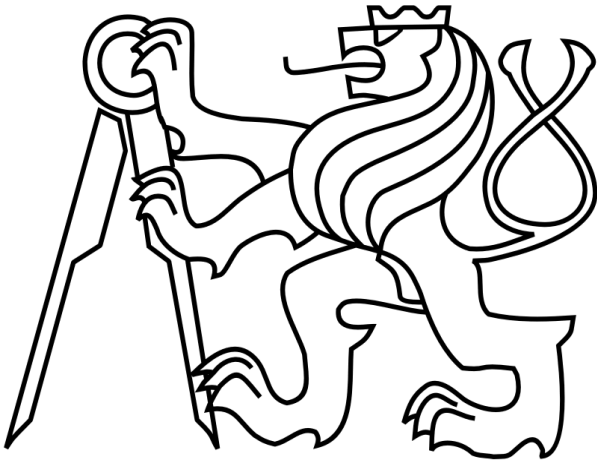
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta elektrotechnická

Katedra kybernetiky



Bakalářská práce

Využití atributů detekovaných podle tváře při interakci Nao robotu s lidmi

Use of faces attributes in Nao-human interaction

Autor: Alena Petráčková

Vedoucí práce: RNDr. Daniel Průša, Ph.D

Praha 2012

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra kybernetiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: Alena Petráčková

Studijní program: Kybernetika a robotika (bakalářský)

Obor: Robotika

Název tématu: Využití atributů detekovaných podle tváře při interakci Nao robotu s lidmi

Pokyny pro vypracování:

1. Seznamte se s robotem Nao [1]. Získejte přehled o cílech projektu HUMAVIPS, který s uvedeným modelem robotu pracuje [2].
2. Seznamte se s knihovnami, které vznikly v rámci projektu HUMAVIPS pro rozpoznávání atributů lidské tváře (pohlaví, odhad věku).
3. Navrhněte aplikaci pro interakci Nao robotu s lidmi, která bude využívat atributy zmíněné v předchozím bodě.
4. Implementaci doplňte programátorskou dokumentací.
5. Aplikaci vyzkoušejte na vhodné skupině lidí. Vypracujte studii o průběhu testování a chování účastníků.

Seznam odborné literatury:

[1] Aldebaran Robotics Academics forum. http://academics.aldebaran-robotics.com/
[2] HUMAVIPS web page. http://humavips.inrialpes.fr/
[3] Harms, D.; McDonald, K.: Začínáme programovat v jazyce Python. Computer Press, Brno, 2006.

Vedoucí bakalářské práce: RNDr. Daniel Průša, Ph.D.

Platnost zadání: do konce zimního semestru 2012/2013

L.S.

prof. Ing. Vladimír Mařík, DrSc.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 9. 11. 2011

Prohlášení

Prohlašuji, že jsem předloženou práci vypracovala samostatně a že jsem uvedla veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze, dne 25.5 2012 ……. …….

Podpis autora práce

Poděkování

Děkuji vedoucímu práce panu RNDr. Danielu Průšovi, Ph.D za ochotu a pomoc při psaní a tvorbě programu do této bakalářské práce. Mé poděkování patří také rodičům za trpělivost a podporu.

Abstrakt

Tato bakalářská práce se zabývá robotem NAO, popisuje jeho funkce a vybavení. Dále se zabývá projektem HUMAVIPS, který s tímto typem robota pracuje. Je uveden popis projektu, představeny jsou jeho cíle a vize.

Hlavní část práce je věnována návrhu a implementaci aplikace pro komunikaci mezi NAO robotem a člověkem. Jedná se o dotazník, kde robot klade dotázaným sérii otázek a zaznamenává jejich odpovědi. Při testování aplikace byla zvolena sada pěti otázek zaměřených na věk, pohlaví, zvolené prostředí a vizi o budoucnosti s roboty.

Výsledkem zpracování anketních otázek je studie o rozložení získaných odpovědí vzhledem k pohlaví nebo věku, popis chování účastníků ankety a představení vhodného vzorce chování NAO robota.

Klíčová slova

Robot Nao, interakce robota a člověka, Python

Abstract

This bachelor thesis deals with the NAO robot, describes its functions and equipment. It also focuses on project HUMAVIPS, which works with this type of robot. The project is described, its goals and visions are presented.

The main part is devoted to the design and implementation of an application for communication between the NAO robot and human. It is a questionnaire consisting of a series of questions that are asked. The robot records answers.

For an application testing, five questions regarding age, sex, chosen environment and a vision of the future with robots have been selected.

The result of processing answers is a study of responses distribution by gender or age and a description of the behavior of participants. Appropriate behavioral patterns of NAO robot have also been suggested.

Keywords

Robot Nao, Nao-human interaction, Python

Obsah

Abstrakt I
Klíčová slova I
Abstract II
Keywords II
Obsah III
Seznam použitých zkratek a symbolů VI
Seznam obrázků VII
Seznam grafů VIII

1. Úvod 1
2. Robot NAO 2
2.1. Popis robota 3
2.2. Modely robota 4
2.2.1. Verze hlav robota NAO 4
2.2.2. Typy těl robota NAO 4
2.3. Software 5
2.3.1. NAOqi 6
2.3.2. Choreographe 6
2.3.3. Monitor 6
2.3.4. NAO Sim 6
3. Projekt HUMAVIPS 7
3.1. Popis projektu 7
3.2. Projekt a jeho cíle 7
3.2.1. Otázky projektu HUMAVIPS 8
4. Aplikace pro interakci robota NAO s lidmi 11
4.1. Popis aplikace 11

4.2. Zvolené prostředí a otázky 11
4.2.1. Základní otázky 12
4.2.2. Otázka budoucnosti 12
4.2.3. Robot na Ortopedii 13
4.2.4. Shrnutí 14
4.3. Popis programu aplikace 15
4.3.1. Box *Vyhodnocení* 15
4.3.2. Rozpoznávání známého obličeje 16
4.3.3. Vzhled robota 16
4.3.4. Detaily a omezení aplikace 17
4.3.5. Schéma funkcí 18
5. Dokumentace programu 20
5.1. Box *Data_face reco.* 20
5.2. Box *Data_visual reco.* 20
5.3. Box *Vyhodnocení* 21
5.3.1. Funkce *def __init__(self):* 21
5.3.2. Funkce *def onInput_inputFace(self, a):* 24
5.3.3. Funkce *def onInput_inputObject(self, b):* 25
5.3.4. Funkce *def starting(self):* 26
5.3.5. Funkce *def oftenDetectedFace(self, facelist):* 27
5.3.6. Funkce *def oftenDetectedAnswer(self, objectlist):* 27
5.3.7. Funkce *def make_nameCoordinates(self, p):* 27
5.3.8. Funkce *def make_yesNoCoordinates(self, p):* 28
5.3.9. Funkce *def classify_distances(self):* 28
5.3.10. Funkce *def classify_faceObject(self):* 28
5.3.11. Funkce *def eyeColor(self, p, sGroup):* 29
5.3.12. Funkce *def saySomething(self, s):* 29
5.3.13. Funkce *def yes(self):* 29
5.4. Učení databáze objektů 29
5.4.1. Zkušenosti a omezení 30
5.5. Generování kódu pohybu 31
5.5.1. Návod 31

6. Aplikace v praxi **34**
6.1. Výsledky ankety 35
6.2. Základní zpracování ankety – pohlaví a věk 37
6.3. Zpracování ankety – konkrétní otázky 38
6.3.1. Budoucnost s roboty 38
6.3.2. Spokojenost s léčbou a ošetřením 39
6.3.3. Doba léčby pacientů 40
6.4. Korelace jednotlivých otázek 41
6.4.1. Stáří a doba léčby pacientů 42
6.4.2. Pohlaví a víra v budoucnost s roboty 43
6.4.3. Stáří a víra v budoucnost s roboty 44
6.5. Vzorec chování robota 45
6.5.1. Vzorec pro pohlaví a věk 45
6.5.2. Vzorec podle otázky "Věříte, že za 100 let nahradí roboti … 45
6.5.3. Vzorec podle spokojenosti pacienta s léčbou a ošetřením 46
6.5.4. Vzorec podle doby léčby pacienta 46
6.5.5. Vzorec – závěr 47
6.6. Shrnutí ankety 47
7. Závěr **48**

Zdroje 49
Zdroje obrázků 50

Příloha A – Program 52
Příloha B – Korelace 64
Příloha C – Asterix 65

Seznam použitých zkratek a symbolů

FN	Fakultní nemocnice
HUMAVIPS	Humanoids with auditory and visual abilities in populated spaces
	Humanoidní roboti s audiovizuálními schopnostmi v zalidněných prostorách
LF	Lékařská fakulta
UK	Univerzita Karlova

Seznam obrázků

Obrázek 1:	*Robot NAO*	*2*
Obrázek 2:	*Popis robota NAO*	*3*
Obrázek 3:	*Verze robota NAO:* V4, V3.3, V3+ a V3.2	*4*
Obrázek 4:	*Typy těla robota NAO: T2, T14, H21 a H25*	*5*
Obrázek 5:	*Schéma software NAO robota*	*5*
Obrázek 6:	*Pohyblivost robota NAO*	*9*
Obrázek 7:	*Robot jako zdravotní sestra*	*13*
Obrázek 8:	*Robot jako pomocník a opora*	*14*
Obrázek 9:	*Boxy programu aplikace*	*15*
Obrázek 10:	*Vzhled robota*	*16*
Obrázek 11:	*Kartičky na detekci*	*17*
Obrázek 12:	*Správné umístění detekční kartičky*	*18*
Obrázek 13:	*Schéma boxů a funkcí programu*	*19*
Obrázek 14:	*Box Data_face reco.*	*20*
Obrázek 15:	*Box Data_visual reco.*	*20*
Obrázek 16:	*Osy pro obraz kamery robota*	*22*
Obrázek 17:	*Hraniční body kartičky a výsledné uvažované souřadnice …*	*23*
Obrázek 18:	*Panel Video monitor, učení nového objektu*	*30*
Obrázek 19:	*Choreographe – Timeline editor I.*	*31*
Obrázek 20:	*Choreographe – Timeline editor II.*	*32*
Obrázek 21:	*Choreographe – Timeline editor III.*	*33*
Obrázek 22:	*Robot na Ortopedii II. LF UK*	*35*
Obrázek 23:	*Korelace jednotlivých otázek*	*41*
Obrázek 24:	*Fungovat, či nefungovat, robot Asterix*	*65*

Seznam grafů

Graf 1:	*Pohlaví dotazovaných*	*37*
Graf 2:	*Věkové rozložení*	*37*
Graf 3:	*Graf pro jednotlivá pohlaví v závislosti na věku*	*37*
Graf 4:	*Graf pro otázku:"Věříte, že za 100 let nahradí roboti sestry …*	*39*
Graf 5:	*Graf pro otázku "Jste spokojen(a) s léčbou a ošetřením …*	*39*
Graf 6:	*Graf pro otázku " Probíhá Vaše léčba zde déle než půl roku?"*	*40*
Graf 7:	*Graf pro stáří a dobu léčby dotazovaných osob*	*42*
Graf 8:	*Graf pro pohlaví a víru v budoucnost s roboty*	*43*
Graf 9, 10:	*Grafy pro jednotlivá pohlaví a víru v budoucnost s roboty*	*43*
Graf 11:	*Graf pro stáří a víru v budoucnost s roboty*	*44*
Graf 12, 13:	*Grafy pro rozdílný věk a víru v budoucnost s roboty*	*44*

1. Úvod

Robot NAO je malý humanoidní robot, který umožňuje interakci se člověkem. Robot je vybaven audiovizuálním příslušenstvím. Jedná se o model, se kterým pracuje projekt HUMAVIPS.

První část práce se zabývá robotem NAO, popisuje fyzické vybavení, ale také softwarové. Jsou zde ukázány různé verze tohoto robota, stejně tak různé verze jeho těla. Dále jsou uvedeny programovací jazyky, které robot podporuje.

Následující část popisuje projekt HUMAVIPS, který se zabývá interakcí s lidmi a tím i pohybem robota v neznámém prostředí. Představuje cíle tohoto projektu a podmínky vymezující vhodný základ pro tvorbu aplikace a výběr vhodné robotické platformy.

V souladu s tímto projektem jsem si zvolila aplikaci pro robota NAO. Robot dělá průzkum mezi lidmi a získává odpovědi na sérii otázek. Z těchto odpovědí je v poslední části práce vypracovaná studie zpracovávající výsledky dotazníku a popis chování účastníků ankety.

Robot se ptá na pět otázek, první dvě nahrazují knihovny pro rozpoznávání atributů lidské tváře. Další tři jsem zvolila s důrazem na prostředí vybrané pro experiment, neopomenula jsem ani otázku zaměřenou na vizi budoucnosti s využitím robotů.

Vypracovaná studie popisuje rozdělení dotázaných podle věku a pohlaví, ale také podle odpovědí na jednotlivé otázky. Je popsáno chování, reakce a dotazy kladené účastníky. Výsledkem je představení vhodného vzorce chování robota a uvedení jeho předností a nedostatků.

Práce je doplněná podrobnou programátorskou dokumentací obsahující jak popis základního chodu aplikace, tak detailní popis jednotlivých funkcí programu.

2. Robot NAO

Robot NAO je vyráběný francouzskou společností Aldebaran, z jejíž dokumentace jsou čerpány následující informace.

Je to humanoidní, autonomní robot, který umožňuje interakci s člověkem nebo jiným robotem NAO. Největší využití nachází na univerzitách v akademickém prostředí, kde je využíváno velkého spektra jeho funkcí. V budoucnu se předpokládá využití robota i v dalších oblastech mimo robotiku, například ve zdravotnictví. Zde by robot mohl pomáhat při léčbě autistických dětí.

Obrázek 1: Robot NAO

Snadné programování NAO zajišťuje program Choreographe dodávaný společně s robotem. Tento program je vhodný i pro začátečníky, protože má snadné ovládání. Stačí zde pouze propojovat "boxy" z defaultní knihovny. Samotné programování zajišťujeme především pomocí jazyků C++ a Python, dalšími podporovanými jazyky jsou .Net, Java, Matlab a Urbi.

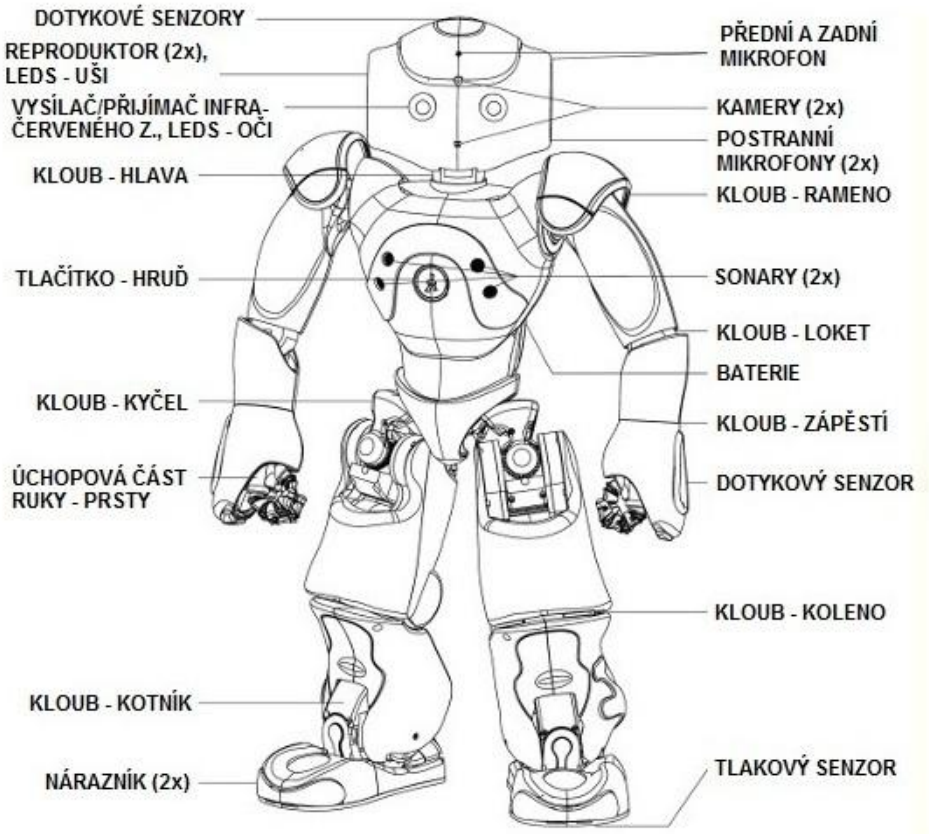
2.1. Popis robota

Robot NAO je vysoký 58 cm, váží 5 kg a jeho tělo má 25 stupňů volnosti.

Na temeni hlavy se nachází tři dotykové senzory. Níže, na čele a temeni, jsou umístěné mikrofony. Na temeni je také umístěn konektor pro připojení ethernetového kabelu. Kamery leží v místě obličeje, kde se také nachází uvnitř očí vysílač a přijímač infračerveného záření. Uši robota jsou znázorněny modrým podsvícením, uvnitř nich jsou umístěny reproduktory. Podsvícení očí a uší je realizováno LED diodami.

Uprostřed hrudníku robota je umístěno tlačítko určené pro zapínání a vypínání, které je z vnějšku podsvíceno. Barva podsvícení signalizuje stav nabití baterie. Na prsou jsou po obou stranách umístěny sonary. Záda robota jsou určená pouze pro baterii, tam je také umístěn konektor pro připojení nabíječky.

Další dotykové senzory se nachází na hřbetech rukou a chodidlech. Klouby na rukou, na nohou, ale i kloub na krku jsou tvořeny elektrickými motory. Ty umožňují rychlost chůze až 14 cm/s. Robota lze připojit k počítači jak ethernetovým kabelem, tak pomocí WiFi.



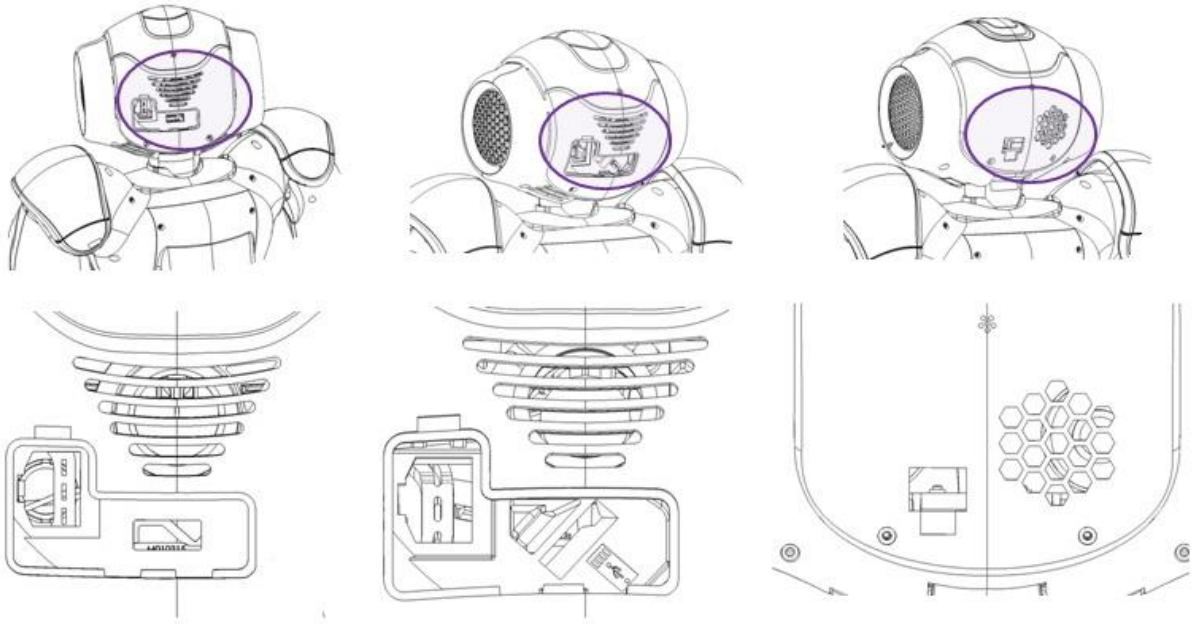
Obrázek 2: Popis robota NAO

2.2. Modely robota

Stejně jako každý výrobek, i robot NAO se neustále vyvíjí a objevují se jeho vylepšené verze.

2.2.1. Verze hlav robota NAO

Tyto verze se liší rozdílným designem temena hlavy. Uvedu například čtyři verze: NAO V4, V3.3, V3+ a V3.2, které mají shodnou zadní část hlavy.

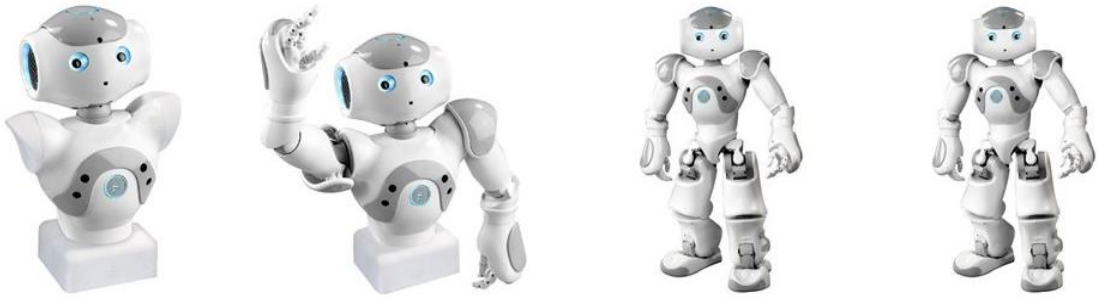


Obrázek 3: Verze robota NAO: V4, V3.3, V3+ a V3.2

2.2.2. Typy těl robota NAO

Tělo robota se také vyrábí v různých typech: NAO T2, T14, H21 a H25. Typ T2 je pouze hlava umístěná na těle bez končetin, typ T14 má k tělu přidané navíc ruce. Tělo typu H21 se využívá především pro robotickou soutěž RoboCup [1]. Oproti typu H25 mu chybí pohyblivost zápěstí a prstů a dotykový senzor na rukou.

[1] http://www.aldebaran-robotics.com/en/Solutions/For-Robocup/the-NAO-league.html

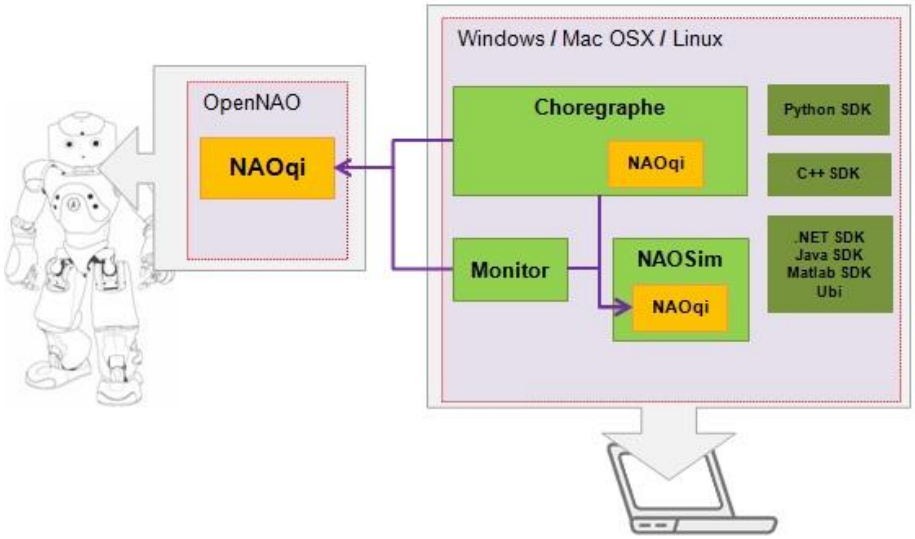


Obrázek 4: Typy těla robota NAO: T2, T14, H21 a H25

ČVUT – FEL vlastní hlavy robota NAO typu V4 a V3.2. Těla pro robota jsou k dispozici také dvě. Typ T2 a H25. Následné zpracování praktické části bylo provedeno na hlavě typu V3.2 a tělu T2, protože zapůjčení těla typu H25 domů nebylo možné. Časová náročnost vývoje aplikace byla výrazná, proto bylo nutné zvolit torzo.

2.3. Software

Software dodávaný s robotem zahrnuje především čtyři programy – NAOqi, Choreographe, Monitor a NAO Sim. NAOqi kontroluje správný chod robota, programování robota zajišťuje především program Choreographe, kde využíváme programovatelných boxů. Pro vidění pohledem robota využíváme program Monitor a pro simulaci pohybů NAO Sim.



Obrázek 5: Schéma software NAO robota

2.3.1. NAOqi

NAOqi je základní částí software, jeho hlavní funkcí je zajišťovat rozhraní pro softwarovou komunikaci s robotem. Spouští se při startu, monitoruje robota a především kontroluje chyby v kódu programu.

2.3.2. Choreographe

Choreographe je program zajišťující prostředí pro programování robota. Nabízí mnoho předem definovaných funkcí a knihoven. Defaultní knihovna nabízí boxy, které umožňují chůzi, mluvení či rozpoznávání objektů a tváří. Tyto boxy lze nastavovat, přepisovat kód a vzájemně je propojovat.

Tento program dále umožňuje definovat sled pohybů pomocí Timeline editoru a naučit robota rozpoznávat objekty, což umožňuje Video monitor. Snadno můžeme povolit nebo ztuhnout motory nebo měnit hlasitost řeči a zvuků robota.

Po připojení je typ těla robota automaticky rozpoznán a objeví se v okně zobrazující robota a jeho polohu. Po spuštění programu kopíruje pohyby NAO robota.

2.3.3. Monitor

Program Monitor umožňuje vidět pohledem kamery robota. Jeho funkce nabízí přepínání mezi horní a spodní kamerou, zapnout a vizualizovat rozpoznávání tváří a naučených objektů. Také umožňuje rozpoznávat speciální značky určené pro robota- NAO marks.

2.3.4. NAO Sim

NAO Sim simuluje pohyby robota, které obsahuje spuštěný program. Simulace probíhá i bez připojení robota.

3. Projekt HUMAVIPS

(anglicky Humanoids with auditory and visual abilities in populated spaces)

Humanoidní roboti s audiovizuálními schopnostmi v zalidněných prostorách, zkráceně HUMAVIPS je projekt zaměřený na humanoidní roboty s audiovizuálním vybavením. Očekává se interakce robota s jedním, či více lidmi v neznámých prostorách.

Francouzská společnost Aldebaran robotics vyrábí humanoidní roboty NAO, kteří se využívají v projektu HUMAVIPS. Ve Francii na projektu pracuje také Národní institut pro výzkum - INRIA. Další evropské země, které se projektu účastní, jsou Česká republika, Švýcarsko a Německo. Za Českou republiku je to ČVUT FEL, za Švýcarsko Institut pro výzkum IDIAP a za Německo Univerzita Bielefeld.

3.1. Popis projektu

Cílem projektu HUMAVIPS je interakce, průzkum a rozpoznávání robota. Interakce probíhá ve skupině lidí, kde se od robota očekává vhodné a co nejvíce lidské chování. Důraz je kladen především na samotnou komunikaci mezi humanoidním robotem a člověkem.

Roboti, na které se tento projekt vztahuje, jsou plně programovatelní. A komunikace se člověkem je založena pouze na využití audiovizuálních schopností.

Audiovizuálně vybavený robot by měl komunikovat s lidmi v neznámých prostorách. Měl by je sám vyhledávat ke komunikaci, nebo se zapojit do rozhovoru více lidí. Očekává se, že jeho sociální dovednosti budou naprosto přirozené a budou napodobovat lidské. To znamená, že po zorientování se a nalezení lidí sleduje jejich gestikulaci, dokáže určit správný vzorec chování a zapojit se do dialogu s nimi.

3.2. Projekt a jeho cíle

Pojetí robotů se postupem času mění. Dříve bývali především v uzavřených prostorách fabrik, v posledních letech ale často sdílejí pracovní prostředí s lidmi. Bezpečně a přesně vykonávají jejich požadavky.

Vybavení robotů může být velmi pestré, od speciálních senzorů, přes kamery až k mikrofonům, motory jsou samozřejmostí. Toto vybavení nabízí otázku zapojení robota do komunikace a spolupráce s lidmi.

Robot se díky kameře, sonarům nebo i dotykovým senzorům dokáže zorientovat v místnosti a bezproblémově pohybovat, komunikační schopnosti robota ale stále nejsou dostatečně pokročilé.

Podmínky pro komunikaci je důležité omezit na interakci robot – člověk, a brát v úvahu chování a zvyky, které jsou vhodné pro současnou dobu a společnost. Tento projekt je zaměřen především na interakci se skupinou lidí v neformální společnosti.

Robot by se ve společnosti měl umět pohybovat – bez kolize kolem lidí procházet a identifikovat jejich činnost. Tomuto chování by měl porozumět zpracováním informace z jejich gestikulace, pozice hlavy a těla a verbálního projevu. Jeho chování by mělo být velmi podobné, sám by měl také gestikulovat a udržovat vhodnou polohu těla vzhledem k danému tématu a situaci.

Chování robota při dialogu lze modelovat pomocí sledování rysů neverbální komunikace, vizualizace, sémantiky nebo určitých signálů. Během rozhovoru by například mohl pomocí gest rukou ukazovat na právě mluvícího člověka, či nově příchozího.

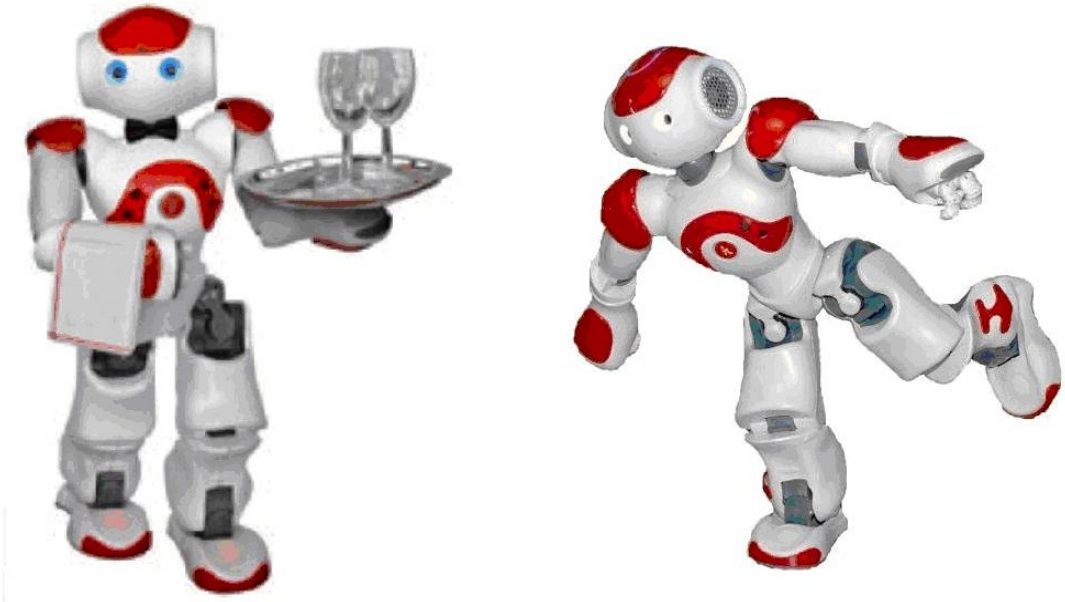
Interakce robota ve skupině lidí je výrazně náročnější než při komunikaci robot – člověk. Dá se říct, že vývoj v tomto odvětví je stále v počátku. Projekt HUMAVIPS se zaobírá otázkami založenými na audiovizuálním získávání informací, se kterými robot dále pracuje.

3.2.1. Otázky projektu HUMAVIPS

Projekt HUMAVIPS se zaobírá především otázkami: Jaké je správné rozpoznávání a naučení objektů, které robot audiovizuálně identifikuje? Které audiovizuální podněty dokáže robot rozlišit v rušném prostředí, například s více lidmi a více objekty v místnosti? Jaké mají osoby v místnosti umístění a kolik jich tam je? Jak vhodně interagovat s lidmi, aby komunikace probíhala v reálném čase? Na tyto otázky se tento projekt také snaží najít odpovědi.

Aby byl projekt úspěšný, je důležité důsledně vybrat správnou autonomní robotickou platformu. Klademe důraz především na:

• Pohyby robota by měli kopírovat lidské pohyby. Zejména u složitých a náročnějších pohybů bychom měli dávat pozor, aby je robot byl schopný vykonat. Požadovaná pohyblivost robota ale zaleží především na našem očekávání činnosti od robota. Robot by měl být schopen chůze, natáčení těla, pohybu hlavou například při sledování osoby, se kterou komunikuje, ale také gestikulace.



Obrázek 6: Pohyblivost robota NAO

• Robot je autonomní platforma. Všechny senzory, se kterými pracuje, jsou umístěny na něm. To je ideální pro interakci ve skupině lidí. Projekt HUMAVIPS se chystá toto nadále rozvíjet. Robot by měl být schopen optimalizovat audiovizuální vybavení na daný cíl - jím vybraného člověka a následně přesunout pozornost na jiného i v jiném zorném poli. Toto znemožňuje využití vzdálených senzorů, například dalších kamer a mikrofonů.

• Reaktivní chování se zajišťuje pomocí audiovizuálního sledování lidského chování. Zpracování tohoto chování vyžaduje optimalizaci a přesnost při rozhodování vyhodnocení. Z důvodu velké zátěže se pro výpočet využívají nejen jeho vlastní procesory, ale i vzdálené za pomoci moderní komunikační techniky.

• Chování a reakce robota musí být pochopitelné i pro člověka. Robot je naprogramovaný, ale lidské chování vytváří mozek. Robot očekává od člověka předem definované, nebo naučené vzorce chování, a z nich vybere ten nejbližší známý. Naopak člověk ale může spekulovat, co bylo záměrem chování robota. Vzájemně se chování od sebe velmi liší a cílem je, ho co nejvíce sobě přiblížit a usnadnit vzájemnou komunikaci. Interpretace robota by se měla podobat lidské reakci.

4. Aplikace pro interakci robota NAO s lidmi

Tato kapitola realizuje dílčí část z popisu komunikace robota s lidmi v předchozí kapitole.

Robot NAO za pomoci otázek provádí průzkum mezi lidmi, zapamatuje si odpovědi a případně známou tvář. Dotazník zpracuji a vytvořím grafy závislostí určitých položených otázek. Následně také vytvořím studii představující možný vhodný vzorec chování robota ve zvoleném prostředí.

Aplikace je vytvořená jako univerzální dotazník, z toho důvodu je použitelná pro průzkum veřejného mínění různého druhu.

4.1. Popis aplikace

Cílem aplikace je rozpoznat obličej člověka před kamerou a detekovat jeho odpovědi na určité otázky. Tato detekce se opakuje pro více osob.

K této aplikaci měl být připojen modul z projektu HUMAVIPS, který rozpoznává různé atributy lidské tváře, jako je odhad věku a pohlaví. Tento modul nahradí první dvě položené otázky, díky kterým zjistím pohlaví a částečně věk. Výsledkem bude vypracovaná studie na závislosti odpovědí na věku či pohlaví a odhad vhodného vzorce chování pro robota ve zvoleném prostředí.

Výsledek této studie by mohl do budoucna nabízet usnadnění komunikace robota s člověkem. Při navázání na tuto aplikaci by robot znal podmínky pro vhodný vzorec chování, který by mohl uplatnit v obdobné situaci.

4.2. Zvolené prostředí a otázky

Studie byla provedena na pacientech ve Fakultní nemocnici v Motole na Ortopedii II. LF UK. Toto prostředí jsem zvolila z důvodu velkého potenciálu využití robotů v budoucnosti ve zdravotnickém prostředí.

4.2.1. Základní otázky

Tyto otázky nahrazují modul, který měl z detekovaného obličeje rozpoznat pohlaví a odhadnout věk osoby. Z časových důvodů mi nebyl tento modul poskytnut, proto ho první dva dotazy z ankety nahrazují.

První dvě základní otázky se tedy týkají pohlaví a věku. Jelikož robot přijímá pouze odpovědi ANO – NE, musí být otázka stručná. "Jste žena?" Znalost odpovědi na tuto otázku přiblíží komunikaci robota na více lidskou, například v českém jazyce by dokázal položit otázku ve správném rodě a využít vhodného oslovení, jako je "paní" či "pan".

Odhad věku je také velmi důležitý, komunikace s dítětem, dospělým člověkem, ale i staršími lidmi se liší. Dítěti může robot tykat a komunikovat prostřednictvím hry, naopak dospělý člověk ocení seriozní a věcné vystupování. Na starší lidi roboti často působí nedůvěryhodně, proto se od robota očekává vstřícné vystupování a často hlasitější projev.

Ambulance ortopedie II. LF UK je určena pro dospělé pacienty, proto jsem jako druhou otázku, otázku na věk, zvolila "Je Vám méně než 50 let?".

4.2.2. Otázka budoucnosti

Následující otázky se týkají prostředí, ve kterém se robot bude pohybovat. NAO bude umístěn v čekárně Ortopedie II. LF UK největší české nemocnice. Třetí otázka se bude týkat budoucnosti, která by plně využívala humanoidních robotů v praxi. Pokud by k tomuto velkolepému a významnému vývoji došlo, FN v Motole by pravděpodobně byla jednou z prvních nemocnic v České republice. Jak vidí budoucnost pacienti? "Věříte, že za 100 let nahradí roboti sestry nebo lékaře?"

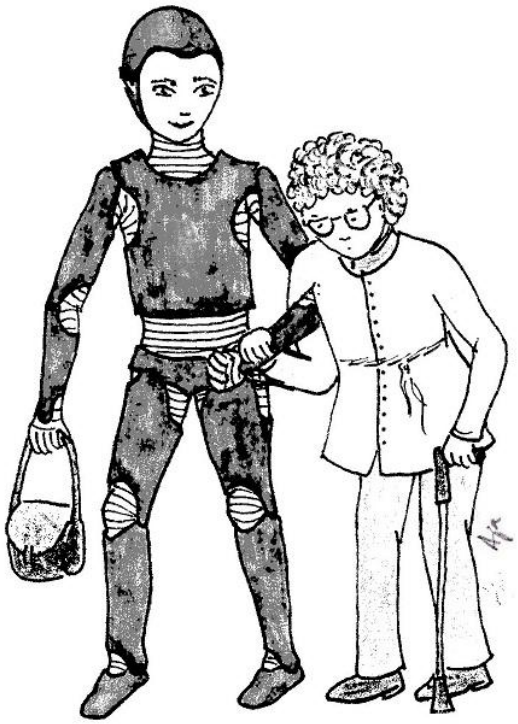
Obrázek 7: Robot jako zdravotní sestra

4.2.3. Robot na Ortopedii

Další otázky jsou zaměřeny konkrétně na zvolenou kliniku. Znalost odpovědí na následující otázky by vzorci chování robota měla dodat empatii. Čtvrtá otázka je zaměřená na kvalitu péče. "Jste spokojen(a) s léčbou a ošetřením na Ortopedii II. LF UK?"

Nejen že by robot zjistil, zdali průběh léčby na klinice je dobrý či vyhovující, ale také by odhadl pravděpodobnou náladu pacienta. Pacient, který odpoví pozitivně, bude pravděpodobně v dobré náladě, naopak pacient, který odpověděl "ne" by mohl být podrážděný a nervózní. Pokud by byla odpověď na tuto otázku negativní, měl by se robot dále vyptávat na důvod a získat co nejvíce informací, případně se snažit zlepšit náladu pacienta. Zpracováním všech získaných informací by robot mohl nabídnout porozumění a nabídnout řešení, tím i zlepšit kvalitu péče. Poslední otázka, na kterou se robot bude ptát, často vypovídá o mobilitě pacienta. "Probíhá Vaše léčba zde déle než půl roku?" Robot by díky této znalosti lépe odhadl zdravotní stav pacienta a odhadl zhoršenou pohyblivost. Sám by také zpomalil tempo chůze, případně čekal déle na reakci člověka, se kterým komunikuje.

Tato otázka by následně vyžadovala další, podrobnější otázky. Robot by podrobněji zjistil zdravotní stav a z něj určil nejlepší způsob komunikace a pomoci. Robot o silné a stabilní tělesné konstrukci by mohl například sloužit jako opora a doprovod.



Obrázek 8: Robot jako pomocník a opora

4.2.4. Shrnutí

Všechny otázky byly vybrány tak, aby jejich následné zpracování dalo možnost vytvořit co nejvhodnější vzorec pro chování robota ve zvolené situaci. Pro shrnutí otázky zopakuji:

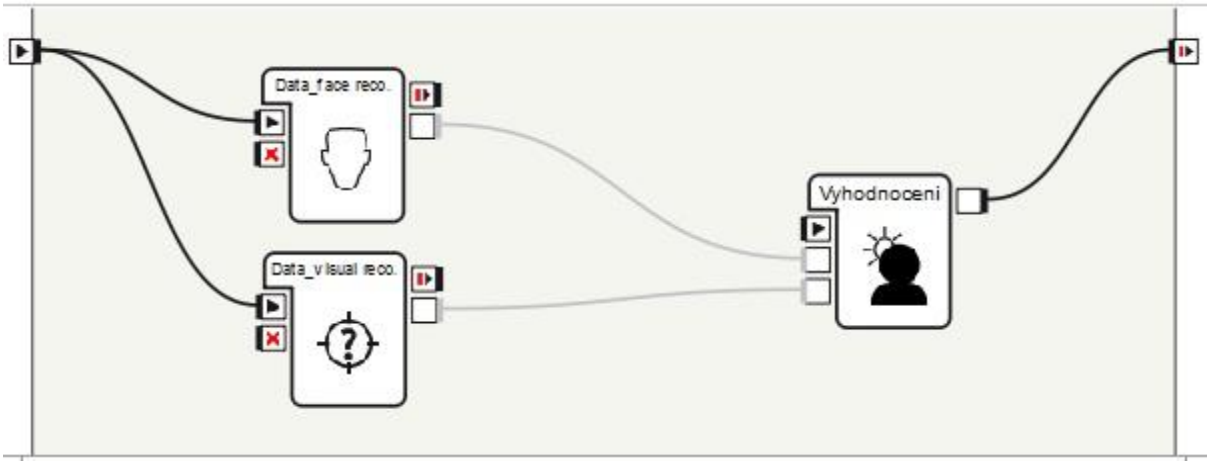
1. Jste žena?
2. Je Vám méně než 50 let?
3. Věříte, že za 100 let nahradí roboti sestry nebo lékaře?
4. Jste spokojen(a) s léčbou a ošetřením na Ortopedii II. LF UK?
5. Probíhá Vaše léčba zde déle než půl roku?

4.3. Popis programu aplikace

Aplikace je rozdělena na tři hlavní části - boxy. První dva *Data_face reco.* a *Data_visual reco.* pouze získávají data z detekce kamery robota. Box *Data_face reco.* získává data o detekovaném obličeji, které okamžitě předává do boxu *Vyhodnocení.* Box *Data_visual reco.* pracuje na stejném principu, jako posledně zmíněný, ale předává data o detekovaném objektu, tj. odpovědi.

Nejobsáhlejší box *Vyhodnocení* zpracovává přijaté informace. Výsledek zpracování je název tváře a odpovědi na určený počet otázek. Pokud robot rozpozná známou tvář, zapamatuje si její jméno, pokud nerozpozná, jako název tváře vrátí prázdný řetězec. Zpracování se opakuje pro další osoby a jejich dotazníkové odpovědi.

Následující schéma ukazuje souvislost mezi jednotlivými boxy.



Obrázek 9: Boxy programu aplikace

4.3.1. Box *Vyhodnocení*

Funkce uvnitř boxu *Vyhodnocení* vytváří běh celého programu. Jsou zde tři hlavní funkce. Funkce *def onInput_inputFace(self, a)*, která přijímá data z boxu *Data_face reco.* a funkce *def onInput_inputObject(self, b)*, která přijímá data z boxu *Data_visual reco*. Obě funkce data zpracují za využití dalších funkcí a předají je k finálnímu zpracování do funkce *def starting(self)*. Ta jejich výsledky spojí, vytvoří pole obsahující název tváře a detekované odpovědi a uloží je do globální proměnné. Uvnitř této proměnné jsou uložené informace o všech

předchozích osobách, které se zúčastnily dotazníku.

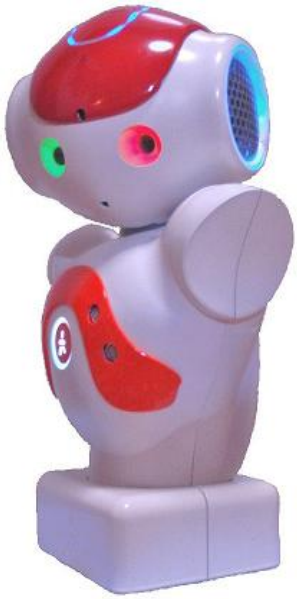
Všechny funkce jsou ve schématu na straně 19 zkráceně popsané a jsou tam znázorněny jejich vzájemné souvislosti. V následující kapitole *5. Dokumentace programu* jsou všechny funkce důkladně popsané.

4.3.2. Rozpoznávání známého obličeje

Abych využila možnost rozpoznání známého obličeje, musím požadovanou tvář robota nejprve naučit. Naučení tváře robotem zajišťuje funkce *self.faceDetectionModule.learnFace("Jméno")* a naopak ke smazání databáze jmen tváří, které robot zná, využiji funkce *self.faceDetectionModule.clearDatabase()*. Tyto funkce k programu přidávám pouze, pokud učení vyžaduji.

4.3.3. Vzhled robota

K oživení vzhledu robota se během různých fází běhu programu využívá změny barvy očí, mluvení robota a kývaní hlavou na znamení úspěšně zvládnutého dotazníku. Mluvený projev robota probíhá v anglickém jazyce z důvodu špatné výslovnosti při mluvení v českém jazyce.



Obrázek 10: Vzhled robota

4.3.4. Detaily a omezení aplikace

Tato aplikace byla navržena do hlučného prostředí, kde nelze využít detekce hlasu a tím i snadného určení odpovědi. Detekce hlasu je u robota NAO nespolehlivá, ve hlučném prostřední majoritně nefunkční. Proto jsem zvolila k získání odpovědí detekci objektů, tj. odpovědí pomocí kartiček.

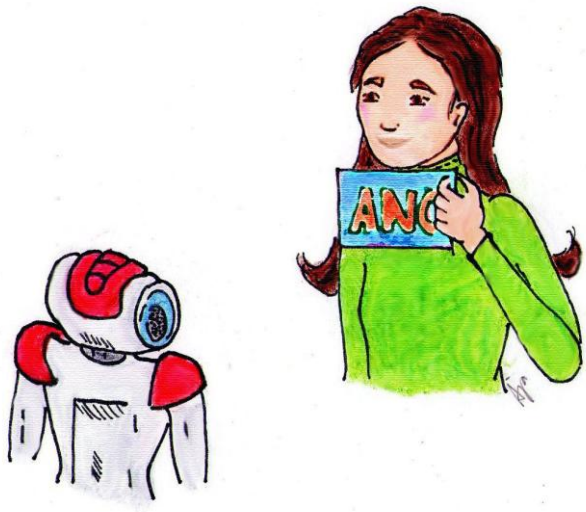
Jako odpovědi v aplikaci nabízím pouze variantu ANO – NE. Pro obě odpovědi využívám speciálně vytvořené obrázkové kartičky, které se robot naučí a po spuštění aplikace je dokáže detekovat.

Program nabízí také vedlejší funkce jako je možnost pozastavit běh programu pomocí kartičky "wait" a opětovného spuštění pomocí kartičky "go", dále smazaní poslední detekce pomocí kartičky "delete" a ukončení běhu programu kartičkou "stop".

Jako detekční kartičky jsem zvolila nepříliš vzhledné obaly od semínek zeleniny z důvodu kvality detekce. Experimentálním ověřením jsem je vyhodnotila jako nejspolehlivější objekty k detekci.

Obrázek 11: Kartičky na detekci

Aplikace je vytvořená s předpokladem pouze jediné osoby, tj. jedné tváře před kamerou robota z důvodu nespolehlivé detekce většího počtu objektů, tj. odpovědí. Vhodné počáteční nastavení polohy hlavy robota je, že poloha obličeje detekovaných osob se nachází uprostřed obrazu snímaného kamerou robota. Toto je z důvodu rozdílné výšky detekovaných osob, program už dále zajistí vhodné natočení hlavy robota. Předpokládané ideální umístění kartičky s odpovědí je asi 20 až 30 cm od obličeje ve výšce těsně pod bradou.



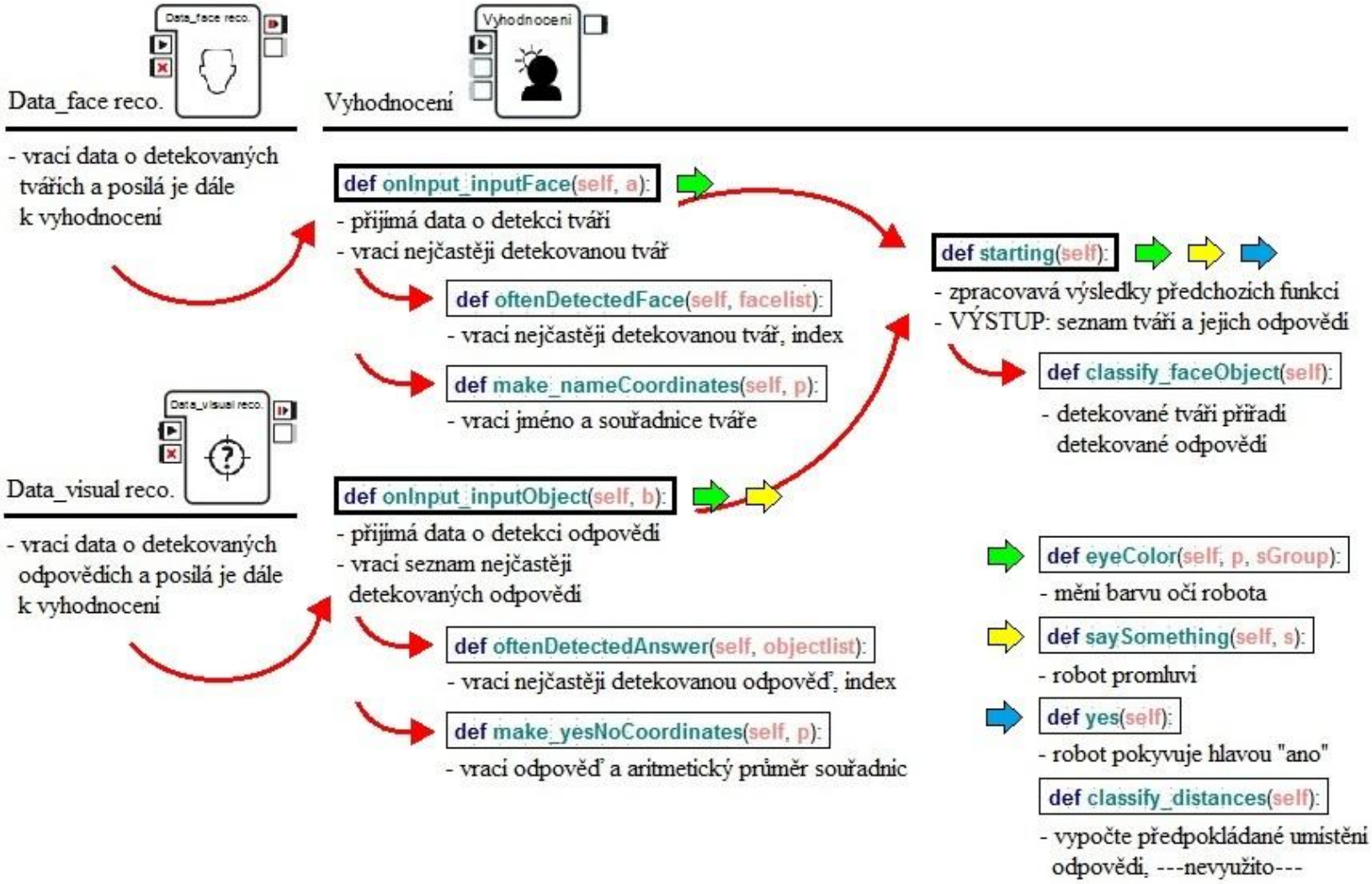
Obrázek 12: Správné umístění detekční kartičky

Při ukončení aplikace občas nastává chyba v podobě doběhu programu. Místo aby se činnost robota ukončila, uvnitř robota program stále dobíhá. Projevuje se to například dodatečnou změnou barvy očí nebo mluvením. Tuto chybu se mi nepodařilo odstranit, protože příčina není známa.

Aplikace byla vytvořena v prostředí Choregraphe 1.12 v jazyce Python.

4.3.5. Schéma funkcí

Následující schéma zobrazuje jednotlivé boxy a především funkce uvnitř boxu *Vyhodnocení*. Propojení jednotlivých funkcí znázorňují šipky. Červené šipky jsou pro funkce programu, zelené pro změnu barvy očí, žluté pro mluvení robota a modré pro kývání hlavou robota.



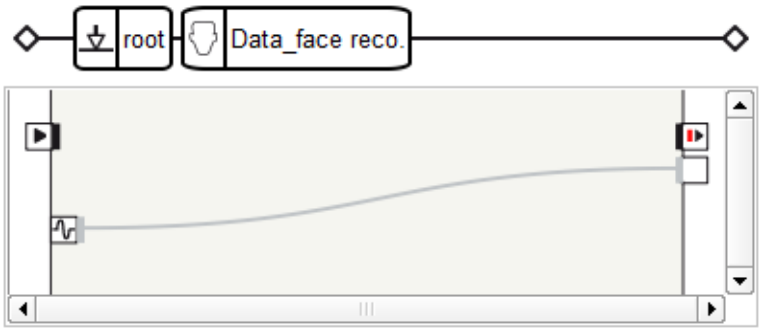
Obrázek 13: Schéma boxů a funkcí programu

5. Dokumentace programu

Signál spuštění programu zaktivní boxy *Data_face reco.* a *Data_visual reco.*, které začnou přijímat signál z detekcí, ten dále přeposílají do boxu *Vyhodnocení*. Tam dochází k finálnímu zpracování. Výsledkem je seznam tváří a jejich detekovaných odpovědí na položené otázky.

5.1. Box *Data_face reco.*

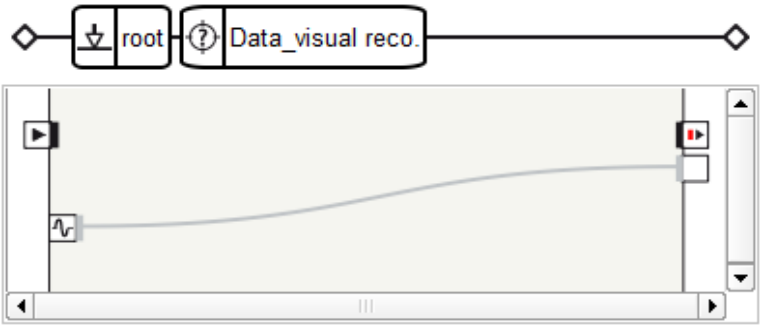
Tento box získává data o detekované tváři a okamžitě je předává dál do boxu *Vyhodnocení*.



Obrázek 14: Box Data_face reco.

5.2. Box *Data_visual reco.*

Box *Data_visual reco.* pracuje na stejném principu jako předchozí. Získává data o detekovaném objektu a okamžitě je předává dál do stejného boxu.



Obrázek 15: Box Data_visual reco.

5.3. Box *Vyhodnocení*

Tento box přijímá data z předchozích boxů, zpracovává je pomocí následujících funkcí. Výsledkem je seznam detekovaných jmen a jejich odpovědí na otázky z dotazníku.

5.3.1. Funkce *def __init__(self):*

Tato první funkce definuje pouze využívané moduly a globálně využívané proměnné. Moduly jsou určené pro další požadované možnosti funkcí robota. V mém případě jsou to především funkce, jejichž využití je vidět navenek a robot se zdá živější a zajímavější. Nejvíce využívaný je ale detekční modul *self.faceDetectionModule = ALProxy("ALFaceDetection")*. Ten umožňuje detekovat tváře, které robot vidí kamerou.

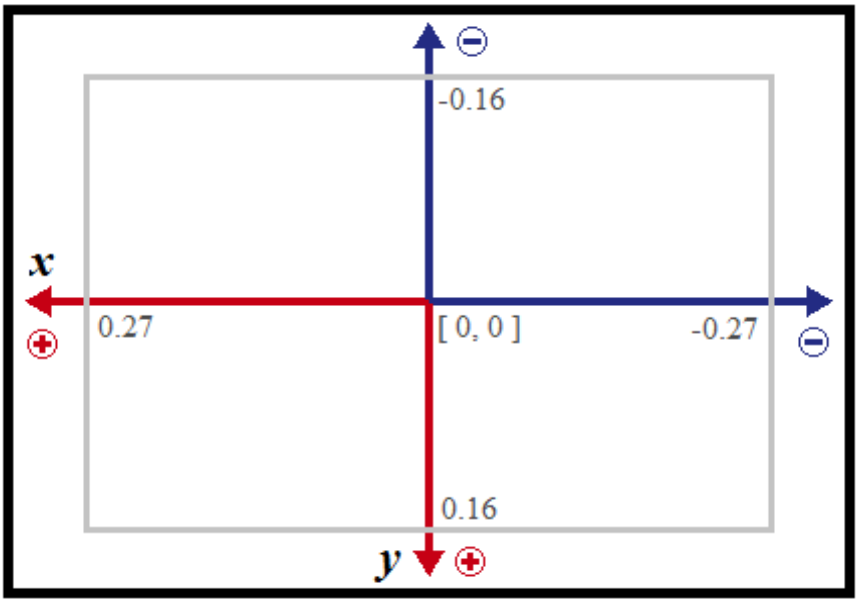
Následující dva moduly, jak již bylo předem zmíněno, jsou důležité pro vzhled robota. První modul *self.motion = ALProxy("ALMotion")* zajišťuje pohyblivost robota. Z důvodu vytvoření aplikace na těle robota NAO typu T2, neboli torzu těla, lze využít pouze pohybů hlavou.

Druhý modul umožňuje robotovi mluvit. Pomocí tohoto modulu *self.tts = ALProxy('ALTextToSpeech')* komunikuje robot s okolím.

Proměnné definované uvnitř této funkce jsou globálně využívané mezi různými funkcemi. V některých jsou uložena důležitá data, například o detekovaném objektu nebo tváři, jiné slouží pouze jako pomocné – značené *self.pomX*, kde X nahrazuje další část názvu proměnné.

Do proměnných *self.head_yaw=0* a *self.head_pitch*=0 se při spuštění programu uloží původní nastavení polohy hlavy. Do této polohy se hlava robota vrací vždy při detekci pro další osobu.

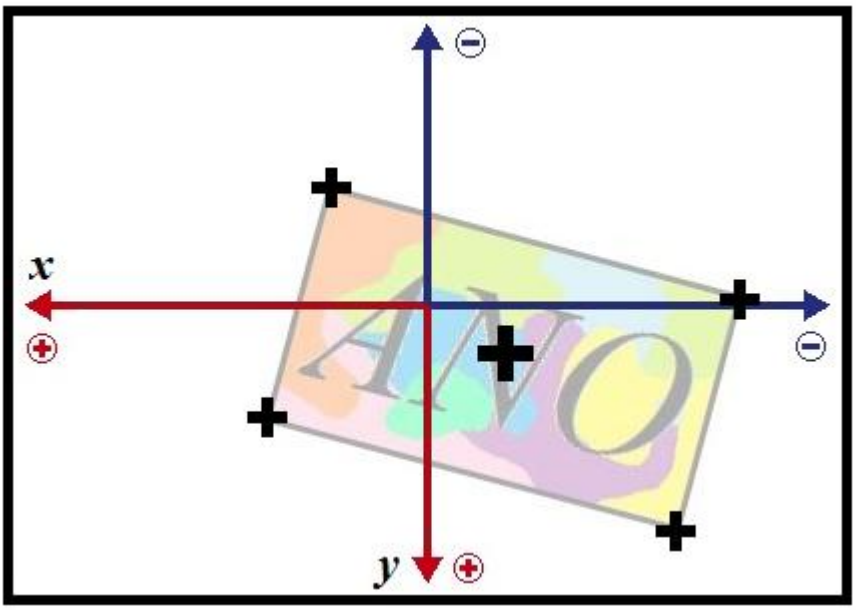
Následující dvě proměnné nesou informace o detekované tváři a objektech neboli odpovědích. Do pole proměnné *self.coordinatesFace=[]* se na první místo uloží název detekované tváře, na následující místa ve zmíněném pořadí: souřadnice umístění středu detekovaného obličeje na ose vodorovné ose *x*, umístění na svislé ose *y*, souřadnice pro levé oko na ose *x* a pro pravé oko na ose *x*. Pravá a levá strana oka je určená vzhledem k obrazu kamery.



Obrázek 16: Osy pro obraz kamery robota

Uvedené souřadnice měly být původně využity pro přiřazení odpovědi, tj. detekované kartičky, k detekované tváři. Z důvodu nekvalitního obrazu kamery a chybných detekcí byla možnost detekce více osob najednou zavržena, proto v této aplikaci uvedené souřadnice nejsou využity.

Do pole proměnné *self.coordinatesObject*=[] se načítají informace o všech detekovaných odpovědích pro jednu osobu. Každá z těchto odpovědí nese po načtení na prvním místě název detekované odpovědi, což znamená “ano“ nebo “ne“. Na dalších dvou místech jsou hodnoty vodorovné souřadnice *x* a svislé *y*, které udávají umístění detekční kartičky na kameře robota. Obě umístění jsou dána aritmetickým průměrem všech souřadnic vymezujících detekční plochu kartičky.



Obrázek 17: Hraniční body kartičky a výsledné uvažované souřadnice x,y kartičky

Tyto souřadnice také nejsou dále využity z výše uvedeného důvodu.

Proměnná *self.namesObject=[]* nese pouze názvy detekovaných odpovědí pro jednu osobu.

Pole proměnné *self.face_answer=[]* nese výsledné informace o jedné osobě a jejích odpovědích, pro ukázku: *['jméno', 'ano/ne', 'ano/ne', ...]*. Počet odpovědí je dán proměnnou *self.number_of_questions*, jejíž hodnota odpovídá počtu otázek.

Do proměnné *self.object_list=[]* se načítají všechny přijaté informace o požadovaném počtu detekovaných objektů, ze kterých se vybere nejčastěji detekovaný. Počet detekovaných objektů je dán hodnotou proměnné *self.number_of_object_detection*. Načítání objektů trvá obvykle delší dobu, proto volím jako ideální počet tři až pět detekcí dle osvětlení místnosti, na kterém je spolehlivost detekce silně závislá. Proměnná *self.object_label_list=[]* nese po načtení pouze názvy, tj. odpovědi.

Proměnná *self.face_list=[]* funguje na stejném principu jako předchozí, s tím rozdílem, že se do ní načítají informace o požadovaném počtu detekovaných tváří, ze kterých se vybere nejčastěji detekovaná. Počet detekovaných tváří je dán hodnotou proměnné *self.number_of_face_detection*. Tato detekce bývá rychlá, proto volím hodnotu kolem deseti. Proměnná *self.face_label_list=[]* nese po načtení pouze jména tváří, pokud tvář nezná, nese prázdné řetězce: *“”*.

Nejdůležitější proměnná je *self.list_of_face_and_answers=[]*. Do té se načítají informace o všech detekovaných lidech a jejich odpovědích, tj. proměnná *self.face_answer*. Tyto informace jsou uloženy v jednotlivých polích. Pro uvedení, výsledný vzhled proměnné: *self.list_of_face_and_answers = [['jméno', 'ano/ne', 'ano/ne', ...], [...], ...]*.

Z pomocných proměnných uvedu pouze *self.pom2*, jejíž hodnota udává pořadí otázky, na kterou robot právě detekuje odpověď.

Proměnné *self.red*, *self.yellow* a další, kde za *self* následuje anglický název barvy, jsou typu pole odpovídajícímu hodnotám RGB. Například: *self.red=[255, 0, 0]*. Tyto proměnné se využívají při nastavování barvy očí. Na startu aplikace jsou obě oči červené.

Červená barva očí značí probíhající detekci, žlutá čekání, modrá a zelená úspěšnou detekci, bílá smazání poslední detekce, tyrkysově modrá přerušení detekce a růžová ukončení programu.

5.3.2. Funkce *def onInput_inputFace(self, a):*

Vstupem do této funkce je proměnná *a*, která obsahuje veškeré informace o detekované tváři. Tato funkce je také jedna ze dvou vstupních, které spouští další funkce.

První podmínka *if self.pomHead==0:* je splněna pouze při spuštění programu. Uvnitř této podmínky se ukládá nastavení polohy hlavy při startu. Do této polohy se hlava vždy vrací při střídání osob.

Následující dvě podmínky jsou splněny, pokud do funkce dorazí signál s daty detekované tváře. Podmínka *if (self.pom1==0):* je splněna pokud nebyla detekovaná kartička "wait", nebo neprobíhá část některé funkce, která si vynutí přerušení detekce.

Nejprve dojde k vhodnému nastavení polohy hlavy robota tak, aby detekovaná tvář byla v horní části kamery a pod ní byl prostor pro detekovanou kartičku s odpovědí. Ze vstupu detekovaných tváří se jich ukládá požadovaný počet, to udává proměnná *self.number_of_face_detection*. Opakovaná detekce je vytvořená z důvodu obtížného rozpoznávání známé tváře. Při více detekcích je šance, že alespoň jednou tvář rozpozná.

Po uložení požadovaného počtu tváří se levé oko, z našeho pohledu, přebarví do zelena a dojde k jejich zpracování pomocí funkce *self.oftenDetectedFace(self.face_label_list)*. Ta z detekovaných tváří vybere tu s nejčastěji detekovaným jménem. Pokud žádnou tvář, kterou bude znát, nerozpozná, vrátí prázdný řetězec a informace o první detekované tváři. Následně

jsou načteny požadované hodnoty do proměnné *self.coordinatesFace*, která se využívá při dalším zpracování.

Na závěr je volána funkce *self.starting()*, ta proběhne pouze, pokud je splněna i detekce objektů – odpovědí.

5.3.3. Funkce *def onInput_inputObject(self, b):*

První podmínka je stejná jako u předchozí funkce. Uložení polohy hlavy proběhne u té funkce, která je spuštěna dříve.

Druhá podmínka je splněna, když do funkce dorazí signál s daty o detekovaném objektu – detekční kartičce. Dále se běh funkce dělí podle detekované kartičky.

Při detekci kartičky “stop“ je ukončen běh programu, robot řekne “stop“ a oči se mu zbarví do růžova.

Kartička “delete“ smaže poslední detekovanou tvář a její odpovědi, robot řekne “delete“ a na krátko přebarví oči do běla. Poslední možnost tyto údaje smazat je ve chvíli, kdy jsou obě detekce dokončené, jejich vyhodnocení zpracováno a oči robota svítí žlutě.

Po vymazání hodnot se oči přebarví do žluta, deset sekund robot čeká a následně je přebarvení do červena. Tím dává najevo, že běží nová detekce. Také zopakuje pořadí detekované osoby a uvede, že se jedná o její o první otázku.

V případě této kartičky není povolena její opakovaná detekce okamžitě po sobě. Zde docházelo k chybě programu, z důvodu jeho zpožděného běhu a tím chybné činnosti. Proto mezi jednotlivým mazáním je nutné alespoň jednou detekovat jakoukoliv jinou kartičku s výjimkou “stop“, která ukončí program.

Detekce kartičky “wait“ pozastaví detekce tváří a odpovědí. Oči se přebarví do tyrkysově modré a robot řekne “wait“. Detekce kartičky “go“ detekci opět spustí. Robot řekne “go“ a vrátí se mu zpět původní barva očí.

Po podmínce, zdali nebyla detekována kartička “wait“, se zpracovávají kartičky vypovídající odpověď. Robot při začátku každé otázky řekne "Question number" a anglicky číslo otázky. Ze vstupu detekovaných odpovědí se jich ukládá požadovaný počet, to udává proměnná *self.number_of_object_detection*. Zde je opakovaná detekce vytvořená, kdyby odpovídající změnil odpověď na otázku. Tato detekce je časově náročnější, proto v případně nedostatku času stačí detekovat pouze jednou.

Po uložení požadovaného počtu odpovědí, se pravé oko, z našeho pohledu, přebarví do modra a robot řekne "OK question number " a anglicky číslo otázky. Následně dojde k jejich zpracování pomocí funkce *self.oftenDetectedAnswer(self.object_label_list)*. Ta z detekovaných odpovědí vybere poslední nejčastěji detekovanou odpověď. Robot odpověď řekne v anglickém jazyce. Následně jsou načteny požadované hodnoty do proměnné *self.coordinatesObject*.

Pokud je počet otázek a tím i odpovědí vetší než jedna, pravé oko se přebarví do žluta, detekce je samozřejmě přerušena, a čeká se pět sekund. Poté se oko přebarví do červena a začíná detekce odpovědi na další otázku. Předchozí popis běhu funkce se opakuje a požadované odpovědi a hodnoty se opět přidávají do proměnné *self.coordinatesObject*, která se po poslední odpovědi využívá k dalšímu zpracování.

Stejně jako v předchozí funkci je na závěr je volána funkce *self.starting()*, která proběhne pouze, pokud je splněna i detekce tváře.

5.3.4. Funkce *def starting(self):*

Tělo této funkce se spustí pouze tehdy, pokud úspěšně proběhly veškeré detekce a zpracování uvnitř funkce *def onInput_inputFace(self, a):* a *def onInput_inputObject(self, b):*. To dá robot najevo vyslovením věty "Questions for person number" anglicky vysloví pořadí osoby, která byla detekována "is done" a souhlasným pokýváním hlavou. Při běhu funkce *def starting(self):* je samozřejmě přerušena detekce tváří a odpovědí.

Zde jsou výsledky předchozích funkcí zpracovány a spojeny do jednoho, který je uložený v proměnné *self.list_of_face_and_answers*. Pro zopakování její tvar vypadá takto: *self.list_of_face_and_answers = [['jméno', 'ano/ne‘, 'ano/ne‘, ...], […], ...]*.

Po úspěšném zpracování se oči přebarví do žluta a nastaví původní poloha hlavy. Pokud je v tuto dobu detekovaná kartička “wait“ a následně opět “go“, robot čeká dvacet sekund. To je z důvodu případné výměny osob a jejich času na přípravu. Obvykle ale robot čeká patnáct sekund, po té se smažou dočasné informace, oči se přebarví do červena a opakuje se detekce pro další osobu.

5.3.5. Funkce *def oftenDetectedFace(self, facelist):*

Tato funkce je volána pouze jednou, jejím vstupem je proměnná *self.face_label_list*. Je to proměnná typu pole obsahující řetězce názvů detekovaných tváří, vypadá například takto *self.face_label_list = ['', '', '', 'jméno', '']*.

Z tohoto řetězce vybere první nejčastěji detekované jméno a jeho index. Pokud nebylo detekováno žádné jméno, vrátí prázdný řetězec a první index.

5.3.6. Funkce *def oftenDetectedAnswer(self, objectlist):*

Tato funkce je také volána pouze jednou a jejím vstupem je proměnná *self.object_label_list*. Je to proměnná typu pole obsahující řetězce názvů detekovaných odpovědí. V tomto případě je to varianta "ano" nebo "ne", vypadá například takto *self.object_label_list = ['ano', 'ano', 'ne', 'ne', ne'']*.

Z tohoto řetězce vybere poslední nejčastěji detekovanou odpověď a její index. Pro uvedený příklad by funkce vrátila *['ne", 4]*.

5.3.7. Funkce *def make_nameCoordinates(self, p):*

Vstupem do této funkce jsou data nejčastěji detekované tváře, z těch vybere požadované informace, které vrací. Ty následně funkce *def onInput_inputFace(self, a):* načte do proměnné *self.coordinatesFace.* Funkce *def make_nameCoordinates(self, p):* tedy zkráceně vrací jméno detekované tváře, souřadnice středu obličeje na vodorovné ose x a svislé ose y a souřadnice na ose x pro levé a pravé oko vzhledem k obrazu kamery.

Jak již bylo dříve zmíněno, tyto souřadnice měli sloužit ke vhodnému přiřazení tváře a detekovaného odpovědi. To ale nebylo možné realizovat vzhledem ke špatné kvalitě obrazu kamery.

5.3.8. Funkce *def make_yesNoCoordinates(self, p):*

Tato funkce funguje na podobném principu jako předchozí. Vstupem jsou data nejčastěji detekované odpovědi a z těch vybere požadované informace, které vrací. Ty následně funkce *def onInput_inputObject(self, b):* přidá do proměnné *self.coordinatesObject*.

Tato funkce ale především vytváří "středový" bod pro detekovanou odpověď. Z hraničních bodů detekční kartičky vytvoří aritmetický průměr, to udělá jak pro horizontální osu *x*, tak pro vertikální osu *y*.

Vrací pole obsahující odpověď, tedy "ano" nebo "ne" a vypočtené souřadnice. Tyto souřadnice, jako všechny ostatní, zůstávají nevyužity, z dříve uvedeného důvodu.

5.3.9. Funkce *def classify_distances(self):*

Funkce *def classify_distances(self):* vypočte předpokládané vhodné umístění odpovědi vzhledem k umístění detekované tváře. Toto umístění se řeší pouze vzhledem k vodorovné ose x, protože svislou osu y většinou plně zabírá detekovaná tvář a pod ní odpověď.

Umístění je dáno pro pravou stranu trojnásobkem vzdálenosti souřadnice pravého oka od středu tváře a pro levou trojnásobkem vzdálenosti souřadnice levého oka od středu tváře.

Tato funkce je nevyužita z dříve uvedených důvodů, v případě využití kvalitnější kamery má potenciál do budoucna.

5.3.10. Funkce *def classify_faceObject(self):*

Je to funkce zajišťující spojení výsledné detekované tváře a výsledných detekovaných odpovědí. Vrací název tváře a detekovaných odpovědí.

5.3.11. Funkce *def eyeColor(self, p, sGroup):*

Funkce *def eyeColor(self, p, sGroup):* mění barvu očí robota, vstupem je RGB odstín barvy a jaké oko bude přebarveno. Tato funkce umožňuje měnit barvu pravého a levého oka nezávisle na sobě.

5.3.12. Funkce *def saySomething(self, s):*

Tato funkce umožňuje robotovi promluvit. Vstupem je řetězec pro mluvený projev robota.

5.3.13. Funkce *def yes(self):*

Funkce *def yes(self):* je vygenerovaná Timeline editorem programu Choreographe a definuje pohyb hlavy robota. Robot souhlasně kývne hlavou.

5.4. Učení databáze objektů

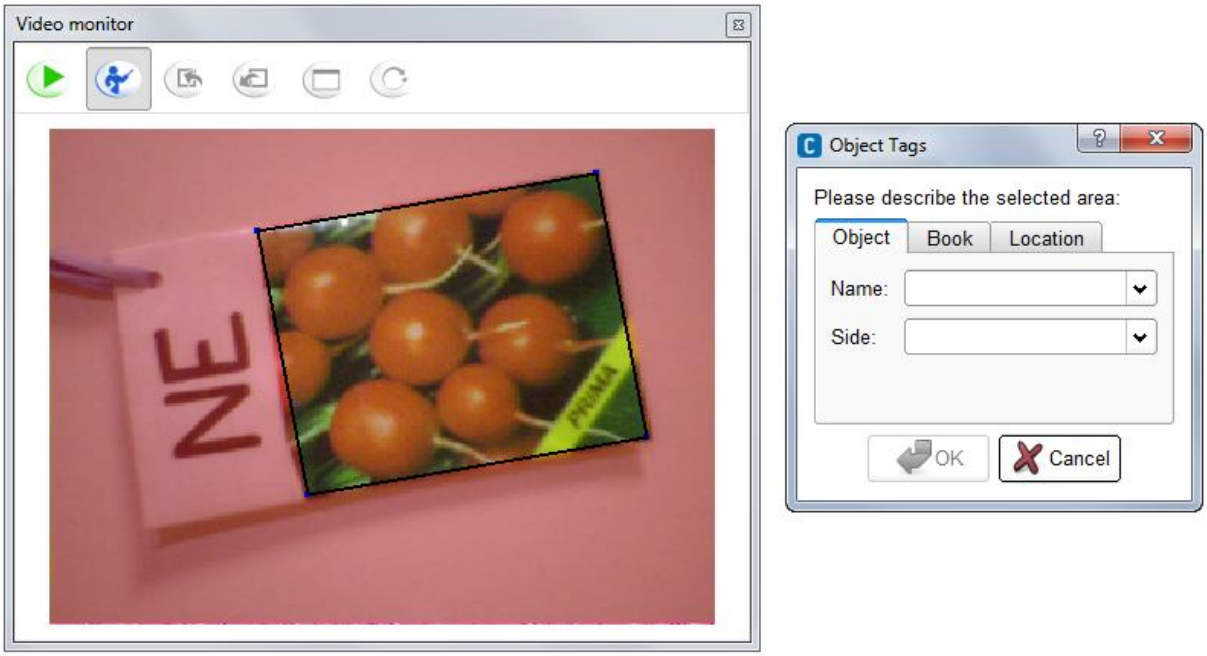
V tomto programu využívám funkce rozpoznávání objektů. Robot si dokáže zapamatovat až deset obrázků, či objektů. K tomu využíváme aplikace *Video monitor* v programu Choreographe.

Panel Video monitor se objeví po kliknutí na *View* a následně *Video monitor*. Na obrazovce je vidět pohledem kamery robota, zde umístím požadovaný objekt na rozpoznávání.

Pro vytvoření nové databáze, kliknu na předposlední tlačítko z leva - *New Vision Recognition Database*. Pro naučení objektu využiji druhého tlačítka zleva - *Learn*. Poté začne odpočítávání, kdy ještě mohu objekt nastavovat do požadované polohy. Následně vyznačím a pojmenuji objekt, který se robot naučí rozpoznávat. Pro více objektů opakuji opět přes tlačítko *Play* a následně *Learn*.

Abych databázi uložila, využiji funkce čtvrtého tlačítka – *Export Vision Recognition Database*. K nahrání databáze do robota slouží poslední tlačítko – *Send current vision recognition database*.

Pokud chci databázi pouze nahrát, kliknu na zbývající tlačítko – *Import Vision Recognition Database* a následně také nahraji databázi do robota tlačítkem *Send current vision recognition database*.



Obrázek 18: Panel Video monitor, učení nového objektu

5.4.1. Zkušenosti a omezení

Naučení objektu je nenáročná část, naopak složitější je naučit se s detekcí správně pracovat.

Problémy způsobuje především nekvalitní obraz kamery, který často znemožní detekci. Pokud detekce neprobíhá, je nutné objekt přiblížit nebo naklápět, aby se změnilo zastínění objektu.

Rozpoznávání objektů je také velmi závislé na osvětlení místnosti. Rozdílné podmínky, především stíny a šero, mohou způsobit, že známý objekt nebude rozpoznán. Toto je velkým problémem v noci, protože běžné osvětlení není dostatečné a světla od stropu vytvářejí stíny. Jediným řešením, jak najít vhodné místo pro práci s robotem, je experimentálně ověřit spolehlivost detekce v daném místě.

Nekvalitní obraz kamery vyřešit nelze, ale v relativně dobře osvětlené místnosti lze kvalitu detekce zlepšit volbou detekovaného objektu. Robot nejlépe detekoval obrázek objektu na jemně lesklém povrchu, který lépe odrážel světlo. Příliš lesklý povrch objektu by ale způsobil přílišný odraz světla, přesvětlení a tím také znemožnil detekci.

Naopak matné povrchy se při umělém osvětlení v noci ukázaly pro detekci nevhodné. Například vytisknutý objekt na běžném bílém papíře světlo pohltil a byl zastíněn.

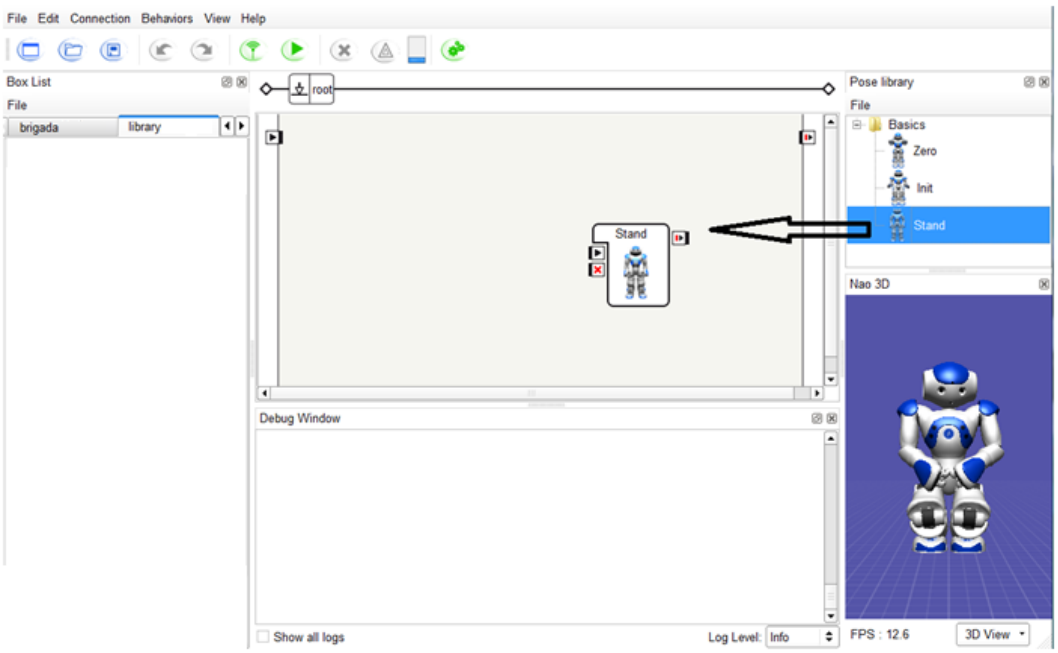
Při denním osvětlení funguje detekce velmi spolehlivě. Oproti detekci tváře ale pomaleji.

5.5. Generování kódu pohybu

Kód pro pohyby hlavou jsem získala vygenerováním kódu pohybu využitím Timeline editoru v programu Choreographe. Následující ilustrace jsou uvedeny pro nastavení pohybu ruky robota.

5.5.1. Návod

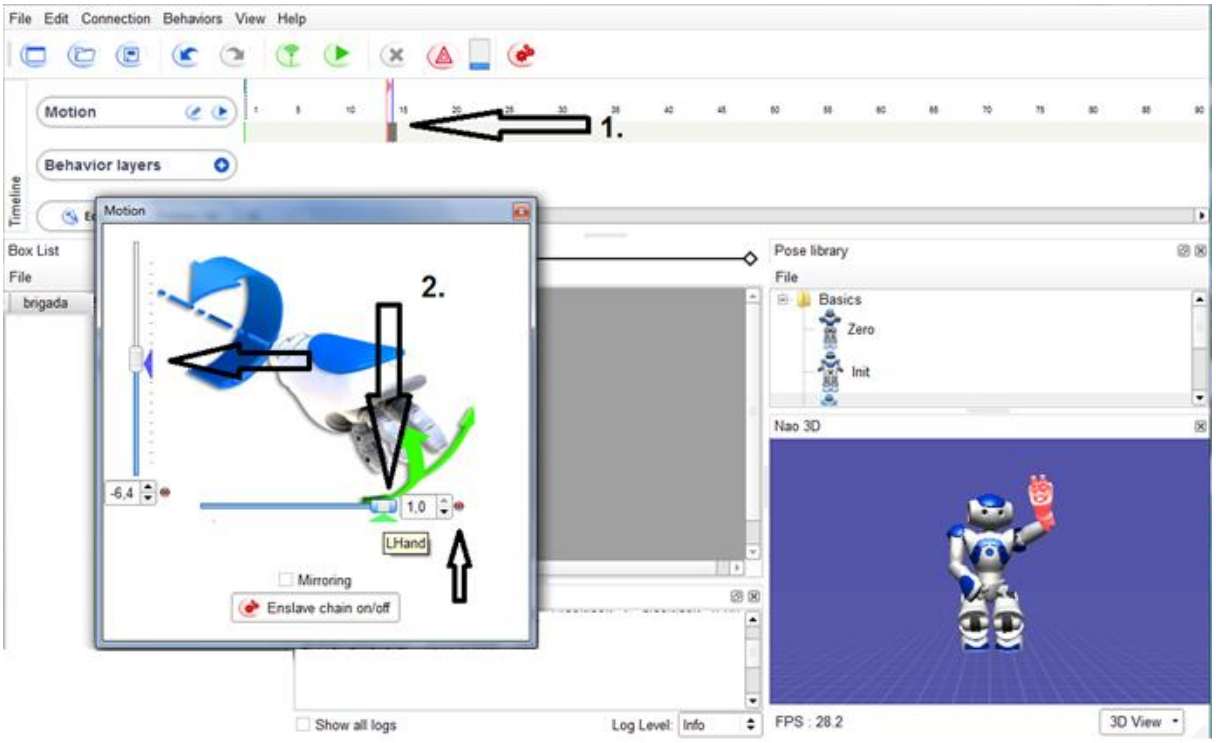
Jednu z defaultních poloh přetáhnu levým tlačítkem myši. Dále dvojklikem levého tlačítka myši rozkliknu libovolně zvolený Box. Tím se otevře Timeline editor.



Obrázek 19: Choreographe – Timeline editor I.

Nejprve na časové ose kliknu na místo, kdy chci, aby se provedl pohyb. Po kliknutí na určitou část modelu robota se zobrazí okno s nabídkou nastavení dané části. V případě typu těla T2 robota NAO lze nastavit pouze polohu hlavy.

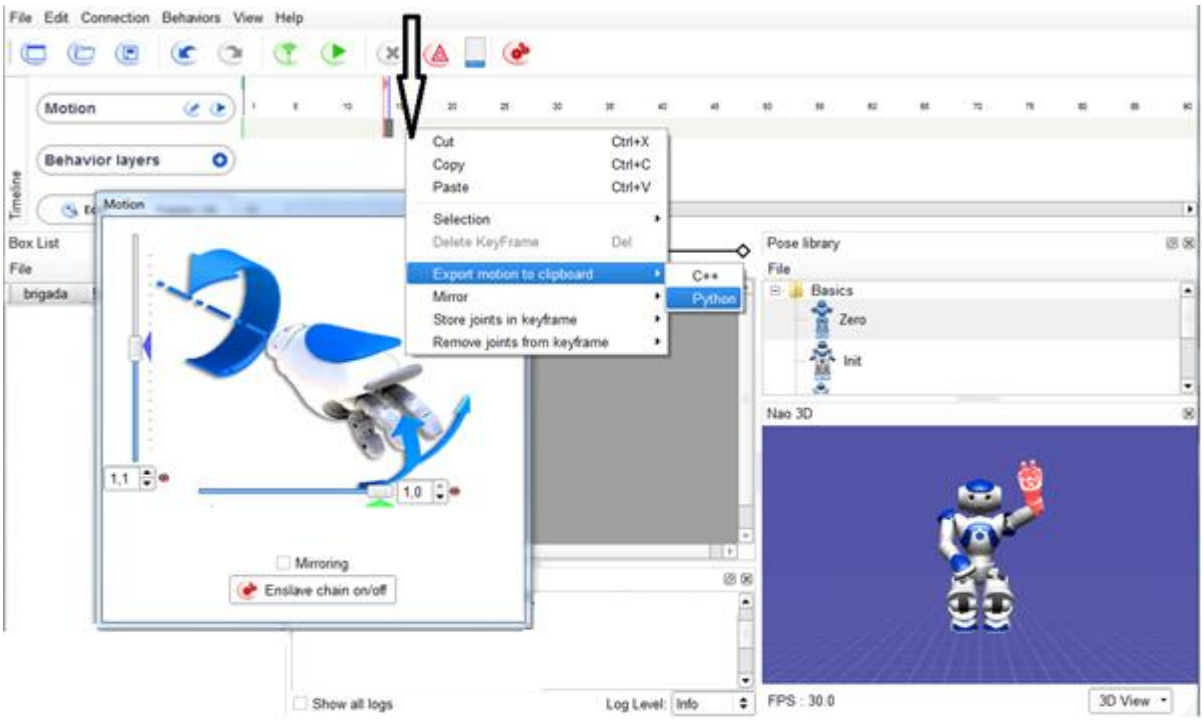
Kliknutím na posuvník, což ukazují dvě větší šipky na následujícím obrázku, nastavím požadovanou polohu. Zajištění polohy poznám podle červeně zbarveného bodu za posuvníkem, to na následujícím obrázku ukazuje malá šipka. Nastavení opakuji pro každou část robota, kterou chci nastavit.



Obrázek 20: Choreographe – Timeline editor II.

Překlikáváním na další místa časové osy mohu nastavit neomezený počet poloh. Kopírování nebo mazání pohybů je též možné.

Po vytvoření požadovaných pohybů vygeneruji kód kliknutím pravého tlačítka myši do časové osy a dále podle následujícího obrázku. Vygenerovaný kód už jen vložím do Box skriptu.



Obrázek 21: Choreographe – Timeline editor III.

6. Aplikace v praxi

Aplikace na provedení průzkumu pomocí robota NAO proběhla na 28 pacientech Ortopedie II. LF UK, dle principů a popisu uvedeném ve 4. kapitole.

Pacienti odpovídali na pět otázek:

1. Jste žena?
2. Je Vám méně než 50 let?
3. Věříte, že za 100 let nahradí roboti sestry nebo lékaře?
4. Jste spokojen(a) s léčbou a ošetřením na Ortopedii II. LF UK?
5. Probíhá Vaše léčba zde déle než půl roku?

Z počátku dne byli pacienti vstřícnější a ochotnější k provedení ankety. Později odmítali komunikovat a byli nervózní a nepříjemní z důvodu dlouhé čekací doby v čekárně.

Mimo otázky robota jsem provedla vlastní zjištění ohledně přístupu lidí k robotovi. Přibližně polovina o robota nejevila zájem a nezajímala se o další informace, jako je různorodá funkčnost a pohyblivost robota. Naopak druhá polovina jím byla zaujatá a měla mnoho otázek.

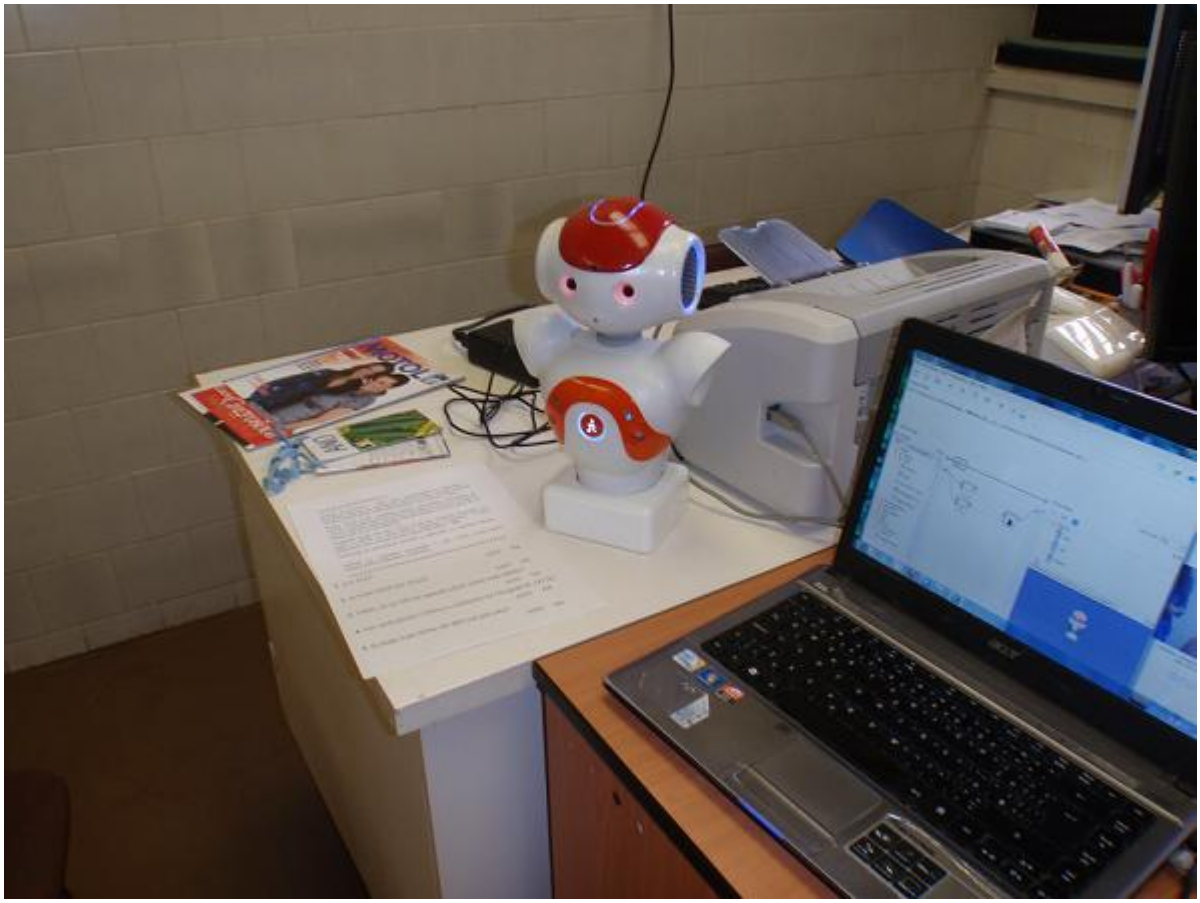
Nejčastější dotazy byly "Co robot umí?" a "To jste postavila Vy nebo u Vás na škole?" Na druhou otázku byla odpověď snadná, ačkoliv mnoho lidí překvapilo, že se robot vyrábí ve Francii a nikoliv v Číně.

Zodpovězení první otázky bylo složitější, protože laická veřejnost s programováním nemá zkušenosti. Předpokládali, že robot má pouze defaultní funkce, se kterými pracuje. To vyžadovalo vysvětlování, že cílem projektu je naprogramovat, tj. naučit, robota novým funkcím.

Přestože pacienty robot zaujal, většinou jejich představa nesplňovala představy cíle projektu HUMAVIPS a považovali ho spíše za "hračku".

6.1. Výsledky ankety

Anketa probíhala 10. května 2012 od 8:00 do 14:30, v době kdy Ortopedii II. LF UK navštívila většina pacientů. Oslovení pacienti se pohybovali od důchodového věku až po jediného dětského pacienta nezávisle na pohlaví.



Obrázek 22: Robot na Ortopedii II. LF UK

Formát následujících uvedených výsledků ankety je upravený pro lepší čitelnost výsledků.

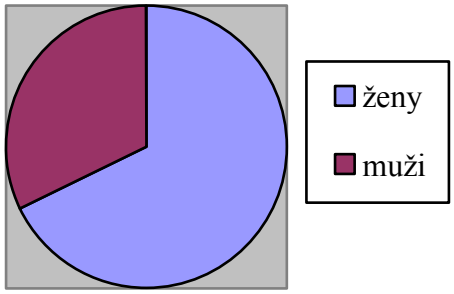
```
1.  [[  '',     'ne',   'ne',   'ne',   'ano',  'ano'  ],
2.   [  '',     'ano',  'ne',   'ano',  'ano',  'ne'   ],
3.   [  'aja',  'ano',  'ano',  'ne',   'ano',  'ne'   ],
4.   [  '',     'ne',   'ano',  'ne',   'ano',  'ne'   ],
5.   [  '',     'ano',  'ano',  'ne',   'ano',  'ano'  ],

6.   [  '',     'ano',  'ne',   'ne',   'ano',  'ano'  ],
7.   [  '',     'ano',  'ne',   'ne',   'ano',  'ano'  ],
8.   [  '',     'ano',  'ne',   'ne',   'ano',  'ano'  ],
9.   [  '',     'ano',  'ne',   'ano',  'ano',  'ano'  ],
10.  [  '',     'ano',  'ne',   'ano',  'ano',  'ano'  ],

11.  [  '',     'ano',  'ne',   'ne',   'ano',  'ne'   ],
12.  [  '',     'ano',  'ne',   'ne',   'ano',  'ne'   ],
13.  [  '',     'ne',   'ano',  'ne',   'ano',  'ano'  ],
14.  [  '',     'ne',   'ne',   'ano',  'ano',  'ano'  ],
15.  [  '',     'ano',  'ano',  'ne',   'ano',  'ano'  ],

16.  [  '',     'ano',  'ano',  'ne',   'ano',  'ano'  ],
17.  [  '',     'ne',   'ano',  'ne',   'ano',  'ne'   ],
18.  [  '',     'ano',  'ne',   'ne',   'ano',  'ano'  ],
19.  [  '',     'ne',   'ano',  'ne',   'ano',  'ano'  ],
20.  [  '',     'ano',  'ano',  'ne',   'ne',   'ne'   ],

21.  [  '',     'ne',   'ano',  'ne',   'ne',   'ne'   ],
22.  [  '',     'ano',  'ano',  'ano',  'ne',   'ano'  ],
23.  [  '',     'ano',  'ne',   'ano',  'ano',  'ne'   ],
24.  [  '',     'ano',  'ne',   'ano',  'ano',  'ano'  ],
25.  [  '',     'ne',   'ne',   'ne',   'ano',  'ano'  ],

26.  [  '',     'ne',   'ne',   'ne',   'ano',  'ne'   ],
27.  [  '',     'ano',  'ano',  'ano',  'ano',  'ano'  ],
28.  [  '',     'ano',  'ano',  'ne',   'ano',  'ne'   ]]
```

Během detekce došlo k jedné, na první pohled zřejmé, chybě. Podle seznamu výsledků dotazníku to vypadá, že detekce třetí série odpovědí byla moje. Moji tvář rozpoznával pod jménem “aja“. Zde ale proběhla chybná detekce tváře.

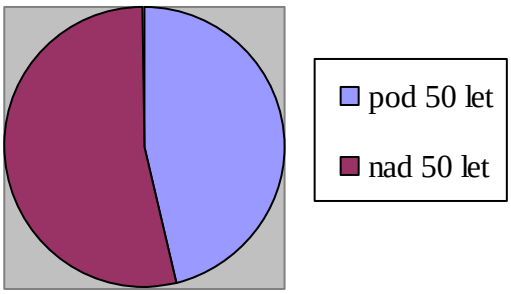
Třetí byla na řadě slečna, která byla stará přibližně jako já, přesto vzhledově velmi odlišná. Její tvář ale robot ve sto procentech případů detekce označil jako mou.

6.2. Základní zpracování ankety – pohlaví a věk

Následující koláčové grafy znázorňují procentuelní zastoupení žen a mužů a stáří pacientů nezávisle na pohlaví.



Graf 1: Pohlaví dotazovaných

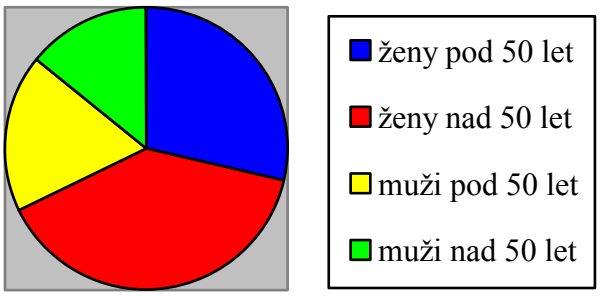


Graf 2: Věkové rozložení

Zastoupení žen v dotazníku bylo přibližně dvou třetinové, to přikládám větší ochotě k provedení průzkumu s robotem. Dotazníku se účastnilo 19 žen, tj. 67,86%, a 9 mužů, tj. 32,14%.

Věkové rozložení je naopak spíše vyrovnané, zájem o interakci s robotem byl nezávislý na věku. Předpokládám, že zaujetí robotem by bylo spíše závislé na vzdělání, případně informovanosti v oboru robotiky. Průzkumu se zúčastnilo 13 osob pod 50 let, tj. 46,43%, a 15 lidí nad 50 let, tj. 53,57%.

Následující graf zobrazuje spojení dvou předchozích. Ukazuje zastoupení jednotlivých pohlaví v závislosti na věku.



Graf 3: Graf pro jednotlivá pohlaví v závislosti na věku

Modrá a červená barva v grafu zastupují většinové procento žen a žlutá a zelená muže. S robotem interagovalo 8 žen pod 50 let, tj. 28,57%, 11 žen nad 50 let, tj. 39,29%, 5 mužů pod 50 let, tj. 17,86% a 4 muži nad 50 let, tj. 14,28%.

Z toho vyplývá, že největší zájem o interakci s robotem projevily ženy nad 50 let a následně ženy pod 50 let. Překvapivě muži měli o provedení dotazníku menší zájem, přestože o techniku obvykle jeví zájem naopak větší. To bych zdůvodnila tím, že muži obvykle méně komunikují.

Věková závislost pro interakci je u obou pohlaví vyrovnaná.

6.3. Zpracování ankety – konkrétní otázky

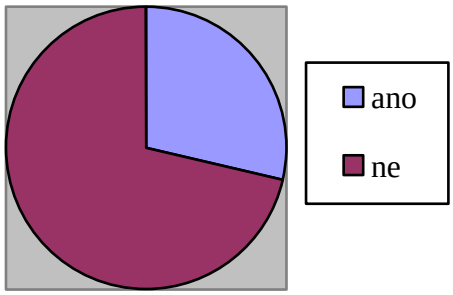
Předchozí otázky pouze nahrazovaly chybějící modul, pomocí nějž měli být odpovědi automaticky rozpoznány. Následující tři otázky se týkají konkrétně dotazníku, na kterém měl robot pracovat.

6.3.1. Budoucnost s roboty

Třetí otázka se týkala víry v budoucnost s roboty. "Věříte, že za 100 let nahradí roboti sestry nebo lékaře?" Více než dvě třetiny dotázaných se k této otázce postavilo negativně.

Vývoj robotů dle následných hodnocení pacientů postupuje pomalu, zatím je považují spíše jako „hračky vědců". Další část pacientů předpokládá „světové problémy", které znemožní vývoj robotů úplně. Dvě třetiny pacientů tedy v robotech ani za velmi dlouhou dobu, jako je sto let, nevidí konkurenci sester a lékařů.

Zbývající třetina vidí naopak budoucnost robotů velmi pozitivně a předpokládá velký vývoj v robotice. Toto odvětví vidí jako významné pro budoucnost a předpokládají pomoc robotů nejen ve zdravotnictví.

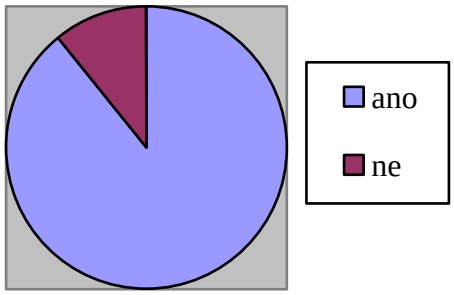


Graf 4: Graf pro otázku:"Věříte, že za 100 let nahradí roboti sestry nebo lékaře?"

Na tuto otázku odpovědělo kladně 8 dotázaných, tj. 28,57%, a 20 dotázaných záporně, tj. 71,43%.

6.3.2. Spokojenost s léčbou a ošetřením

Čtvrtá otázka se zabývá spokojeností pacientů na klinice, kde proběhl dotazník. "Jste spokojen(a) s léčbou a ošetřením na Ortopedii II. LF UK?"



Graf 5: Graf pro otázku "Jste spokojen(a) s léčbou a ošetřením na Ortopedii II. LF UK?"

Zde majoritní část pacientů projevila spokojenost. Nespokojenost u tří pacientů, kteří se zúčastnili dotazníku, byla převážně způsobená dlouhou čekací dobou na ošetření. Z výsledků je také patrné, že nespokojenost se projevila v pozdější hodinu, odhadem kolem poledne.

Pacienti, kteří byli ošetřeni včas, byli ve 100% případů spokojeni. Naopak ti, kteří strávili v čekárně i několik hodin, byli velmi podráždění a odmítali se zúčastnit dotazníku,

přestože nečinně seděli v čekárně.

Kolem jedné hodiny se čekárna začala uvolňovat a opět se zkrátila čekací doba. Dle průběhu dotazníku se i navrátila spokojenost dotázaných.

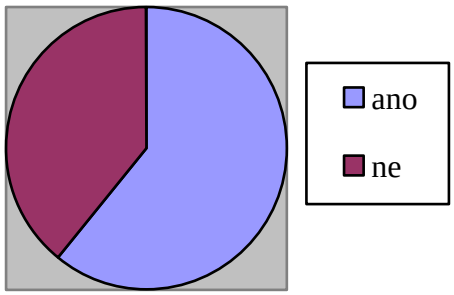
Většina pacientů hodnotí ošetření velmi kladně a i přes obvykle dlouhé čekací doby jsou spokojeni s léčbou a ošetřením.

Spokojenost vyjádřilo 25 dotázaných, tj. 89,29%, a 3 dotázaní, tj. 10,71%, vyjádřili nespokojenost.

Na závěr této kapitoly bych vyjádřila nesouhlas s pacienty, kteří ošetření a léčbu shledali jako špatné, protože tento názor se týkal především doby strávené v čekárně a nikoliv konkrétně ošetření a léčby.

6.3.3. Doba léčby pacientů

Poslední otázka dotazníku zněla: "Probíhá Vaše léčba zde déle než půl roku?".



Graf 6: Graf pro otázku " Probíhá Vaše léčba zde déle než půl roku?"

Zde, ze 28 dotázaných, 17 pacientů potvrdilo dlouhodobou léčbu, tj. 60,71%, a 11, tj. 39,29%, odpovědělo, že se léčí kratší dobu než šest měsíců.

Převaha dlouhodobě léčených pacientů muže být způsobena neochotou krátkodobých pacientů provést dotazník z důvodu dlouhé doby strávené v čekárně. Jak již bylo dříve zmíněno, čekající pacienti byli podrážděni a neteční k dotazníku. Dlouhodobí pacienti jsou zvyklí na dlouhé čekací doby a přivítali interakci s robotem, který naplnil čas v čekárně něčím zajímavým.

6.4. Korelace jednotlivých otázek

Pro celkové vyhodnocení dotazníku jsem zvolila zpracování pomocí programu MATLAB. Zde jsem si nechala vypsat a vykreslit korelaci jednotlivých otázek příkazem *corrcoef([otázka 1, otázka 2, otázka 3, otázka 4, otázka 5])*.

Pozn.: Odpověď ano reprezentovala 1, odpověď ne 0.

	otázka 1	otázka 2	otázka 3	otázka 4	otázka 5
otázka 1	1.0000	-0.1260	0.2660	0.0088	0.0727
otázka 2	-0.1260	1.0000	-0.2717	-0.3721	-0.1309
otázka 3	0.2660	-0.2717	1.0000	-0.0365	0.1850
otázka 4	0.0088	-0.3721	-0.0365	1.0000	0.1942
otázka 5	0.0727	-0.1309	0.1850	0.1942	1.0000

Obrázek 23: Korelace jednotlivých otázek

Pokud je hodnota korelace rovna jedné, otázky spolu v úplné shodě korelují, v tomto případě je ale rovna jedné pouze ve shodujících se otázkách. Na barevném schématu viditelné pod rezavou barvou. Rozdílné otázky spolu příliš nekorelují, tj. nemají souvislost. Záporná hodnota znační opačnou korelaci. To znamená, že pokud na otázku bude odpověď ANO, projevuje se souvislost k odpovědi NE u druhé otázky pro odpovídající hodnotu korelace.

Potvrdila se prakticky nulová souvislost mezi odpovědí na první a čtvrtou otázku. Spokojenost s léčbou a ošetřením je nezávislá na pohlaví. Souvislost první a páté otázky se také očekávala nulová, což hodnota korelace opět potvrzuje. Doba léčby pacienta, případně dlouhodobé léčení, je bez závislosti na pohlaví.

Mezi třetí a čtvrtou otázkou taktéž není očekávána souvislost, to opět potvrzuje hodnota korelace. Víra v budoucnost s roboty nahrazující sestry a lékaře nemá nic společného se spokojeností s ošetřením a léčbou na Ortopedii II. LF UK.

Očekávaná souvislost mezi věkem a délkou léčení na ortopedické klinice se neukázala jako významná. Hodnota korelace je pouze -0,1309, záporná hodnota ukazuje opačnou korelaci. Ta dává nepříliš značně najevo souvislost, že pokud dotazovaná osoba byla mladší 50 let,

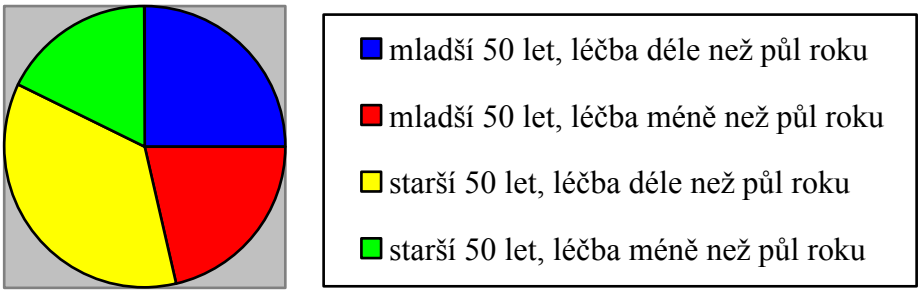
léčí se pravděpodobněji kratší dobu než půl roku a opačně.

Nejvyšší hodnoty dosáhla korelace mezi druhou a čtvrtou otázkou, zde má hodnotu -0,3721. Z této hodnoty vyplývá, že nespokojení s léčbou a ošetřením na Ortopedii II. LF UK jsou především lidé mladší 50 let. Nespokojenost ale vyjádřili pouze tři pacienti.

Pro zaměření této práci jsou nejzajímavější korelace pro třetí otázku "Věříte, že za 100 let nahradí roboti sestry nebo lékaře?" a to především pro pohlaví a věk. Hodnota korelace ukazuje, že ženy věří v budoucnost s využitím robotů více než muži. Mladí lidé překvapivě v tuto budoucnost věří méně, než lidé starší 50 let. Ani jedna z hodnot nebyla příliš vysoká, aby ukázala velký rozdíl představy o budoucnosti v závislosti na pohlaví a věku.

Následující podkapitoly rozebírají důkladněji procentuelní složení odpovědí na jednotlivé otázky.

6.4.1. Stáří a doba léčby pacientů

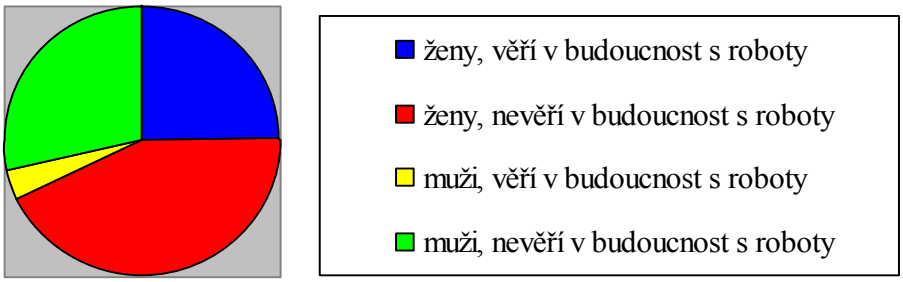


Graf 7: Graf pro stáří a dobu léčby dotazovaných osob

Graf ukazující rozložení dotázaných odpovídá korelaci, ze které slabě vyplývala skutečnost, že pacienti starší 50 let se léčí dlouhodobě.

Mezi dotázanými pacienty bylo 7, tj. 25%, mladších 50 let a zároveň léčících se déle než půl roku, 6, tj. 21,43%, mladších 50 let a zároveň léčících se méně než půl roku, 10, tj. 35,71%, starších 50 let a zároveň léčících se déle než půl roku a 5, tj. 17,86%, starších 50 let a zároveň léčících se méně než půl roku.

6.4.2. Pohlaví a víra v budoucnost s roboty

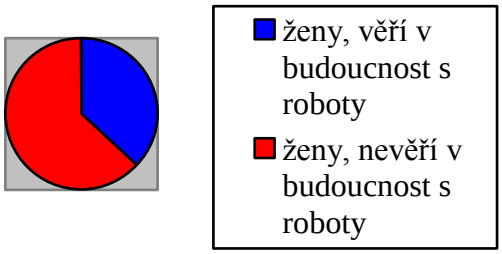


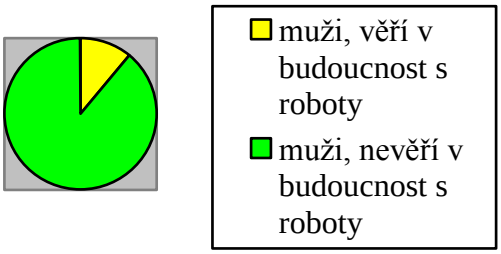
Graf 8: Graf pro pohlaví a víru v budoucnost s roboty

První graf této kapitoly ukazuje celkové rozložení pohlaví a víry v budoucnost s roboty mezi pacienty. 7 žen, tj. 25%, věří v budoucnost s roboty, 12 žen, tj. 42,86%, věří v budoucnost s roboty, 1 muž, tj. 3,57%, věří v budoucnost s roboty a 8 mužů, tj. 28,57%, věří v budoucnost s roboty.

Překvapivý výsledek této ankety je, že pouze jeden muž věří v budoucnost robotů jako sester či lékařů.

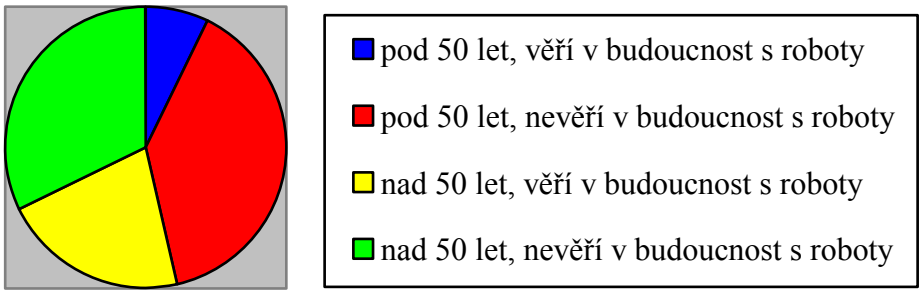
Následující grafy potvrzují hodnotu korelace, která vypovídala, že existuje větší procento žen než mužů, které věří, že roboti za 100 let zastoupí lékaře nebo sestry.





Graf 9, 10: Grafy pro jednotlivá pohlaví a víru v budoucnost s roboty

6.4.3. Stáří a víra v budoucnost s roboty

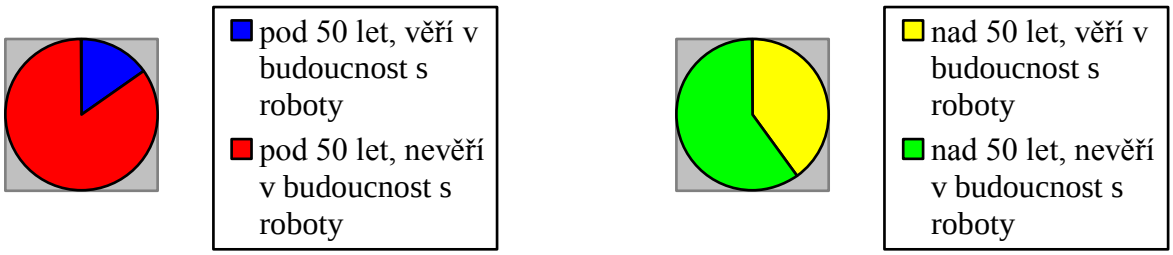


Graf 11: Graf pro stáří a víru v budoucnost s roboty

První graf ukazuje celkové rozložení stáří a víry v budoucnost s roboty mezi pacienty. 2 pacienti mladší 50 let, tj. 7,14%, věří v budoucnost s roboty, 11 pacientů mladších 50 let, tj. 39,29%, nevěří v budoucnost s roboty, 6 pacientů starších 50 let, tj. 21,43%, věří v budoucnost s roboty a 9 pacientů starších 50 let, tj. 32,14%, nevěří v budoucnost s roboty.

Z grafu je patrné že velmi málo mladých lidí věří ve využití robotů jako sester či lékařů.

Následující grafy potvrzují hodnotu korelace, která vypovídala, že existuje větší procento mladších 50 let, než starších 50 let, kteří nevěří, že roboti za 100 let zastoupí lékaře nebo sestry.



Graf 12, 13: Grafy pro rozdílný věk a víru v budoucnost s roboty

6.5. Vzorec chování robota

Ze znalosti detekovaných odpovědí by robot měl určit vhodné chování pro danou osobu, se kterou komunikuje. Tento vzorec chování bude mít robot naprogramovaný, tj. naučený.

V následujících podkapitolách představím možné vzorce chování pro určité odpovědi. Tyto představy již byly částečně zmíněny ve čtvrté kapitole.

6.5.1. Vzorec pro pohlaví a věk

Ze znalosti pohlaví a věku by robot měl přidat do komunikace oslovování, jako například paní nebo pane. Při komunikace v českém jazyce by hovořil k dané osobě ve správném rodě, a při důkladnějším zjištění věku by dětem tykal a dospělým vykal.

Pokud by byl robot kvalitně vybavený komunikačními schopnostmi, mohl by navázat rozhovor na téma pro danou osobu pravděpodobně bližší.

6.5.2. Vzorec podle otázky "Věříte, že za 100 let nahradí roboti sestry nebo lékaře?"

Znalost odpovědi na tuto otázku vypovídá často o důvěře dotazovaného člověka k robotům, vyjadřuje i obecný názor na ně a důvěru v jejich vývoj.

Pokud dotazovaná osoba v tuto budoucnost věří, pravděpodobně bude s robotem komunikovat více otevřeně a se zájmem a bude mít doplňující dotazy.

Lidé, kteří takový vývoj nepředpokládají, často považují robotiku jako nepříliš vyvinuté odvětví, které potřebuje mnohem více času, nebo předpokládají lidskou nenahraditelnost. V tomto případě by se měl robot pokusit vzbudit jejich důvěru, například říci o sobě něco víc.

V případě robota NAO je ale komunikace prostřednictvím rozhovoru složitá ze dvou důvodů. V prvním případě je problém v robotovi, jedná se o nespolehlivou detekci rozpoznávání hlasu. Druhým důvodem problematické komunikace je, že robot neumí hovořit v českém jazyce, protože mnoho lidí z České republiky jiným jazykem nemluví. Tento fakt jsem si ověřila i během dotazníku, kdy robot komunikoval s dotázanými v anglickém jazyce. Na světové úrovni se tento problém dá považovat za zanedbatelný.

Problém je řešitelný, stačilo by rozšířit možnosti syntetizování řeči o podporu českého jazyka.

6.5.3. Vzorec podle spokojenosti pacienta s léčbou a ošetřením

Komunikace se spokojeným pacientem je vždy snazší, dá se u něj předpokládat lepší nálada a ochota ke komunikaci. U těchto pacientů by robot mohl mít více dotazů a více s nimi komunikovat.

Naopak nespokojení pacienti se často brání komunikaci s robotem, protože ho nepovažují za něco, co vyřeší jejich problém. Robot by se v tuto chvíli měl snažit najít problém a ukázat se jako užitečný. Mnoho pacientů by ale dalo přednost ukončení interakce, protože jsou podrážděni a nechtějí se na toto téma dále rozebírat. V tuto chvíli je velmi složité určit správný vzorec pro nadcházející chování.

Vhodné pokračování interakce by měl vyhodnotit z tónu hlasu, gestikulace a případně dalších reakcí, což robot NAO bohužel zatím neumožňuje.

6.5.4. Vzorec podle doby léčby pacienta

Ze znalosti věku a odpovědi na tuto otázku by robot odhadoval možnou zhoršenou pohyblivost pacienta. V ideálním případě by položil další otázky, díky kterým by zjistil konkrétní handicap.

Například pro pacienta, který má problém s chůzí, by také zpomalil tempo chůze. Při komunikaci se starými lidmi by reagoval hlasitěji a pomaleji a naopak při komunikaci s někým mladým, například se zlomenou rukou, by mohl reagovat stejně jako na zdravého člověka.

6.5.5. Vzorec – závěr

Volba vhodného vzorce chování je velmi závislá na vybavení robota a jeho schopnostech. Jedná se o schopnosti jak komunikační, tak pohybové.

Robot NAO je rozměrově velmi malý a jeho chůze velmi pomalá. V pohybové rovině je pro účely ortopedické kliniky nevyužitelný, naopak v komunikační rovině nabízí zajímavou alternativu běžného dotazníku na papíře. Tento dotazník je stejně tak anonymní, ale je vytvořen zábavnou formou.

Vylepšené verze dotazníku by mohly být interaktivní. Nabízely by například doplňující otázky nebo by robot mohl zodpovídat dotazy pacientů.

6.6. Shrnutí ankety

Zúčastnění dotazníku shledali robota jako zajímavou, ale pomalejší alternativu běžné ankety na papír. Většina lidí by přivítala individuální přístup robota, který by udával vhodné tempo komunikace.

Tempo komunikace robota jsem nastavila pomalejší, protože se dotazníku účastnili i starší lidé. Často ale i mladí měli v průběhu dotazy a zvolené tempo interakce se ve většině případu ukázalo jako ideální.

Průběh dotazníku probíhal bez problému, lidé interakci zvládali a na závěr působili zaujatým a nadšeným dojmem.

7. Závěr

Navrhla jsem a implementovala aplikaci realizující komunikaci mezi NAO robotem a lidmi formou dotazníku. Robot byl umístěn na Ortopedii II. LF UK ve FN v Motole, tam pokládal pacientům pět otázek. První dvě nahrazovaly knihovny pro rozpoznávání atributů lidské tváře, protože mi z časových důvodů nebyly poskytnuty. Následující tři otázky jsem zvolila tak, aby jejich zpracování umožnilo zvolit vhodný vzorec chování a také nabídlo zajímavá zjištění o pacientech a jejich názorech nejen na téma robotika.

Prostředí pro programování robota zajišťuje program Choreographe. Aplikace byla vytvořená ve verzi 1.12.

Průzkum proběhl na robotovi s typem těla T2, proto jsem ho nejprve vhodně umístila naproti pacientovi. Když robot zdetekoval lidskou tvář, natočil vhodně hlavu pro snazší detekci kartičky. Pacient pomocí těchto kartiček, které robot detekoval, ukazoval své odpovědi na sérii otázek.

Zpracování odpovědí ukázalo, jaká skupina lidí má o interakci s robotem NAO největší zájem, jak jsou pacientů spokojení s léčbou a ošetřením na Ortopedii II. LF UK, délku jejich léčby, vizi pacientů o budoucnosti s roboty a korelace jednotlivých otázek.

O dotazník formou interakce s robotem projevily zájem především ženy, naopak věkové rozložení bylo velmi vyrovnané. Spokojenost s léčbou a ošetřením vyjádřilo 89,29% dotázaných.

Představila jsem vhodný vzorec chování, podle kterého by se robot choval po provedení jednotlivých otázek dotazníku. Podle znalosti pohlaví by volil vhodná oslovení, například paní nebo pan a podle staří by určil, zdali dotazovanému tykat nebo vykat a zvolil vhodnou hlasitost hovoru.

Otázka týkající se doby léčby, představovala ve spojení s otázkou věku podklad pro odhad mobility pacienta. Z této znalosti by robot mohl například zpomalit tempo chůze.

Přestože méně než 30% dotázaných věří, že za sto let budou humanoidní roboti na lidské úrovni, přeji projektu HUMAVIPS mnoho úspěchů a významných pokroků v tomto oboru.

Zdroje

[1] Aldebaran Robotics Academics forum
http://academics.aldebaran-robotics.com/

[2] Aldebaran Robotics – NAO
http://www.aldebaran-robotics.com/en/
především:

[3] Robot NAO
http://www.aldebaran-robotics.com/en/Pressroom/About/NAO.html

[4] Dokumentace SOFTWARE
http://www.aldebaran-robotics.com/documentation/dev/index.html

[5] Dokumentace SOFTWARE
http://www.aldebaran-robotics.com/documentation/index.html

[6] RobotShop, informace o robotovi NAO typu H25
http://www.robotshop.com/aldebaran-robotics-nao-academic-robot-v3plus-9.html

[7] HUMAVIPS web page
http://humavips.inrialpes.fr/

[8] Příloha projektu:
http://humavips.inrialpes.fr/files/2011/03/HUMAVIPS-annexI-final.pdf

[9] Dokumentace programu Choreographe

[10] Programovací jazyk Python
http://programujte.com/clanky/54-serial-python/1/

Zdroje obrázků

[1] Obrázek 1
http://techiser.com/aldebaran-robotics-nao-h25-humanoid-robot-133578.html

[2] Obrázek 2
http://www.aldebaran-robotics.com/documentation/nao/hardware/product-range.html
přeloženo do českého jazyka

[3] Obrázek 3
http://www.aldebaran-robotics.com/documentation/nao/hardware/product-range.html

[4] Obrázek 4
http://www.robotshop.com/eu/aldebaran-robotics-nao-t2-humanoid-robot.html
http://www.robotshop.com/aldebaran-robotics-nao-t14-humanoid-robot.html
http://www.robotshop.com/aldebaran-robotics-nao-h21-humanoid-robot.html
http://www.robotnik.es/en/products/humanoids/nao-h25

[5] Obrázek 5 Program Choreographe – záložka Help – Choreographe – Software

[6] Obrázek 6
http://files.iai.heig-vd.ch/LaRA/web/p/rahe/2009/NaoWaiterGraz2009.jpg
http://streetwear92.blogspot.com/2012/01/aldebaran-robotics-nao.html
odstraněno pozadí obrázku

[7] Obrázek 7 Alena Petráčková, 2012

[8] Obrázek 8 Alena Petráčková, 2012

[9] Obrázek 9 Program Choreographe 1.12

[10] Obrázek 10 Alena Petráčková, foto robot NAO, 2012
odstraněno pozadí obrázku

[11] Obrázek 11 Alena Petráčková, foto, 2012

[12] Obrázek 12 Alena Petráčková, 2012

[13] Obrázek 13 Alena Petráčková, 2012

[14] Obrázek 14 Program Choreographe 1.12

[15] Obrázek 15 Program Choreographe 1.12

[16] Obrázek 16 Alena Petráčková, 2012

[17] Obrázek 17 Alena Petráčková, 2012

[18]	Obrázek 18	Program Choreographe 1.12
[19]	Obrázek 19	Program Choreographe 1.10.52
[20]	Obrázek 20	Program Choreographe 1.10.52
[21]	Obrázek 21	Program Choreographe 1.10.52
[22]	Obrázek 22	Alena Petráčková, foto Ortopedie II. LF UK, FN v Motole, 2012
[23]	Obrázek 23	Program MATLAB
[24]	Obrázek 24	Alena Petráčková, foto robot NAO, 2012

Příloha A - Program

Program aplikace „dotazník“

Box *Vyhodnocení*

Pozn. 1: V jazyce python se před komentáře vkládá symbol #. Jelikož program Choreographe neovládá český jazyk, jsou i komentáře psané bez diakritiky.

Pozn. 2: Funkce *self.log()* vypisuje do “Debug window“.

```
class MyClass(GeneratedClass):
  def __init__(self):
    GeneratedClass.__init__(self)

    self.faceDetectionModule = ALProxy( "ALFaceDetection" )
    self.motion = ALProxy("ALMotion")
    self.tts = ALProxy('ALTextToSpeech')
    self.ttsStop = ALProxy('ALTextToSpeech', True) #Create another proxy as wait is blocking if audioout is remote

    self.head_yaw=0
    self.head_pitch=0
    self.coordinatesFace=[]
    self.coordinatesObject=[]
    self.namesObject=[]
    self.face_answer=[]
    self.object_list=[]
    self.object_label_list=[]
    self.face_list=[]
    self.face_label_list=[]
    #pom1= 1...pauza, 0... beh detekce
    self.pom1=0
    #pom2 ... 1 az pocet otazek (udava cislo otazky)
    self.pom2=1
    #pomHead ... 0 pro nastaveni idealniho stavu(pocatek),kam se vzdy pro dalsi osobu vraci, 1 jindy
    self.pomHead=0
    #pomStarting ... kdyz probiha funkce starting, je rovna 1, jinak 0
    self.pomStarting=0
    #pomWait ... kdyz detekuje Wait je rovna 1, jinak 0
    self.pomWait=0
```

```
    #self.pomDel ... pokud posledni detekovany objekt delete je rovna 1, jinak 0
    #nulove se opet stane po detekci jineho objektu -> problem je pokud chceme smazat pouze
    nadetekovany oblicej - v tu chvili je nutne detekovat jakoukoliv odpoved a az po té smazat,
    protoze kdyby i pri detekci obliceje se pomDel menila na nulovou mohlo by dochazet
    k opakované detekci delete stridave s oblicejem, tim i chybnemu mazani a chybne zmene
    barvy oci
    self.pomDel=0
    #pokud pomDelStarting = 1 nedobehne zbytek funkce starting (delete uvnitr starting
    detekovat pouze kdyz uz jsou zlute oci)
    self.pomDelStarting=0
    self.number_of_questions=5
    self.number_of_face_detection=5
    self.number_of_object_detection=2
    self.list_of_face_and_answers=[]

    self.ids = []
    self.ledName = "FaceLeds"

    self.pink=[236, 0, 135]
    self.red=[255, 0, 0]
    self.orange=[255, 85, 0]
    self.yellow=[255, 255, 0]
    self.green=[0, 255, 0]
    self.blue=[0, 0, 255]
    self.white=[255, 255, 255]
    self.megenta=[255, 0, 255]
    self.cyan=[0, 255, 255]

    self.eyeColor(self.red, "FaceLeds")



def onLoad(self):
    self.timeFilteredResult = [];
    self.nPicturesDetected = -1;
    self.ids = []
    pass



def onUnload(self):
    for id in self.ids:
        ALLeds.stop(id)
    pass
```

```
    #--------------onInput_inputFace(a)--------------
    #prvni cast funkce:
    #podminka s self.pomHead nastavuje pocatecni polohu hlavy,kterou nastavim, kam se
    pri stridani osob vzdy vrati
    #druha cast funkce:
    #vstupem funkce jsou informace o detekovane tvari
    #ze vstupu detekovanych tvari vybere nejcasteji detekovanou
    #zde úvodni podminky porovnavaji, ze detekce probehla a dorazili pozadovane informace
    o tvari
    #pri prvni detekci kazde osoby se nastavi vhodna poloha hlavy - vidi tvar a pod ni odpoved
    #následně se do self.facet_label_list a self.face_list ukladaji pozadovane informace o nami
    urcenem poctu tvari
    #vyuzite funkce vyberou nejcasteji detekovanou, levé oko(z naseho pohledu) se prebarvi
    do zelena
    #dale se pracuje jen s nejcasteji detekovanou tvari a jejimi informacemi, ktere se posilaji
    do funkce starting
    def onInput_inputFace(self, a):
        if self.pomHead==0:
            self.head_yaw=ALMotion.getAngles("HeadYaw", True)
            self.head_pitch=ALMotion.getAngles("HeadPitch", True)
            self.pomHead=1
            self.saySomething("Person   number   one. Question   number   " + str(self.pom2))
        if(len(a) > 0):
            if(len(a[1]) > 0):
                if (self.pom1==0):
                    #<=xxx ... tj, prida to odpoved 1 navic, tim zajistim ze se nasledujici podminka se
                    nebude opakovat
                    if len(self.face_label_list)<=self.number_of_face_detection:
                        if len(self.face_label_list)==0:
                            alpha=a[1][0][0][1]
                            beta=a[1][0][0][2]
                            ALMotion.changeAngles(["HeadYaw", "HeadPitch"], [float(alpha),
float(beta)+0.12], 0.05)
                        self.face_label_list=self.face_label_list+[a[1][0][1][2]]
                        self.face_list=self.face_list+[a]
                        self.log("IIIIIIIIIIIIIIIIIII poradi detekovane tvare: " +
str(len(self.face_label_list)))
                    if(len(self.face_label_list)==self.number_of_face_detection):
                        self.log("IIIIIIIIIIIIIIIIIII vypis detekovanych tvari: " + str(self.face_label_list))
                        optimal_answer=self.oftenDetectedFace(self.face_label_list)
                        #data tvare se kterymi dale pracuji jsou ulozeny pod stejným indexem
                        v self.object_list jako vraci optimal_answer[1]
                        optimal_answer_info=self.face_list[optimal_answer[1]]
                        self.coordinatesFace=self.make_nameCoordinates(optimal_answer_info)
                        self.eyeColor(self.green, "RightFaceLeds")
                        self.log("IIIIIIIIIIIIIIIIIII ---spousteni starting z detekce tvare---")
                        self.starting()
```

```
#--------------onInput_inputObject(b)--------------
#prvni podminka - jako u predchozi funkce - tam kam dorazi signál drive se provede
#druha podminka udava, ze detekce probehla a dorazili pozadovane informace o objektu
#dalsi podminky jsou pro funkcnost- detekce objektu stop - pro ukonceni programu, delete -
pro smazani posledni detekce, wait - pro pozastaveni a go -pro opetny beh programu
#dochazi zde ke zmene barvy oci a mluveni
#v následujici casti funkce probiha samotná funkce pracujici s detekovanými odpovedmi:
#ze vstupu detekovanych objektu vybere nejcasteji detekovany
#vstupem jsou informace o detekovanem objektu
#do self.object_label_list a self.object_list se ukladaji pozadovane informace o nami urcenem
poctu objektu,
#vybere nejcasteji detekovany a dale se pracuje jen s jeho informacemi - ty posila do funkce
starting
#pro vetsi pocet otazek, tim i odpovedi probiha cast teto funkce vickrát, zaroven dochází
ke zmene barvy oci a mluveni robota
#vzdy po nacteni odpovedi se zmeni barva praveho oka(z naseho podledu) na modrou,
pri cekani na dalsi odpoved na zlutou a pri dalsi
#detekci opet na cervenou
def onInput_inputObject(self, b):
    if self.pomHead==0:
        self.head_yaw=ALMotion.getAngles("HeadYaw", True)
        self.head_pitch=ALMotion.getAngles("HeadPitch", True)
        self.pomHead=1
        self.saySomething("Person number one. Question number " + str(self.pom2))
    if(len(b) > 1):
        if (str(b[1][0][0][0])=="stop"):
            self.saySomething("Stop")
            self.log("IIIIIIIIIIIIIIIIII objekt - detekovano: " + str(b[1][0][0]))
            self.pomDel=0
            self.eyeColor(self.pink, "FaceLeds")
            self.outputStop()
        elif (str(b[1][0][0][0])=="delete"):
            if (self.pomDel==0):
                self.pomDel=1
                self.saySomething("Delete")
                self.log("IIIIIIIIIIIIIIIIII objekt - detekovano: " + str(b[1][0][0]))
                self.eyeColor(self.white, "FaceLeds")
                self.pom1=1
                if self.pomStarting==1:
                    self.pomDelStarting=1
                    del self.list_of_face_and_answers[len(self.list_of_face_and_answers)-1]
                time.sleep(3)
                self.pom2=1
                self.coordinatesFace=[]
                self.coordinatesObject=[]
                self.namesObject=[]
                self.face_answer=[]
                self.object_list=[]
                self.object_label_list=[]
```

```
                self.face_list=[]
                self.face_label_list=[]
                self.eyeColor(self.yellow, "FaceLeds")
                self.log("IIIIIIIIIIIIIIIIIII ---delete---          celkovy detekovany seznam: " +
str(self.list_of_face_and_answers))
                time.sleep(10)
                self.saySomething("Person   number   " + str(len(self.list_of_face_and_answers)+1)
+ " Question   number   " + str(self.pom2))
                self.eyeColor(self.red, "FaceLeds")
                self.pomStarting=0
                self.pom1=0
            elif (str(b[1][0][0][0])=="wait"):
                self.saySomething("Wait")
                self.log("IIIIIIIIIIIIIIIIIII objekt - detekovano: " + str(b[1][0][0]))
                self.pomDel=0
                self.eyeColor(self.cyan, "FaceLeds")
                self.pom1=1
                self.pomWait=1
            elif (str(b[1][0][0][0])=="go"):
                self.saySomething("Go")
                self.log("IIIIIIIIIIIIIIIIIII objekt - detekovano: " + str(b[1][0][0]))
                self.pomDel=0
                if(len(self.face_label_list)==0)and(len(self.object_label_list)==0):
                    self.eyeColor(self.yellow, "FaceLeds")
                    time.sleep(5)
                    self.eyeColor(self.red, "FaceLeds")
                elif(len(self.face_label_list)>=self.number_of_face_detection) and
(len(self.object_label_list)>=self.number_of_object_detection):
                    self.eyeColor(self.yellow, "FaceLeds")
                else:
                    if(len(self.face_label_list)>=self.number_of_face_detection):
                        self.eyeColor(self.green, "RightFaceLeds")
                    else:
                        self.eyeColor(self.red, "RightFaceLeds")
                    if len(self.object_label_list)>=self.number_of_object_detection:
                        self.eyeColor(self.blue, "LeftFaceLeds")
                    else:
                        self.eyeColor(self.red, "LeftFaceLeds")
                self.pom1=0
                self.pomWait=0
            elif (self.pom1==0):
                self.pomDel=0
                #<=xxx ... tj, prida to odpoved 1 navic, tim zajistim ze se nasledujici podminka
                se nebude opakovat
                if len(self.object_label_list)<=self.number_of_object_detection:
                    self.object_label_list=self.object_label_list+b[1][0][0]
                    self.object_list=self.object_list+[b]
                    self.log("IIIIIIIIIIIIIIIIIII otazka " +str(self.pom2) + ", poradi detekovaneho
objektu: " + str(len(self.object_label_list)))
```

```
            if (len(self.object_label_list)==self.number_of_object_detection) and
(self.pomStarting==0):
                self.log("IIIIIIIIIIIIIIIIII vypis detekovanych objektu: " + str(self.object_label_list)
+ ", otazka cislo: " +str(self.pom2))
                self.saySomething("OK  question  number  " + str(self.pom2))
                self.eyeColor(self.blue, "LeftFaceLeds")
                if self.pom2<self.number_of_questions:
                   self.pom1=1
                   self.eyeColor(self.blue, "LeftFaceLeds")
                optimal_answer=self.oftenDetectedAnswer(self.object_label_list)
                if (str(optimal_answer[0])=="ano"):
                   self.saySomething("Answer  is  yes")
                else:
                   self.saySomething("Answer  is  no")
                #data odpovedi se kterymi dale pracuji jsou ulozeny pod stejným indexem
                v self.object_list jako vraci optimal_answer[1]
                optimal_answer_info=self.object_list[optimal_answer[1]]
                if self.pom2<=self.number_of_questions:
                   self.coordinatesObject=self.coordinatesObject+[self.make_yesNoCoordinates(
optimal_answer_info)]
                   self.namesObject=self.namesObject+[self.coordinatesObject[0]]
                   if self.pom2==self.number_of_questions:
                      self.log("IIIIIIIIIIIIIIIIII ---spousteni starting z detekce objektu---")
                      self.starting()
                   elif self.pom2<self.number_of_questions:
                      time.sleep(5)
                      while self.pomWait==1:
                         continue
                      self.eyeColor(self.yellow, "LeftFaceLeds")
                      time.sleep(5)
                      while self.pomWait==1:
                         continue
                      self.eyeColor(self.red, "LeftFaceLeds")
                      self.saySomething("Question  number  " + str(self.pom2+1))
                      self.object_label_list=[]
                      self.object_list=[]
                      self.pom1=0
                      self.pom2=self.pom2+1
```

```
    #--------------starting()--------------
    # tato funkce vytvari zaverecné vyhodnoceni
    # spusti se pouze pokud ma informace o tvari i o objektu(odpovedich), zaroven dochazi
    ke zmene barvy oci a robot to oznami slovne
    # vysledkem je: pro jednu osobu a jednu otazku: ['aja', 'ne'], pro vice osob seznam: [['aja', 'ne'],
    ['', 'ano'], ...]
    def starting(self):
        #probehne pouze pokud probehla detekce obliceje a objektu/odpovedi
        if (len(self.coordinatesFace)>=1 and len(self.namesObject)==self.number_of_questions):
            self.pomStarting=1
            self.pomDelStarting=0
            self.saySomething("Questions for person number "+ str(len(
self.list_of_face_and_answers)+1) + " is done")
            self.yes()
            face_object=self.classify_faceObject()
            self.log("IIIIIIIIIIIIIIIIII detekovana tvar a odpoved(i): " + str(face_object))
            self.list_of_face_and_answers=self.list_of_face_and_answers+[face_object]
            self.log("IIIIIIIIIIIIIIIIII celkovy detekovany seznam: " + str(
self.list_of_face_and_answers))
            time.sleep(5)
            while self.pom1==1:
                continue
            ALMotion.changeAngles(["HeadYaw", "HeadPitch"], [float(self.head_yaw[0]),
float(self.head_pitch[0])], 0.05)
            self.eyeColor(self.yellow, "FaceLeds")
            time.sleep(15)
            if self.pomDelStarting==0:
                pom=0
                while self.pom1==1:
                    pom=1
                    continue
                if pom==1:
                    time.sleep(20)
                self.pom1=1
                self.pom2=1
                self.coordinatesFace=[]
                self.coordinatesObject=[]
                self.namesObject=[]
                self.face_answer=[]
                self.object_list=[]
                self.object_label_list=[]
                self.face_list=[]
                self.face_label_list=[]
                self.eyeColor(self.red, "FaceLeds")
                self.saySomething("Person number " + str(len(self.list_of_face_and_answers)+1) +"
Question number " + str(self.pom2))
                self.pomStarting=0
                self.pom1=0
            self.pomDelStarting==0
```

```
#--------------oftenDetectedFace(facelist)--------------
#urci nejcasteji detekovanou tvar (pokud zna jmena)
#facelist=['', '', '', 'aja', ''] -> aja
#do funkce vstupuje seznam facelist s urcitym poctem detekovanych tvari, z nich vybere
nejcastejsi
#vraci: [prvni nejcastesi jmeno, index prvního nejcastejsího jmena]
def oftenDetectedFace(self, facelist):
    frequency=[]
    namelist=facelist
    x=0
    #vytvorim pole stejne velke jako facelist-tj. pocet detekovanych obliceju
    for j in range(0,len(facelist)):
        frequency=frequency+[0]
        j=j+1
    #prochazim facelist a následne vybiram nejcastejsi znamou tvar (pokud zadnou nezna,
    vratí '')
    for x in range(0,len(facelist)):
        pom=0
        if (str(facelist[x])==""):
            x=x+1
        elif (x==0):
            frequency[0]= frequency[0]+1
            x=x+1
        else:
            y=0
            for y in range(0,x):
                if facelist[x]==namelist[y]:
                    frequency[y]= frequency[y]+1
                    y=y+1
                    pom=1
            if (pom==1):
                namelist[x]=""
            if (pom==0):
                frequency[x]= frequency[x]+1
            x=x+1
    #answer=[prvni nejcastesi jmeno, index prvního nejcastejsího jmena]
    answer=[facelist[frequency.index(max(frequency))], frequency.index(max(frequency))]
    return answer
```

```
    #--------------oftenDetectedAnswer(objectlist)--------------
    #urci nejcasteji detekovanou odpoved
    #objectlist=['ne', 'ne', 'ano', 'ne', 'ne'] -> ne
    #do funkce vstupuje seznam objectlist s urcitym poctem detekovanych odpovedi, z nich
    vybere nejcastejsi
    #vraci: [ano/ne, index posledni z nejcastejsích odpovedi]
    def oftenDetectedAnswer(self, objectlist):
        #answer=[odpoved, index posledni z nejcastejsích odpovedi]
        answerY=["ano",""]
        answerN=["ne",""]
        frequencyY=0
        frequencyN=0
        x=0
        for x in range(0,self.number_of_object_detection):
            if (str(objectlist[x])=="ano"):
                frequencyY=frequencyY+1
                answerY[1]=x
            if (str(objectlist[x])=="ne"):
                frequencyN=frequencyN+1
                answerN[1]=x
            x=x+1
        if frequencyY>frequencyN:
            return answerY
        else:
            return answerN



    #--------------make_nameCoordinates(p)--------------
    #z detekovaneho obliceje vybere pozadovane informace
    #vraci: [jmenoTvare, souradniceOblicejX, souradniceOblicejY, souradniceLeveOkoX,
    souradnicePraveOkoX]
    def make_nameCoordinates(self, p):
        #nameCoordinates=[jmenoTvare, souradniceOblicejX, souradniceOblicejY,
        souradniceLeveOkoX, souradnicePraveOkoX]
        #Prave,leve oko podle obrazku kamery, x souradnice
        nameCoordinates=[p[1][0][1][2], p[1][0][0][1], p[1][0][0][2], p[1][0][1][3][0],
p[1][0][1][4][0]]
        return nameCoordinates
```

```
    #--------------make_yesNoCoordinates(p)--------------
    #vytvori pole detekovanych objektu/ odpovedi a jejich "stredovych" souradnic
    #vraci: [odpoved_ano/ne, xS, yS]
    def make_yesNoCoordinates(self, p):
        x=0
        sumaX=0
        sumaY=0
        averageX=0
        averageY=0
        #p[1][0][3][[x0, y0],[x1, y0], ...] ... boundary points [x, y]-> hranicni body (ohranicujici
        detekovany objekt)
        #len(p[1][0][3] ... pocet hranicnich bodu
        #z hranicnich bodu vytvorim pouze jeden "stredovy" [xS, yS], ktery bude dan aritmetickym
        prumerem vsech techto bodu
        for x in range(0,len(p[1][0][3])):
            sumaX=sumaX+p[1][0][3][x][0]
            sumaY=sumaY+p[1][0][3][x][1]
            x=x+1
        averageX=sumaX/(len(p[1][0][3]))
        averageY=sumaY/(len(p[1][0][3]))
        #do pole yesNoSouradnice si ukladam [odpoved_ano/ne, xS, yS]
        yesNoCoordinates=[p[1][0][0][0], averageX, averageY]
        #vraci pozadovane souradnice a nazev detekovane odpovedi  ... yesNoSouradnice =
        [odpoved_ano/ne, xS, yS]
        return yesNoCoordinates



    #--------------classify_distances()--------------
    #pocita rozmezi souradnic, kde se bude nachazet odpoved ANO-NE pod oblicejem
    #LEye_CFace_distance, REye_CFace_distance ... vzdalenost "x" oci od stredu tvare
    #distances_forObject_LR ...vzdalenosti "x" od stredu obliceje ve kterych uvazujeme odpoved
    (3x vzdalenost oci-stred_tvare od stredu_obliceje)
    #distances_forObject_LR=[L_souradnice, R_souradnice]
    #---nevyuzito---
    def classify_distances(self):
        LEye_CFace_distance=abs(self.coordinatesFace[1] - self.coordinatesFace[3])
        REye_CFace_distance=abs(self.coordinatesFace[1] - self.coordinatesFace[4])
        distances_forObject_LR=[self.coordinatesFace[1]+(3*LEye_CFace_distance),
self.coordinatesFace[1]-(3*REye_CFace_distance)]
        return distances_forObject_LR
```

```
#--------------classify_faceObject()--------------
#zajistuje spojeni vysledne detekovane tvare a vyslednych detekovanych odpovedi.
#vraci nazev tvare a detekovanych odpovedi
#self.face_answer=['jmeno', 'ano/ne', ...]
def classify_faceObject(self):
    self.face_answer.append(self.coordinatesFace[0])
    for x in range(0,self.number_of_questions):
        self.face_answer.append(self.coordinatesObject[x][0])
        x=x+1
    return(self.face_answer)



#--------------eyeColor(p, sGroup)--------------
#tato funkce meni barvu oci robota
def eyeColor(self, p, sGroup):
    #sGroup = "LeftFaceLeds"
    #sGroup = "RightFaceLeds"
    #sGroup = "FaceLeds"
    id = ALLeds.post.fadeRGB(sGroup, 256*256*p[0] + 256*p[1] + p[2], 1)
    self.ids.append(id)
    ALLeds.wait(id, 0)
    self.ids.remove(id)
    pass



#--------------saySomething(self, s)--------------
#robot promluvi retezec "s"
def saySomething(self, s):
    sentence = "\RSPD="+ str(100)+ "\ "
    sentence += "\VCT="+ str(100) + "\ "
    sentence += str(s)
    sentence +=  "\RST\ "
    id = self.tts.post.say(sentence)
    self.ids.append(id)
    self.tts.wait(id, 0)
    self.ids.remove(id)
```

```
    #--------------yes()--------------
    #funkce vygenerovana Choreographem pro pohyb - kyvani hlavou robota
    def yes(self):
        # Choregraphe bezier export in Python.
        from naoqi import ALProxy
        names = list()
        times = list()
        keys = list()

        names.append("HeadYaw")
        times.append([ 1.30000, 5.00000, 9.20000])
        keys.append([ [ 0.00000, [ 3, -0.43333, 0.00000], [ 3, 1.23333, 0.00000]], [ 0.00000, [ 3, -1.23333, 0.00000], [ 3, 1.40000, 0.00000]], [ 0.00000, [ 3, -1.40000, 0.00000], [ 3, 0.00000, 0.00000]]])

        names.append("HeadPitch")
        times.append([ 1.30000, 2.40000, 4.30000, 5.00000, 9.20000])
        keys.append([ [ 0.00000, [ 3, -0.43333, 0.00000], [ 3, 0.36667, 0.00000]], [ -0.40143, [ 3, -0.36667, 0.00000], [ 3, 0.63333, 0.00000]], [ 0.27925, [ 3, -0.63333, 0.00000], [ 3, 0.23333, 0.00000]], [ -0.13614, [ 3, -0.23333, 0.00000], [ 3, 1.40000, 0.00000]], [ 0.00000, [ 3, -1.40000, 0.00000], [ 3, 0.00000, 0.00000]]])

        try:
          # uncomment the following line and modify the IP if you use this script outside
          Choregraphe.
          # motion = ALProxy("ALMotion", IP, 9559)
            motion = ALProxy("ALMotion")
            motion.angleInterpolationBezier(names, times, keys);
        except BaseException, err:
            print err



    def onInput_onStop(self):
        pass
```

Příloha B - Korelace

Příkazy využité v programu MATLAB pro získání korelací jednotlivých otázek

Pozn.: V programu MATLAB se komentáře značí symbolem %. Uvedené příkazy jsem pouze zadávala jako příkaz.

```
%VYUZITE PRIKAZY:
%ano 1, ne 0
%vysledky ankety:
A=    [ 0 0 0 1 1;
        1 0 1 1 0;
        1 1 0 1 0;
        0 1 0 1 0;
        1 1 0 1 1;
        1 0 0 1 1;
        1 0 0 1 1;
        1 0 0 1 1;
        1 0 1 1 1;
        1 0 1 1 1;
        1 0 0 1 0;
        1 0 0 1 0;
        0 1 0 1 1;
        0 0 1 1 1;
        1 1 0 1 1;
        1 1 0 1 1;
        0 1 0 1 0;
        1 0 0 1 1;
        0 1 0 1 1;
        1 1 0 0 0;
        0 1 0 0 0;
        1 1 1 0 1;
        1 0 1 1 0;
        1 0 1 1 1;
        0 0 0 1 1;
        0 0 0 1 0;
        1 1 1 1 1;
        1 1 0 1 0];
%vykresleni korelace:
imagesc(corrcoef([A(:,1)';A(:,2)';A(:,3)';A(:,4)';A(:,5)'])),title('Korelace otazek dotazniku')
%hodnoty korelace:
corrcoef([A(:,1)';A(:,2)';A(:,3)';A(:,4)';A(:,5)'])
```

Fungovat, či nefungovat

Robot Asterix,

Robot NAO, ČVUT FEL