--1600 Hz, 1600-3200 Hz, a 3200 Hz-vzorkovací frekvence. Separace se provádí tak, že je spočítáno FFT vstupního signálu, následně se hodnoty z FFT rozdělují do jednotlivých skupin podle hodnot frekvencí ve spektru.



Obrázek 3.1: Originální signál (nahoře) a jeho spektrum (dole) [20]

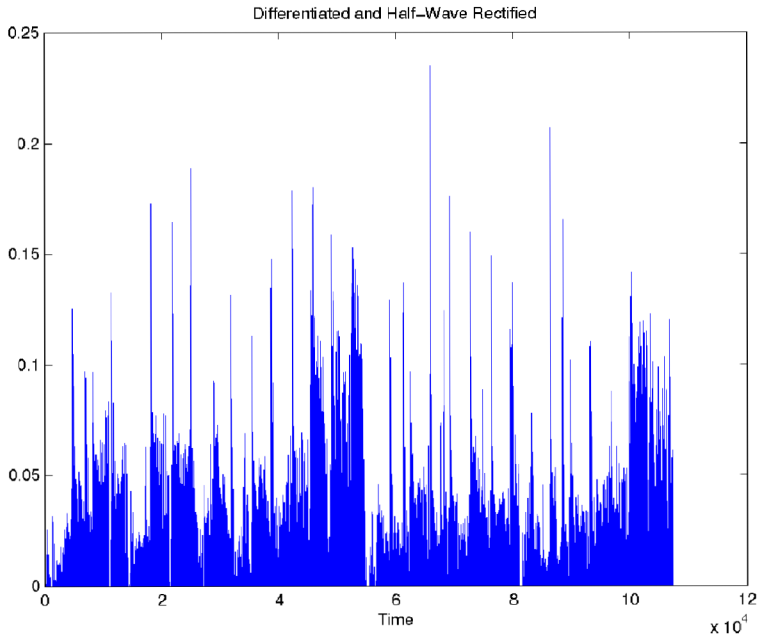
- vyhlazením (smoothing, obr. 3.2) signálu dosáhneme toho, že je možné vidět jenom požadované změny signálu bez šumu, a změny hodnot amplitudy zvuku jsou lépe viditelné. Poté se každá skupina spojuje s pravou polovinou Hammingova okénka o délce 400 ms. Hammingovo okno je nezápornou funkcí, která v podstatě představuje jednu periodu sinusoidy [22]. Tato funkce je symetrická v čase kolem hodnoty 0. Následují pomocné operace zkracující dobu výpočtu programu.



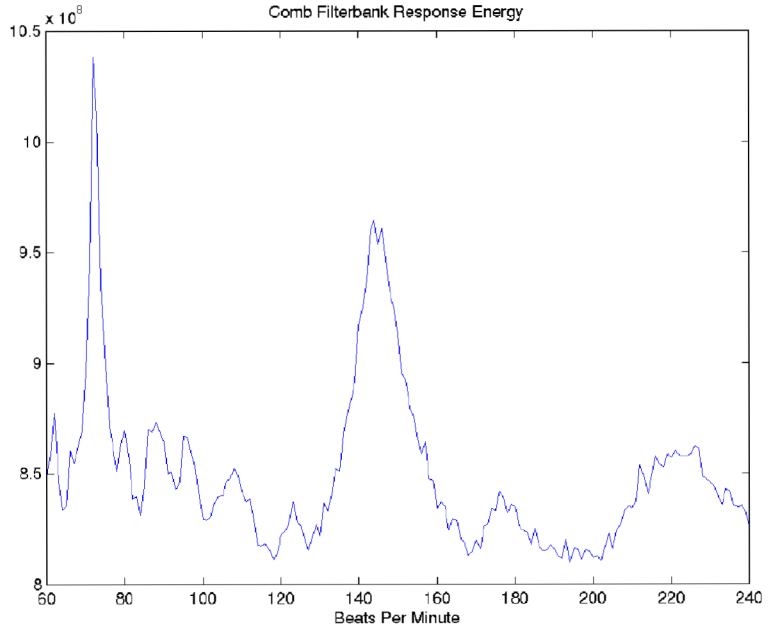
Obrázek 3.2: Vyhlazení signálu [20]

- následuje nalezení beatu (úderu) v každé skupině, náhlá změna signálu pro algoritmus značí beat. Po těchto krocích v signálu zbude pouze část vlny, ve které je možné hledat pouze zvýšení amplitudy, což značí výskyt úderu (obr. 3.3). Nejdříve v tomto kroku vstup projde přes diferenciátor (rozlišení), který odečítá následující vzorek signálu od vzorku stávajícího. Nakonec se při půlvlnovém usměrnění vyberou pouze kladné hodnoty.

- V tomto kroku se signály spojují s comb filtry – hřebenovými filtry. Comb filtr je filtr s přidáním zpožděného samotného signálu. Čím lépe se jejich tempa budou po jejich spojení shodovat, tím větší bude hodnota energie. Sada filtrů je vytvořena podle intervalu hodnot BPM, od minimální do maximální hodnoty. Tato sada se spojuje s každou skupinou segmentovaných signálů. Při spojení dostaneme energii. Pak se nalezené hodnoty energií všech pásem od výstupu každého comb filtru (tj. hodnoty BPM) sečtou. Maximální hodnota BPM je tempem zkoumané skladby. Graf s BPM je na obr. 3.4.



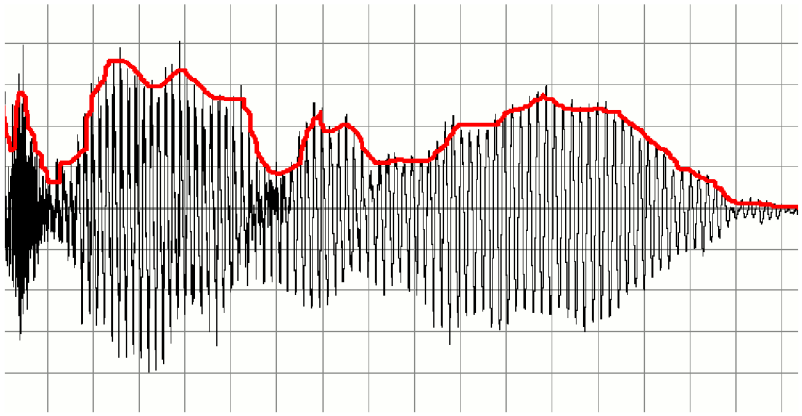
Obrázek 3.3: Signál rozlišený v čase a následně půlvlnově usměrněný [20]



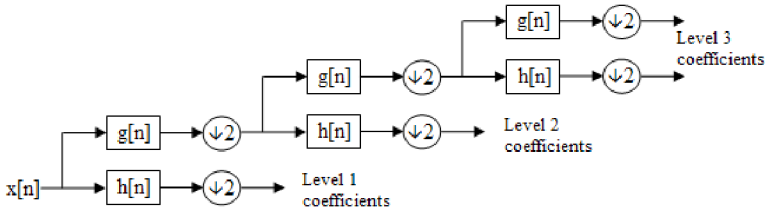
Obrázek 3.4: Určení BPM [20]

3.2 Algoritmus Beat-track od Eder de Souza

Algoritmus Beat-track od Eder de Souza je implementací algoritmu od George Tzanetakis [23] [24]. V případě tohoto algoritmu je signál také rozdělován na skupiny podle frekvencí, následně se pomocí extrakce obálky signálu 3.5 získává podstatná informace a vynechává se informace nepotřebná, jako je šum (jako krok v algoritmu MIT, kde je smoothing). Následně se provádí autokorelace za účelem nalezení dominující hodnoty BPM zkoumaného signálu. Signál se rozděluje podle frekvencí pomocí DWT – Diskrétní vlnkové transformace [25] (obr. 3.6). DWT je určena pro vlnkové signály, které jsou vzorkované. V porovnání s Fourierovou transformací zahrnuje informaci nejenom o frekvenci, ale i o časovém okamžiku. Přes signál prochází různé vlnky s různými parametry, část původního signálu a zvolená vlnka se vynásobí (je provedena konvoluce) z čehož je nalezeno množství vlnek v signálu. Během DWT signál prochází přes jednotlivé filtry. Filtr dolní (low) propust (značí se g) (3.3), na dekompozici signálu se používá horní (high) propust (h) (3.4), kde k je zpoždění v čase, n je index vzorku. Pomocí horní propusti se redukují nižší frekvence, a vyšší se zesilují. Pomocí dolní propusti se zesilují nižší frekvence, a vyšší se zeslabují [21]. Na výstupu etapy DWT jsou 2 koeficienty. Tato transformace může obsahovat několik úrovní (levels).



Obrázek 3.5: Extrakce obálky signálu [23]



Obrázek 3.6: DWT [23]

$$y_{low}[n] = \sum_{k=-\infty}^{\infty} x[k]g[2n-k] \tag{3.3}$$

$$y_{high}[n] = \sum_{k=-\infty}^{\infty} x[k]h[2n-k] \tag{3.4}$$

Výstup z předchozích kroků prochází Full Wave Rectification, což je výpočet absolutní hodnoty. Posléze je znovu aplikována dolní propust, kde y je výstupem z předchozího kroku, x je původní signál podle vztahu:

$$y[n] = (1-\alpha)x[n] - \alpha y[n] \tag{3.5}$$

Kde α je koeficientem filtru s dolní propustí.

Downsampling provádí redukci signálu pomocí odstranění hodnot. Např. downsampling s parametrem 3 znamená, že nový signál bude obsahovat jenom každou čtvrtou hodnotu původního signálu. Poté se provádí normalizace, která spočívá v odečtení střední hodnoty signálu od signálu původního. Po normalizaci se v signálu zvýrazní potřebné hodnoty. Jedním z posledních kroků je autokorelace pro detekci tempa, což je provedeno korelací signálu se sebou samotným zpožděným v čase (k je zpoždění, n je index vzorku) 3.6. Tímto způsobem jsou hledány dominantní hodnoty funkce (BPM).

$$y[k] = \frac{1}{N} \sum_{n=0}^{N-1} x[n]x[n+k] \tag{3.6}$$

Následuje krok s peak pickingem, což je hledání hodnoty a pozice vrcholu (peak), tj. maximální hodnoty ve výstupu z autokorelace. Pozice maxima je hledána podle následujícího vztahu:

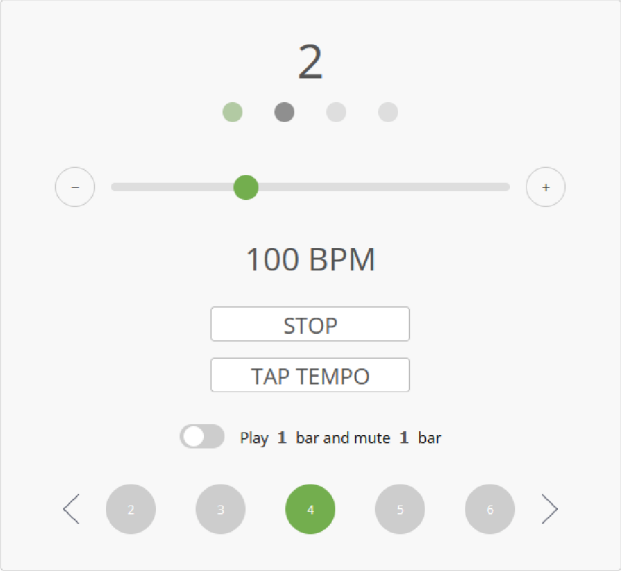
$$y[k] = 60 * k/x \tag{3.7}$$

k bylo ve výpočtu stanoveno na 250 Hz. x je pozice maximální hodnoty ve výstupu autokorelace.

3.3 Ověření funkčnosti algoritmů

Výše uvedené algoritmy byly otestovány na jednotlivých skladbách, stažených z webu, kde jsou umístěné skladby, na které se nevztahuje autorské právo [27]. BPM u skladeb bylo určeno několika způsoby:

- ručně – při poslechu hudby s neznámým parametrem BPM v online metronomu [26] byl ručně odhadnut (klepáním na tlačítko „TAP TEMPO“, z čehož online metronom vygeneroval BPM – obr. 3.7). V metronomu je možnost vybrat počet čtvrťových not v taktu, posléze jsou stisknutím tlačítka „TAP TEMPO“ několikrát za sebou generovány údery pro určení tempa.
- v online BPM analyzátoru [28], kam je možné nahrát soubor s hudbou, který je zanalyzován a je vyhodnoceno BPM.



Obrázek 3.7: Ukázka online metronomu [26]

- pomocí algoritmů zmíněných v sekcích 3.1 a 3.2.

Každá analyzovaná skladba měla odlišný parametr BPM, obsahovala několik hudebních nástrojů a byla v různých žánrech. Výsledky jsou ukázány v tabulce 3.1.

Název	**Ručně**	**Online analýza**	**MIT**	**Beat-track**
La Citadelle.mp3	65 BPM	130 BPM	87 BPM	130 BPM
Lukewarm Banjo.mp3	120 BPM	120 BPM	60 BPM	120 BPM
Del Rio Bravo.mp3	60 BPM	120 BPM	60 BPM	80 BPM
Bit Bit Loop.mp3	100 BPM	100 BPM	200 BPM	50 BPM
Bavarian Seascape.mp3	110 BPM	110 BPM	220 BPM	110 BPM
Beat Thee.mp3	46 BPM	91 BPM	91 BPM	200 BPM
Village Tarantella.mp3	145 BPM	73 BPM	145 BPM	72 BPM
Lovely Piano Song.mp3	50 BPM	100 BPM	67 BPM	50 BPM

Tabulka 3.1: Porovnání metod nalezení BPM jednotlivých skladeb

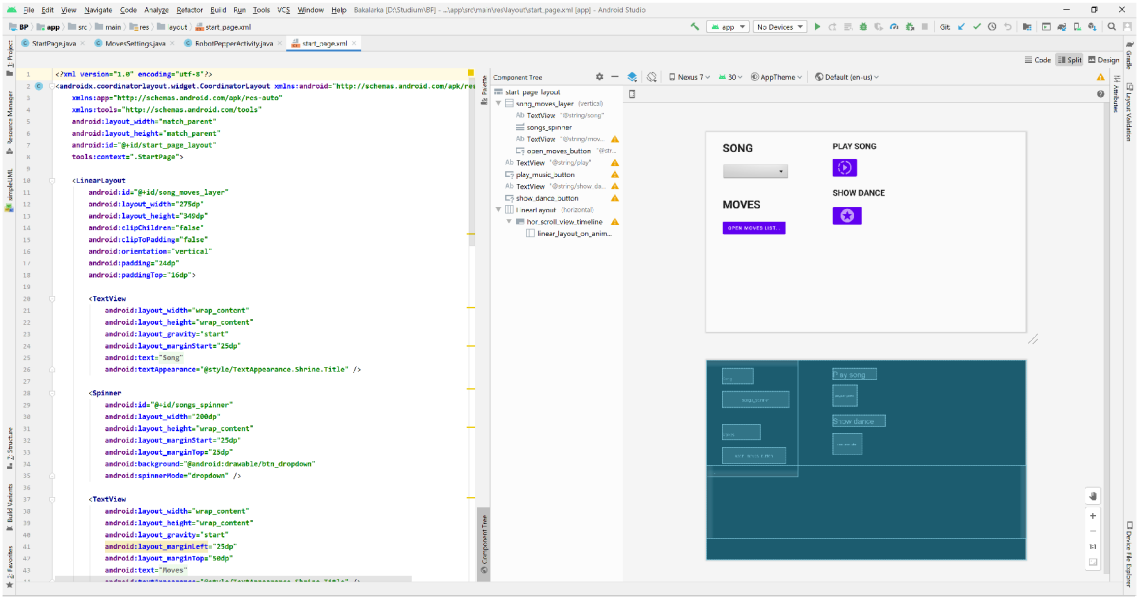
Jedním z problémů určení rytmu je to, že rozdíl mezi správným BPM a detekovaným je dělitelný dvěma, např. správný BPM je 120 BPM, a algoritmus má na výstupu 60 BPM. Příčinou je to, že 60 BPM je pouze dvakrát pomalejší, než 120, rytmus zůstává stejný, ale mění se tempo. Výsledky ručního určení rytmu se také liší od ostatních výsledků, protože správnost určení ručně záleží na vlastním „citu“ tempa. Ale všechny metody většinou generovali výstup správně, což bylo násobkem správné hodnoty BPM.

Algoritmus MIT je výrazně pomalejší, než algoritmus od George Tzanetakis, ale výsledné BPM algoritmu od George Tzanetakis se občas lišilo od jiných použitých metod. Algoritmus od George Tzanetakis měl také nižší spodní hranici detekce tempa, než algoritmus MIT. Minimální hodnota BPM, kterou by algoritmus byl schopný určit, je tedy 40 BPM. Algoritmus od MIT má vyšší horní hranici – 240 BPM. Výsledky výstupu z Online analyzéru tempa se ve více případech shodovali s výsledky určení tempa pomocí algoritmů a pomocí ručního určení tempa.

4 Popis realizace aplikace pro ovládání robota

Pro demonstraci pohybových možností robota Pepper s použitím různých skladeb a tanců v různých žánrech byla naprogramována aplikace v Android Studiu, která umožňuje uživateli vytvořit tanec ze seznamu pohybů, tj. vybrat pohyb, nebo animaci, pro robota a také vybrat hudbu z předložené nabídky. Aplikace následně synchronizuje robota s vybranou skladbou a robot předvede tanec.

4.1 Vytváření aplikace ve vývojovém prostředí Android Studio

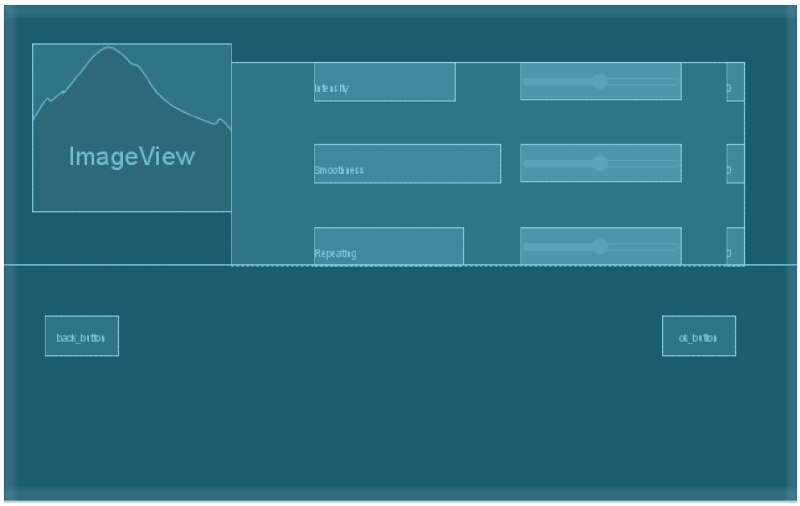


Obrázek 4.1: Ukázka Layout Editoru

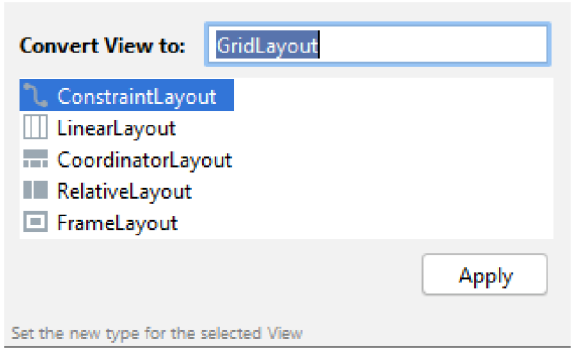
Vytváření oken aplikace probíhá v layout editoru (obr. 4.1). Layout, v českém jazyce rozložení, představuje hranice každého jednoho prvku rozhraní aplikace. Může obsahovat buď celé okno aplikace, nebo samostatný prvky (např. scrollovací menu). Programování oken je možné provádět v souboru formátu značkovacího jazyka XML přímo použitím editoru nebo psaním kódu v programu aplikace. Na layout

se přidávají různé UI elementy (elementy uživatelského rozhraní), jako jsou tlačítka, obrázky, nebo rozbalovací menu, které se zobrazují ve složce Component Tree a v samotném souboru. Tyto soubory se nachází ve složce app/res/layout. Soubor obsahuje jednotlivé tagy, každý tag má parametr id a každý je možné nastavit buď v editoru nebo v kódu. Základní tagy, použité v aplikaci, jsou následující:

- Layout – např. CoordinatorLayout, LinearLayout – definuje způsob rozdělení plochy. Okno se rozděluje na části (obr. 4.2), například v CoordinatorLayout se nahoru přidává jedna hlavní „vrstva“, pod hlavní vrstvou jsou vrstvy vedlejší 4.1. Vedlejší vrstvy mohou obsahovat další layouty či jednotlivé elementy. Tag obsahuje následující základní parametry: layout_width a layout_height, neboli šířka a výška, orientation udává orientaci, vertikální a horizontální. Tento parametr určuje orientaci layoutu vůči orientaci zařízení. Padding určuje odstup vnějších prvků. GridLayout je layout ve formě „mřížky“, obsahuje parametry columnCount udávající počet sloupců v mřížce, rowCount udávající počet řádků a alignmentMode pro zarovnání.



Obrázek 4.2: Ukázka hranic jednotlivých UI prvků jednoho okna aplikace



Obrázek 4.3: Typy layoutů

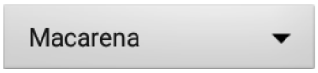
- View (obr. 4.4a a 4.4b), jako je např. ImageView nebo TextView, obsahuje komponenty jako jsou textový nadpis nebo obrázek. Základními parametry jsou layout_width a layout_height, layout_marginStart, layout_marginTop – odstup od horní či dolní hranice layoutu. Dále obsahuje vlastní resource, např. text či obrázek. TextAppearance dovoluje upravit vzhled textu a jeho styl. Layout_gravity pomáhá určit polohu textu uvnitř layoutu. TextSize mění rozměr textu. Src u ImageView popisuje zdroj (soubor s obrázkem). U CardView můžeme použít parametr clickable, který jestli je nastaven na hodnotu true, tak při kliknutí v mezích view je na něj aplikace schopná reagovat. CardElevation je výška CardView.

(a) TextView s vnořeným textem

(b) CardView s vnořeným obrázkem

Obrázek 4.4: Ukázka elementů

- Spinner (obr. 4.5a) je prvek, který umožňuje vybrat jeden element z nabídky. Následně zobrazuje aktuálně vybraný prvek. Základní parametry: layout_width, layout_height, layout_marginStart, layout_marginTop, spinnerMode umožňuje vybrat vzhled spinneru, background určuje vzhled pozadí za textem spinneru.
- SeekBar (obr. 4.5b) je rozšířením ProgressBar, neboli ukazatel průběhu. Přetahováním doleva a doprava se mění a zobrazuje hodnota (v aplikaci od 0 do 100). Použitými parametry jsou layout_width a layout_height.



(a) Spinner s vybranou položkou

(b) Seekbar s nastavenou hodnotou

Obrázek 4.5: Ukázka elementů

- MaterialButton – element „tlačítko“, na které je možné kliknout pro provedení požadované akce. Použitými parametry jsou layout_width, layout_height, layout_marginStart, layout_marginTop, text mění text tlačítka, gravity určuje polohu prvku uvnitř hranic tlačítka, icon mění vzhled tlačítka, iconGravity a iconSize pomáhají určit polohu a rozměr ikony.
- ViewPager se používá pro listování stránek doleva-doprava. Při listování se mění obsah stránky. Obsahuje parametry layout_width, layout_height a layout_behavior, které určují typ a vzhled přepínání stránek.

4.2 Popis tvorby aplikace ve vývojovém prostředí Android Studio

Pro implementaci základních aktivit aplikace jako je stisknutí tlačítka nebo volba skladby byly použity standardní metody vývojového prostředí Android Studio. Každé okno aplikace je naprogramováno v oddělené třídě.

Základní třídou je MainActivity (obdoba metody Main) [29]. Tato třída v apllikaci rozšiřuje určitý typ aktivity. Pro potřeby této práce třída rozšiřuje třídu AppCompatActivity. Třída Fragment je samostatnou částí uživatelského rozhraní, která je používána opakovaně [30]. Fragment má vlastní layout, životní cyklus a může sám zpracovávat vstupy. Jelikož Fragment může být samostatnou části UI, každé okno navržené aplikace může implementovat Fragment. Fragmentem může být i část okna aplikace, jako je např. scrollovací prvek okna. Při běžící aktivitě je možné přidávat, odebírat, nebo měnit libovolné fragmenty. Přechod mezi fragmenty se provádí pomocí interface NavigationHost, který obsahuje metodu navigateTo, kde jako parametr je exemplář třídy s potřebným oknem či částí okna.

U každého elementu uživatelského rozhraní je možné nastavit id nebo název elementu. Podle názvu je možné se obrátit na elementy v kódu aplikace a volat na ně potřebné metody. Id v těle metody používá metoda onCreateView (u Fragmentu) nebo onCreate (u tříd typu Activity). U Fragmentu se prvek uživatelského rozhraní, jako je samotný layout, obrázek či text, vyvolává pomocí metody LayoutInflater inflate. U třídy typu Activity se používá metoda onCreate s metodou setContentView (int layoutResID), layoutResID – resource id layoutu. Každý element podřízený tomuto prvku je možné vyvolat pomocí metody findViewById, která jako parametr přijímá id elementu přes R.id.nazev_elementu (R je zkrátka od resources).

V aplikaci se pro vytvoření tance používají UI elementy „tlačítka“. Pro nastavení akce po stisku tlačítka pro určité tlačítko je používána metoda setOnClickListener, ve které je definována metoda View.OnClickListener(), která obsahuje metodu onClick. V těle poslední metody jsou implementovány potřebné aktivity.

```
// FragmentName – jmeno exemplaru tridy Fragment,
// button – tlacitko

button.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    ((NavigationHost) getActivity())
                            .navigateTo(new FragmentName());
                }
            });
```

QiSDK obsahuje třídu RobotPepperActvity. RobotPepperActivity dědí od abstraktní třídy RobotActivity, která implementuje třídu AppCompatActivity. Na to, aby bylo možné ovládat robota přes naprogramované metody, se používají metody interface RobotLifecycleCallbacks [31]. Tento interface obsahuje metody onRobotFocusGained(QiContext qiContext) – aktivita získala fokus na robota, onRobotFo-

cusLost() – aktivita ztratila fokus na robota, onRobotFocusRefused(String reason) – aktivitě je odmítnut fokus na robota, parametrem této metody je příčina odmítnutí fokusu na robota.

Pro vytvoření a spuštění animace na robotovi se používají třídy AnimationBuilder a AnimateBuilder. Nejdříve je volána metoda AnimationBuilder.with, která nastavuje kontext, následně je volána metoda .withResources(resId), kde resId je id vytvořené animace. Na závěr je volána metoda .build(). Tímto způsobem získáme objekt typu Animation. Následně jsou provedeny podobné operace pomocí AnimateBuilder (AnimateBuilder.with(qiContext).withAnimation(animation).build()). Pro spuštění animace se využívá metoda .async().run().

```
Animation myAnimation = AnimationBuilder.with(qiContext)
                        .withResources(R.raw.animationResource)
                                .build();
Animate animate = AnimateBuilder.with(qiContext)
                                .withAnimation(myAnimation)
                                .build();
animate.async().run();
```

Součástí aplikace je uchování volby pohybů uživatele, jejich pořadí, nastavení, a také případné obnovení stránek aplikace. Pro tyto účely byly použity soubory, do kterých se ukládají a ze kterých se načítají požadovaná data. Nastavení pohybu je uloženo ve formátu JSON, kde každá položka obsahuje název vybraného pohybu. Na sestavení tance pro robota z jednotlivých animací se využívá skutečnost, že hotový tanec je možné načíst z animačního souboru jako text, který je členěný podle framů, části těla robota, atd. Mění se jenom frame a stupeň otočení končetiny robota. Vhodným řešením bylo sestavení textových řetězců z jednotlivých řádků. Frame se mění podle počtu framů na jeden beat. Ten je možné spočítat pomocí zjištěného BPM hudby. Jedním z parametrů je fps, tj. počet počet framů/snímků za jednu sekundu. Dělením BPM 60 zjistíme počet beatů/úderů za jednu vteřinu. Dělením fps počtem beatů za vteřinu dostaneme počet framů na jeden beat:

$$framePerBeat = \frac{fps \cdot 60}{BPM} \tag{4.1}$$

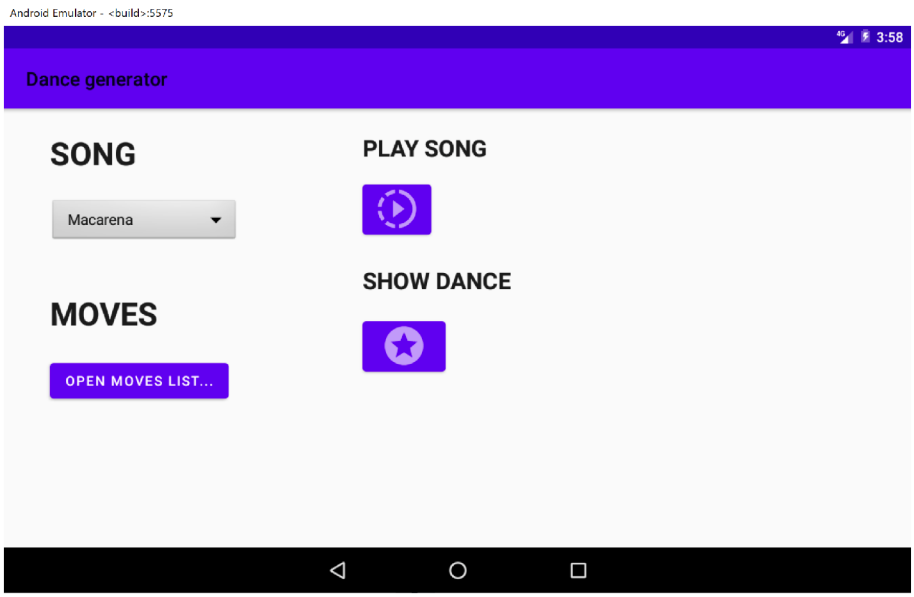
Hodnota framePerBeat je přičítána k aktuálnímu „framu“ pro každý další pohyb. Z textových řetězců je sestaven celý animační soubor, jehož část je zobrazená níže.

```
<ActuatorCurve fps="25" actuator="LShoulderPitch"
mute="false" unit="degree">
    <Key value="90.5273514" frame="40">
        <Tangent side="right" abscissaParam="13.3333333"
        ordinateParam="0"
        editor:interpType="bezier_auto"/>
    </Key>
    <Key value="-17.8714504" frame="80">
        <Tangent side="left" abscissaParam="-13.3333333"
```

```
            ordinateParam="0"
            editor:interpType="bezier_auto"/>
        </Key>
    </ActuatorCurve>
```

4.3 Přehled oken aplikace

Při spuštění aplikace je zobrazeno hlavní okno (obr. 4.6).



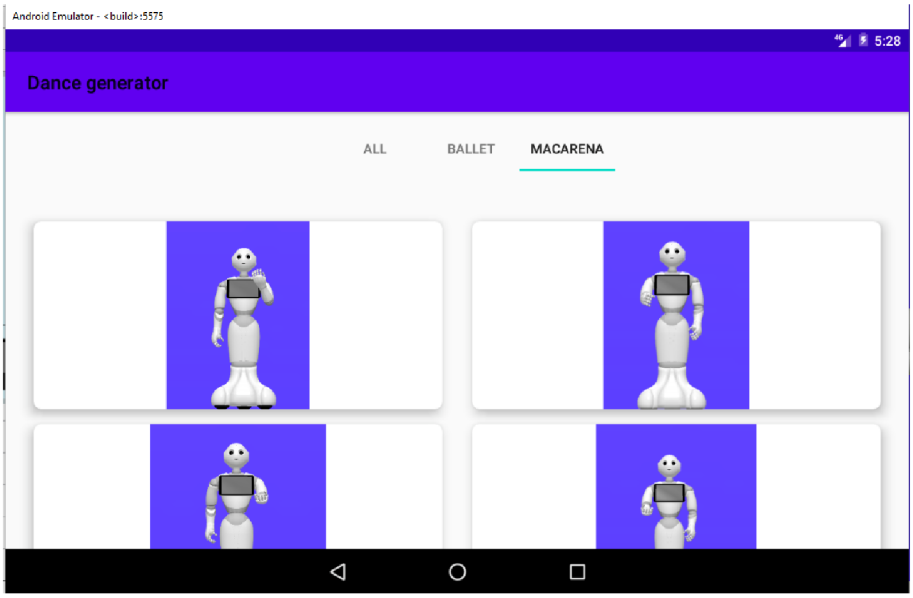
Obrázek 4.6: Hlavní okno aplikace

V hlavním okně jsou k nalezení následující prvky:

- Pod nadpisem SONG se v levém rohu nachází rozbalovací seznam, který slouží k výběru požadované hudby. V tomto seznamu se zároveň zobrazuje aktuálně vybrána skladba.
- Pod nadpisem PLAY SONG se nachází tlačítko, po jehož stisknutí se přehraje hudba vybraná ze seznamu. Při opětovném stisknutí tlačítka se hudba zastaví.
- Pod nadpisem MOVES je tlačítko „Open moves list…“. Po jeho stisknutí se otevře další okno aplikace, ve kterém je zobrazen seznam pohybů/animací.
- Pod nadpisem SHOW DANCE se nachází tlačítko spuštění připraveného tance/animace. Po jeho stisknutí se spustí animace na robotovi s hudebním doprovodem.

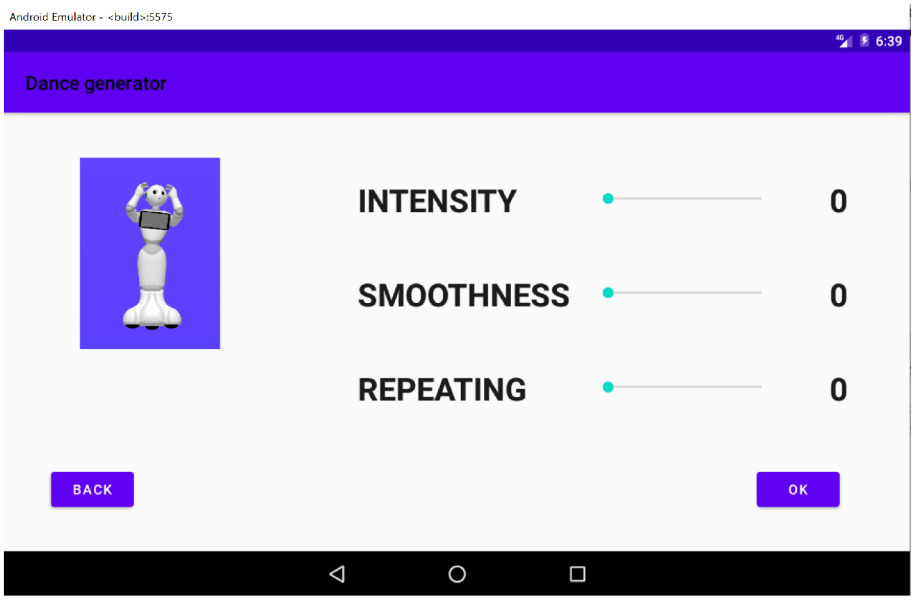
Po stisknutí tlačítka „Open moves list…“ aplikace přepne na okno se seznamem pohybů (obr. 4.7). Pohyby jsou předem připravené animace. Tyto animace jsou

rozděleny do skupin podle stylu (salsa, tango, rumba, …). Přepínání mezi skupinami stylů je umožněno listováním doleva-doprava. V každé skupině lze skrolovat dolů a nahoru pro zobrazení dalších pohybů a animací. Každá skupina je zobrazena jako seznam obrázků s demonstrací polohy končetin robota. Stisknutím obrázku uživatel přepne na další okno, přičemž při přechodu na nové okno robot předvede vybraný pohyb.



Obrázek 4.7: Okno aplikace se seznamem pohybů

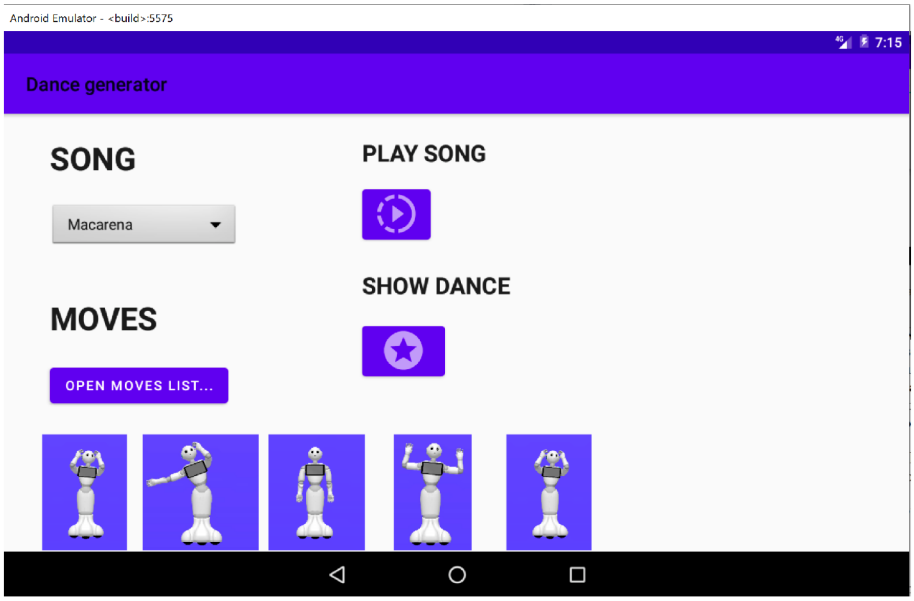
Další okno (obr. 4.8) aplikace obsahuje obrázek pohybu a 3 parametry pro nastavení, každý z nich má rozsah od 0 do 100:



Obrázek 4.8: Okno aplikace s nastavením parametrů pohybu/animace

- Intensity neboli intenzita pohybu. Tento parametr určuje rychlost provedení pohybu.
- Smoothness určuje plynulost spojení pohybů mezi sebou.
- Repeating určuje počet opakování pohybů za sebou.

Stisknutím tlačítka Back se aplikace vrátí na předchozí okno se seznamem pohybů. Stisknutím tlačítka Ok uživatel potvrdí svoji volbu a přejde na startovní stránku, v dolní části okna je zobrazen vybraný pohyb (obr. 4.9).



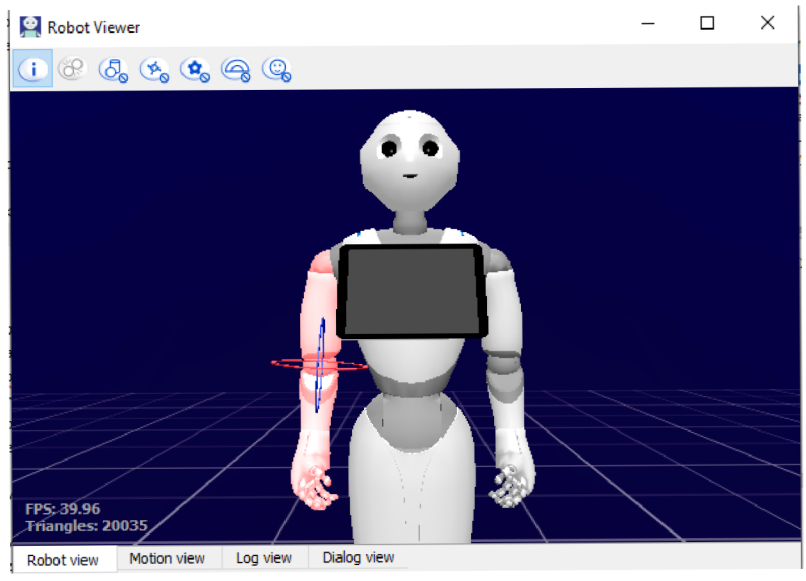
Obrázek 4.9: Hlavní okno s přidanými pohyby

4.4 Připojení k robotovi Pepper a spuštění aplikace

Spustit a odladit naprogramovanou aplikaci je možné v emulátoru v menu Tools - Pepper SDK - Robot SDK Manager. Při startu se spustí emulace Android tabletu a simulátor robota Robot Viewer (obr. 4.10). Vedle tlačítka pro spuštění aplikace se nachází zdroj, na kterém bude aplikace běžet. V případě emulace bude zdroj pojmenován jako „Virtual Device“, virtuální zařízení. Zařízení reálného robota bude mít název podobný „ARTNCORE LPT_200AR“. Je možné spustit více zdrojů najednou a před spuštěním aplikace zvolit požadované zařízení. V aplikaci Robot Viewer je možné ovládat robota, např. zvedat ruku. Samotná emulace robota se nachází v záložce Robot View. V dolní části okna je možné přepnout do jiných View. Motion View ovládá robota přes změnu parametrů kloubů. Log View zobrazuje logy různých úrovní, jako např. Error, Warning. Přes Dialog View si uživatel a robot mohou vzájemně posílat zprávy.

Pro připojení k reálnému robotovi je potřeba znát IP adresu tabletu. Po otevření v menu Android Studio Pepper SDK - Connect se zobrazí pole Address, kam se

zadává zjištěná IP adresa. Po úspěšném spojení se vedle tlačítka pro spuštění aplikace zobrazí jméno připojeného robota, a otevře se aplikace Robot Viewer pro přímé ovládání robota. Emulovaná aplikace Robot Viewer má modré pozadí, aplikace pro přímé ovládání robota má zelené pozadí.



Obrázek 4.10: RobotView v Android Studio

5 Závěr

Podle zadání práce byla vytvořena aplikace, která poskytuje možnost připravit pro robota Pepper tanec. Pro tanec uživatel vybírá požadovanou skladbu, kterou program s tancem synchronizuje. Uživatel má možnost skládat tanec z pohybů z různých stylů. Pro synchronizaci byl použit parametr tempa hudby (BPM – beats per minute), k jehož nalezení byl implementován algoritmus, který ho dokáže analýzou signálu určit.

Program byl realizovaný ve vývojovém prostředí Android Studio z toho důvodu, že součástí zařízení robota je tablet s operačním systémem Android. Forma aplikace byla zvolena, protože v aplikaci lze snadno ukázat možnosti, které uživatel má, a výsledný tanec. Android Studio nabízí různé formy realizace rozhraní, pomocí tohoto prostředí je možné vytvořit aplikaci splňující požadované cíle různými způsoby. Vytvořená aplikace obsahuje následující funkce:

- vybrat pohyb,
- přidat pohyb do výsledného tance,
- nastavit počet opakování zvoleného pohybu,
- vybrat skladbu,
- nastavit rychlost pohybu,
- nastavit plynulost pohybů pomocí interpolace,
- spustit tanec.

Možná vylepšení pro budoucí vývoj aplikace se nabízejí v podobě přidání více funkcí, jako je možnost zvolit část animace, kterou by robot následně přehrál, nebo měnit tempo skladby (zrychlovat či zpomalovat), či možnost vytváření uživatelem definovaných pohybů, které by následně bylo možné ukládat do seznamu pohybů. Dále například změna implementace „časové osy“ s animacemi, aby uživatel viděl přesný čas a délku trvání jednotlivých pohybů.

Algoritmus, který zjišťuje BPM, občas detekuje tempo s mírnými odchylkami, ale vždy v souladu s tempem hudby. Dalším problémem algoritmu je jeho spodní limit rozsahu detekovaného tempa, 60 BPM, algoritmus tedy musí zpracovat minimálně 2,2 sekundy. Odchylky vznikají i ve chvíli, kdy se rytmus/tempo v průběhu písničky mění. Na většinu písniček nalezené algoritmy ale funguji dobře. Pro lepší volbu pohybů by bylo vhodné sledovat nejen tempo, ale i další parametry, jako jsou např.

použité hudební nástroje, tonalita skladby, její žánr (rock, jazz, klasická hudba) apod.

Co se týče pohybu robota, prostředí Animation Editor nabízí sadu nástrojů na vytvoření pohybů. S robotem se pohodlně pracuje, v tomto prostředí lze vyzkoušet různé kombinace pohybů, upravovat je, přehrávat animace, nastavovat ručně plynulost pohybu. Simulace robota v Robot Viewer se však může lišit od skutečných pohybů robota. Samotný robot má klouby umístěné podobně jako člověk, nicméně postrádá nohy, což při tanečních realizacích ovlivňuje dynamiku pohybů. V editoru je možné nastavit i rychlost pohybu, řídí se však určitými omezeními. Robot se může hýbat jen v určitých směrech a jeho klouby jsou schopné se pohybovat často velice omezeně, např. o 0,5°. Robot tedy nemůže replikovat lidský tanec naprosto přesně.

Další omezení pohybů robota může nastat při detekci překážky. Dokonce je možné, že samotný uživatel bude zaznamenán jako překážka ve chvíli, kdy spustí tanec z připevněného tabletu. V takovém případě je možným řešením přenastavení bezpečnostní vzdálenosti senzorů, nebo přímo celkové vypnutí kontroly vzdálenosti.

Pro objektivní zhodnocení pohybů robota je nutno podotknout, že robot provádí pohyby dobře, ale kdyby měl předvádět tance v roli učitele, nebo se zúčastnit taneční soutěže, tak by nebyl příliš úspěšný. To je způsobené tím, že přechody mezi jednotlivými pohyby robot provádí nejkratší cestou, na rozdíl od člověka, který je schopný do pohybu přidat větší flexibilitu. Zároveň tak mohou pohyby robota působit prudce a stroze. Strohost pohybů je ale možné zredukovat vhodnou volbou navazujících pohybů a pečlivou úpravou parametrů jednotlivých pohybů.

Použitá literatura

[1] MOGG, Trevor. *Pepper the robot fired from grocery store for not being up to the job* [online]. 2018 [cit. 2022-05-07]. Dostupné z: https://www.digitaltrends.com/cool-tech/pepper-robot-grocery-store/.

[2] *Pepper - Developer Guide* [online] [cit. 2022-05-07]. Dostupné z: http://doc.aldebaran.com/2-5/family/pepper_technical/index_dev_pepper.html.

[3] *SoftBank Robotics documentation: Joints* [online] [cit. 2022-05-07]. Dostupné z: http://doc.aldebaran.com/2-5/family/pepper_technical/joints_pep.html.

[4] *SoftBank Robotics documentation: Links* [online] [cit. 2022-05-09]. Dostupné z: http://doc.aldebaran.com/2-5/family/pepper_technical/links_pep.html.

[5] *SoftBank Robotics documentation: Sonars* [online] [cit. 2022-05-07]. Dostupné z: http://doc.aldebaran.com/2-5/family/pepper_technical/sonar_pep.html.

[6] *SoftBank Robotics documentation: Contact and tactile sensors* [online] [cit. 2022-05-07]. Dostupné z: http://doc.aldebaran.com/2-5/family/pepper_technical/contact-sensors_pep.html.

[7] *Creating a robot application* [online] [cit. 2022-05-07]. Dostupné z: https://developer.softbankrobotics.com/pepper-qisdk/getting-started/creating-robot-application.

[8] *Creating Keyframes* [online] [cit. 2022-05-07]. Dostupné z: https://developer.softbankrobotics.com/pepper-qisdk/tools/animation-editor/creating-keyframes.

[9] *XML introduction* [online] [cit. 2022-05-07]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/XML/XML_introduction.

[10] *Boston Dynamics "Spot" and Softbank Robotics "Pepper" Collaborative Robot dance* [online] [cit. 2022-05-07]. Dostupné z: https://www.youtube.com/watch?v=G9p9jdmJQOQ.

[11] AALTONEN, Iina et al. *Hello Pepper, May I Tickle You? Children's and Adults' Responses to an Entertainment Robot at a Shopping Mall* [online]. 2017 [cit. 2022-05-07]. Dostupné z: https://dl.acm.org/doi/10.1145/3029798.3038362.

[12] *Pepper at Westfield Shopping Centers: Impact Story* [online] [cit. 2022-05-13]. Dostupné z: https://www.youtube.com/watch?v=FhWAsnPOn_w.

[13] POLLMANN, Kathrin et al. *Robot vs. Voice Assistant: Is Playing with Pepper More Fun than Playing with Alexa?* [Online]. 2020 [cit. 2022-05-07]. Dostupné z: https://dl.acm.org/doi/10.1145/3029798.3038362.

[14] THÖRN, Oscar, Peter KNUDSEN a Alessandro SAFFIOTTI. *Human-Robot Artistic Co-Creation: a Study in Improvised Robot Dance* [online]. 2020 [cit. 2022-05-07]. Dostupné z: https://ieeexplore.ieee.org/abstract/document/9223446.

[15] *Robot dance at the "Feast Concert" (Örebro University, January 31, 2020)* [online] [cit. 2022-05-07]. Dostupné z: http://crea.oru.se/Music/.

[16] SCHRUM, Mariah, Chung Hyuk PARK a Ayanna HOWARD. *Humanoid Therapy Robot for Encouraging Exercise in Dementia Patients* [online]. 2019 [cit. 2022-05-07]. Dostupné z: https://ieeexplore.ieee.org/abstract/document/8673155.

[17] SCHÜSSLER, Sandra et al. *The Effects of a Humanoid Socially Assistive Robot Versus Tablet Training on Psychosocial and Physical Outcomes of Persons With Dementia: Protocol for a Mixed Methods Study* [online]. 2020 [cit. 2022-05-07]. Dostupné z: https://www.researchgate.net/publication/339142770_The_Effects_of_a_Humanoid_Socially_Assistive_Robot_Versus_Tablet_Training_on_Psychosocial_and_Physical_Outcomes_of_Persons_With_Dementia_Protocol_for_a_Mixed_Methods_Study.

[18] YE, Sean, Karen FEIGH a Ayanna HOWARD. *Learning in Motion: Dynamic Interactions for Increased Trust in Human-Robot Interaction Games* [online]. 2020 [cit. 2022-05-07]. Dostupné z: https://ieeexplore.ieee.org/document/9223437.

[19] *Tempo* [online] [cit. 2022-05-07]. Dostupné z: https://en.wikipedia.org/wiki/Tempo.

[20] *Beat This: A Beat Synchronization Project* [online] [cit. 2022-05-07]. Dostupné z: https://www.clear.rice.edu/elec301/Projects01/beat_sync/beatalgo.html.

[21] ING. JAN NOUZA, CSc. prof. *Signály a informace* [online] [cit. 2022-05-07]. Dostupné z: https://elearning.tul.cz/course/view.php?id=7024.

[22] *Hann or Hanning or Raised Cosine* [online] [cit. 2022-05-09]. Dostupné z: https://ccrma.stanford.edu/~jos/sasp/Hann_Hanning_Raised_Cosine.html.

[23] SOUZA, Eng Eder de. *Beat per Minutes calculation* [online]. 2012 [cit. 2022-05-07]. Dostupné z: https://github.com/ederwander/Beat-Track.

[24] TZANETAKIS, George. *Tempo Extraction using Beat Histograms* [online]. 2005 [cit. 2022-05-09]. Dostupné z: https://www.researchgate.net/publication/228744106_Tempo_extraction_using_beat_histograms.

[25] TALEBI, Shawhin. *The Wavelet Transform* [online]. 2020 [cit. 2022-05-07]. Dostupné z: https://towardsdatascience.com/the-wavelet-transform-e9cfa85d7b34.

[26] MUSICCA. *Online metronome* [online] [cit. 2022-05-07]. Dostupné z: https://www.musicca.com/metronome.

[27] *FREEPD, 100 % Free Music - Creative Commons 0* [online] [cit. 2022-05-15]. Dostupné z: https://freepd.com/misc.php.

[28] TUNEBAT. *Song Key & BPM Finder* [online] [cit. 2022-05-07]. Dostupné z: https://tunebat.com/Analyzer.

[29] *Activity* [online] [cit. 2022-05-07]. Dostupné z: https://developer.android.com/reference/android/app/Activity.

[30] *Fragments* [online] [cit. 2022-05-07]. Dostupné z: https://developer.android.com/guide/fragments.

[31] *Mastering Focus & Robot lifecycle* [online] [cit. 2022-05-07]. Dostupné z: https://developer.softbankrobotics.com/pepper-qisdk/principles/mastering-focus-robot-lifecycle.