TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Humanoidní robot Nao v úloze avatara hostitele

Bakalářská práce

Studijní program:	B2646 Informační technologie
Studijní obor:	1802T007/90 – Informační technologie

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

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Zadání bakalářské práce

Humanoidní robot Nao v úloze avatara hostitele

Jméno a příjmení: **Pavel Vican**
Osobní číslo: M18000110
Studijní program: B2646 Informační technologie
Studijní obor: Informační technologie
Zadávající katedra: Ústav informačních technologií a elektroniky
Akademický rok: **2021/2022**

Zásady pro vypracování:

1. Seznamte se s humanoidním robotem NAO na pracovišti školitele.
2. Proveďte rešerši stávajícího stavu dostupných úloh pro roboty NAO, ve kterých vystupují jako avataři.
3. Navrhněte software, pomocí kterého bude možné provozovat robota NAO v roli avatara. Vzdálený oprerátor bude mít k dispozici audio a video z robota a jeho stavové informace, vše v reálném čase. Dále bude možné pronášet části dialogu a vykonávat předpřipravené pohyby. Při návrhu zohledněte možnost snadné modifikovatelnosti ukázkové úlohy robota jako hostitele.
4. Navržené programové vybavení realizujte a ověřte jeho funkcionalitu.
5. V závěru diskutujte výhody a nevýhody realizovaného návrhu včetně možných vylepšení.

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

Seznam odborné literatury:

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

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

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

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

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

L.S

prof. Ing. Zdeněk Plíva, Ph.D.
děkan

prof. Ing. Ondřej Novák, CSc
vedoucí ústavu

V Liberci dne 19. října 2021

Prohlášení

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

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

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

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

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

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

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

15. května 2022 Pavel Vican

Abstrakt

Cílem bakalářské práce je průzkum dostupných úloh, kde robot Nao vystupuje jako avatar. Následně navrhnout software pro daného robota, pomocí kterého bude možno ho v takovém režimu ovládat. Rolí avatara se v této práci rozumí dálkové ovládání na živo vzdáleným operátorem pomocí programu. Operátor má možnost s robotem chodit v prostoru, vykonávat dialog pomocí text-to-speech a využít předpřipravené příkazy. K ovládání robota v tomto režimu bude třeba získávat živě jeho stavové informace. K těmto informacím patří stav baterie, přenos videa a zvuk z jeho mikrofonů. Aplikace byla vytvořena pomocí programovacího jazyka Python 2.7 s ovládací knihovnou NAOqi od tvůrců robota. Další významné podpůrné knihovny pro zpracování aplikace jsou VLC, paramiko a Tkinter. Aplikace je ovládána zadáním textu na proslov či odpovídajícímu předpřipraveným příkazům. Zbytek ovládání je proveden přes tlačítka. Pro připojení k robotovi je třeba být ve stejné síti či obecně mít přístup k jeho IP adrese a portu.

Klíčová slova

Nao; Python 2.7; humanoidní robot; vzdálené ovládání; dynamická robotika

Abstract

Bachelor’s thesis goal is to research the available tasks, in which robot Nao acts as an avatar. Subsequently, design a software for the robot, with which it will be possible to control it in such a way. The role of the avatar in this thesis means live control by a remote operator using the software. The operator has the opportunity to walk with the robot in space, engage in dialogue using text-to-speech and execute pre-prepared commands. To control the robot in this mode, it will be necessary to obtain its status information live. This information includes battery status, live video feed and audio from its microphones. This application was created using Python 2.7 programming language with special control library NAOqi. Other important supporting libraries for application processing are VLC, paramiko and Tkinter. The application is used by entering text into particular text fields to either speak or execute pre-programmed commands. Other controls are performed via buttons. To connect to a robot, one must be on the same network or generally have access to its IP address and port.

Keywords

Nao; python 2.7; Python; humanoid robot; remote control; dynamic robotics

Obsah

SEZNAM OBRÁZKŮ

POUŽITÉ ZKRATKY, ZNAČKY A SYMBOLY

FPS – snímky za sekundu

TTS – text-to-speech – text na řeč

GUI – Graphical User Interface – grafické uživatelské rozhraní

API – Application Programming Interface – Aplikační programovací rozhraní

1 Úvod

Cílem bakalářské práce bylo provést rešerši stávajícího stavu dostupných úloh, ve kterých vystupuje robot Nao jako avatar a prošetřit jejich provedení. Úlohy je třeba rozeznávat dle podobnosti ke konkrétnímu zadání, jelikož obsah jiných prací nemusí být v jejich částech či konečnému cíli porovnatelný.

Dalším cílem bylo navrhnout software, kterým bude lze provozovat robota Nao dálkově v roli avatara. Součástí programového vybavení bude tedy ovládání pohybů do stran a vpřed (dopředu, vlevo, vpravo), možnost dialogu pomocí text-to-speech (také jako TTS) a vykonávat předpřipravené příkazy. Rozsah předpřipravených příkazů se může znatelně lišit jak obsahem, tak i rozsahem, například pozdrav zamáváním, potřesení rukou dotykem senzoru po vyvolání příkazu či sekvence obdoby tance. K ovládání robota na dálku bude operátor potřebovat jeho stavové informace (jako je stav baterie), živý záznam z videokamer a zvuk pořízený na robotovi. Operátor má také mít k dispozici jiné nastavení a polohy nezbytné pro plynulý provoz a samostatnost robota. To zahrnuje změnu polohy robota (mezi odpočinkovou a aktivní polohou), nastavování hlasitostí aplikace či robota samotného. Nakonec možnost přepínání mezi kamerami robota pro lepší orientaci v prostoru, pokud bude třeba.

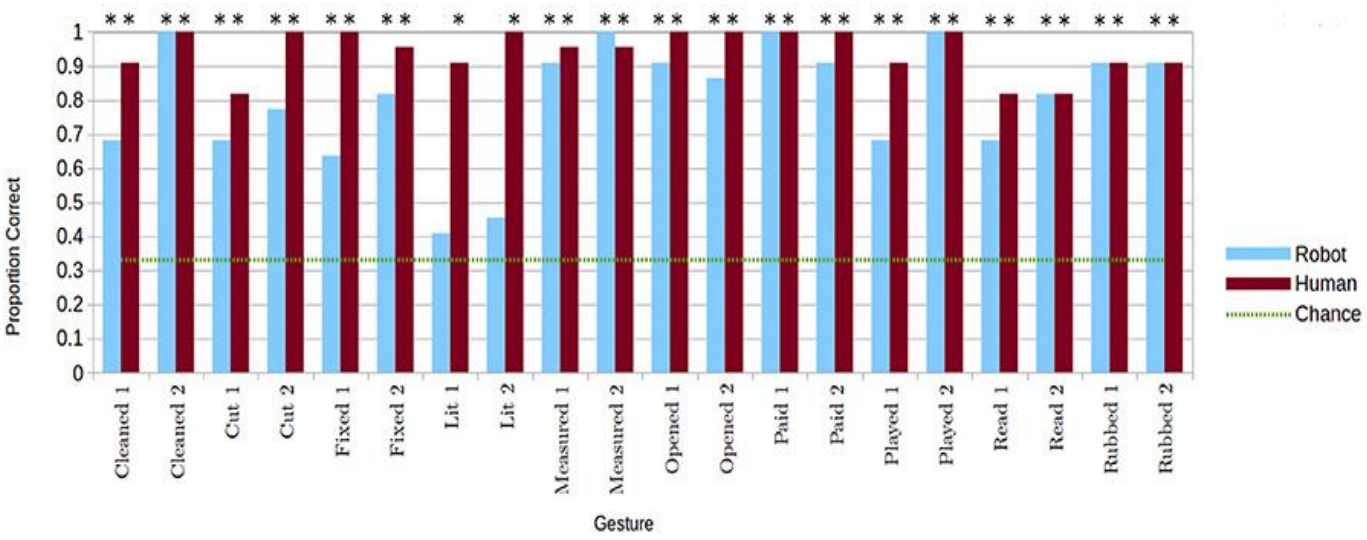
Mezi volby programovacího jazyka patří C++ a Python 2.7. Jako programovací jazyk byl zvolen Python 2.7 použit v prostředí JetBrains Pycharm 2019. Podstatné jsou využité knihovny VLC pro zvuk z robota, paramiko pro SSH připojení k robotovi a Tkinter pro zpracování grafického rozhraní programu operátora. Jako nejdůležitější knihovna potřebná pro uskutečnění bakalářské práce je NAOqi, jenž patří k nezbytným částem robota Nao od Softbank Robotics, pomocí které se dá robot dálkově ovládat a inicializovat bez zanechání dat na robotovi samotném.

Důvodem zpracování tohoto typu práce byl pocit nedostatku obdobných dostupnějších řešení nejen na univerzitě, ale i na západní části světa. Roboti Nao a Pepper od společnosti Softbank Robotics jsou zastoupeny ve světě mimo Asii (především Japonsko) velice zřídka a nemusí tomu tak být. Dalším důvodem je možnost použití této práce jako platformu pro další rozšíření více řešiteli, pro to je třeba klást důraz na snadnější čitelnost a upravitelnost programu.

2 ÚLOHY, KDE NAO VYSTUPUJE JAKO AVATAR

Existuje mnoho podobných aplikací, žádná není ale principem zpracování stejná. Způsoby těchto prací už přirozeně budou psány v Python 2.7 či C++ jazycích kvůli knihovně Naoqi [1], kterou je robot Nao ovládán. Další možností zpracování jsou nepřímo programové pomocí programu Choreographe, který patří k robotovi Nao přímo od tvůrčí společnosti SoftBank Robotics. Tyto neprogramové řešení ale budou limitované svou schopností kvůli předem daným ovládacím prvkům s horší upravitelností v porovnání s programovacími jazyky Python a C++.

Prvně vybraný vědecký článek [2] se zaobírá využitím gest a mluveného slova robota Nao a člověka pro porovnání rozpoznávání určitých aktivit. Experimentální studie byla provedena na „22 účastnících (10 žen, 12 mužů) s věkem mezi 18-55 let“ [2]. Nao robot hraje v této studii roli avatara, kde převod akcí člověka, co bude robot co nejpřesněji provádět bylo třeba nějak zaznamenat. Pro to byl vybudován systém pomocí silné ROS [3] knihovny. Která bude mít za úkol převádět informace z pohybu člověka přes Kinect (snímač pohybu účastníka) a speciálních pohybových rukavic pomocí niž robot čerpá informace pro vykonání stejných akcí, jenž pomocí nich zaznamenal. Výhoda výběru ROS knihovny je její modulární struktura, kde v případě selhání určitých modulů systém poběží dál mezitím co se modul, u nějž došlo k selhání restartuje. Tento systém může být naprogramován jak jazykem C++, tak Python či kombinací obou těchto jazyků. Byl vytvořen seznam příkazů vybraných z často citovaného průzkumu o integraci gest a slov jenž probíhal u mladších i starších dospělých [4]. Provedený experiment poté zkoumal úspěšnost rozpoznávání provedených gest (prováděných robotem či člověkem) artikulovaných, pouze pohybových nebo kombinace obojí u účastníků. Účastníci měli po zakončení ukázky (jak na robotovi či člověku) správně vybrat z šesti jim prezentovaným obrázkům dva, u kterých se domnívají, že je ukázka znázornila.



Obrázek 1 Graf ukazující správnost ukázaných gest zaznamenaných účastníky. [2]

Výše uvedený graf popisuje rozdíly ve vnímání určitých gest účastníky experimentu. Z dat je zřejmé, že rozdíl vnímání gest od robota oproti člověku je u účastníků jiný. Odlišnost není však tak velká kromě u pár určitých gest. Studie tedy využívá robota Nao pro testování ukazování stejných gest od člověka v porovnání s robotem a měří, zda účastníci správně rozeznají prezentovaná gesta. Pro porovnání naměřených hodnot studie provádí různé statistické testy (jako například McNemar test, ANOVA a další) pro porovnávání správnosti a hlubší rozbor do možných důvodů rozdílu dat u robotického avatara a člověka. V závěru jako jeden z poznatků studie shrnula, že rozpoznávání gest konaných robotem má sice horší výsledky, nejsou ale na tolik odlišné, aby se robot nemohl vůbec používat pro dané aktivity. V případě zvukového doprovodu prováděných akcí robot kompenzoval naměřené rozdíly v datech oproti případů, kde se pracovalo pouze s konáním s neverbálními gesty. Užitím robota v dané aplikaci také ukázalo, že je robot více přesvědčivý (dle účastníků), když používá neverbální gesta. Účastníci také kladně hodnotili výstup a dobrý vzhled robota se záměrem další budoucí spolupráce v podobných studiích. Studie tedy ukázala kladný způsob využití robotů (v tomto případě robot Nao) v úložce avatara, kdy robot dělá určité pohyby a může tím přesvědčovat své okolí potencionálně lépe, než by mohl lidský jedine.

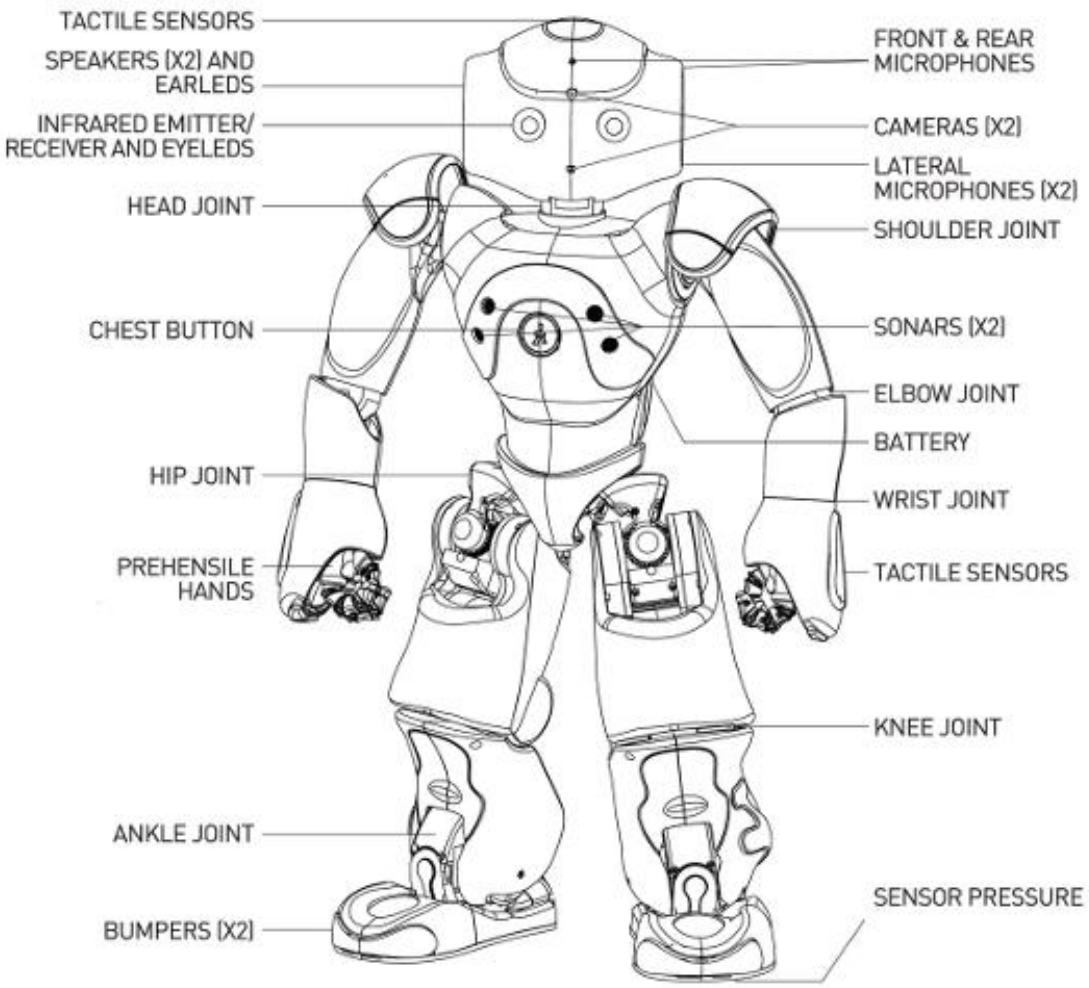
Nejbližší tématu podobná a více pokročilá práce je zachycena pouze v podobě videa [4]. Ukázka robota se koná dle mluveného jazyka v Itálii, která je nahrána na Italsky mluvící YouTube kanál. Z videa můžeme sledovat robota Nao vzdáleně ovládaného pomocí notebooku. Video představuje jakousi aplikaci, jenž má široké spektrum tlačítek a modulů pro ovlivnění činnosti robota. Rozsah této aplikace je impozantní (alespoň dle vzhledu grafického prostředí a množství tlačítek, tedy možných funkcí) a ukazuje všechny možnosti různých předpřipravených příkazů,

jako je tanec a podobně. Robot je také prezentován s možností rozeznat různorodé obrázky a patřičně na ně reagovat. Jelikož robot komunikuje pouze v italštině tak není přesně rozumět, co říká.

Další práce je přímo pojmenovaná Nao Robot Avatar [5]. Práce je publikována Tureckou univerzitou v Istanbulu Boğaziçi. Zadání je zpracování a ovládání Nao robota pomocí brýlí na virtuální realitu, které si operátor nasadí. Pomocí gest a pohybů lze robota Nao ovládat. Tato práce výrazně odlišné zpracování, jelikož je zde k dosažení daného cíle využita právě virtuální realita. Cílem práce je zvládnout dojít s robotem k vlajce a následně se jí dotknout. K ovládání na dálku je třeba robotovi pro vyhnutí se nevolnosti, ke které by mohlo dojít z používání virtuální reality nasadit speciální brýle (konkrétně dvě webkamery), které svůj obraz poskytnou operátorovi pro odlišný obraz (nebo alespoň jeho iluzi) pro každé oko. Tato práce je naučná pro využití rozpoznávacích algoritmů, které jsou potřeba pro rozeznání gest a pohybů operátora. Tyto algoritmy dále vyhodnotí z daných gest požadovaný příkaz pro robota, aby vykonal. Z poskytnutého videa je vidět na operátorovi, že zvedá nohy v rytmu chůze, aby algoritmus vyhodnotil příkaz pro robota, aby šel kupředu. Dalším zajímavým poznatkem je možnost volného pohybu robota hlavy robota (ačkoliv se znatelnou odezvou) pomocí hýbání hlavou operátora s brýlemi. Tomuto projektu chybí na obsahu k dalším akcím, je tedy poměrně krátce rozepsaný rozsah.

Ostatní zdroje informací o těchto typech prací jsou těžko dostupné pro anglicky a česky psanou část internetu. Hlavním důvodem je velice malé zastoupení robotů od společnosti Softbank (tvůrců Nao a Pepper robotů) v západní části světa. Největší zastoupení těchto robotů je na asijském kontinentu, obzvláště pak v Japonsku, jelikož odtud společnost pochází a více zde podporuje lokální rozvoj. Důvodem tohoto úsudku bylo při vyhledávání podobných úloh mnohokrát větší četnost zastoupení videí a jiných zdrojů popsanými asijskými jazyky. Tyto zdroje by bylo obtížné popisovat a byli pro to vynechány.

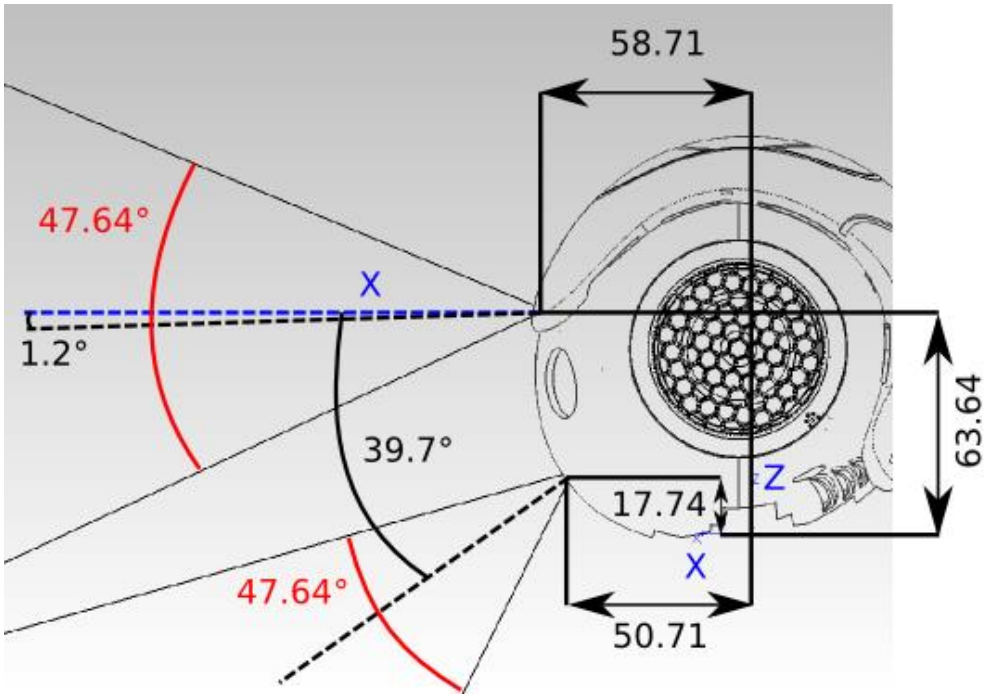
3 VYBAVENÍ ROBOTA NAO



Obrázek 2 Diagram robota Nao popisující lokace všech použitelných periferií.

Robot Nao disponuje mnoha různými periferiemi [6]. Obsahuje 2 kamery, 2 mikrofony (přední a zadní), senzory na končetinách, dotykové senzory na hlavě a mnoho dalšího potřebného pro ovládání humanoidního robota viz. Obrázek 2. Z důvodu rozsahu práce a nevyužití všech použitelných součástí nebudou popsány do všechny dostupné součásti. Například sonarové (ultrazvukové) senzory či tlakové senzory na špičkách nohou. Robot disponuje krajně nedostatečným procesorem Intel ATOM Z530 1,6 GHz [7], jenž dělá problémy hlavně u zachytávání obrazu, audia a obecně vnitřních výpočtů. Z tohoto důvodu je třeba omezit vnitřní náročnější operace, například konverzi barev z kamery je lepší poslat na externí zařízení (počítač) a tam převést YUV -> RGB. Robot také váží více než 5 kilogramů, je tedy poměrně těžký vůči jeho malé výšce.

3.1 Kamery



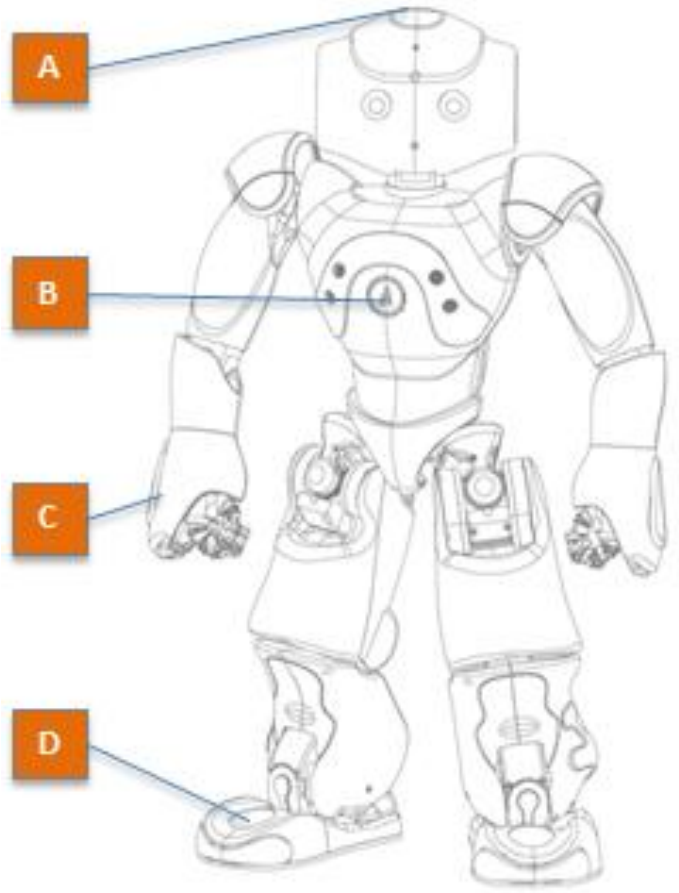
Obrázek 3 Lokace a úhel pohledu kamer robota Nao

Robot disponuje dvěma kamerami [8]. První kamera je ve předu mezi očima mírně nahoře pro vidění dopředu, před robota.

Druhá kamera se nachází u „úst“ robota (lokace je schválně vybrána, aby připomínala lidská ústa), kde je pouze kamera samotná na daném místě. Tato kamera má za úkol mířit na spodní část robota pro snazší vidění níže – až po nohy či samotné tělo, pokud se hlava dostatečně nakloní.

Obě kamery disponují kapacitou snímat obraz maximálně na HD+ kvalitě – 1280x960 pixelů. Maximální snímací frekvence je 30 obrazů za sekundu (také jako FPS). Nativní kamerou snímaný formát barev je YUV422, to je limitující kvůli omezené výpočetní kapacitě robota. Tato výpočetní kapacita je silně limitována typem snímacího režimu, tedy pokud chceme vysílat na naše zařízení živý přenos obrazu, tak nám robot není schopen zasílat obraz o frekvenci 30 FPS kvůli obtížnosti vypočítávat převod barev dostatečně rychle.

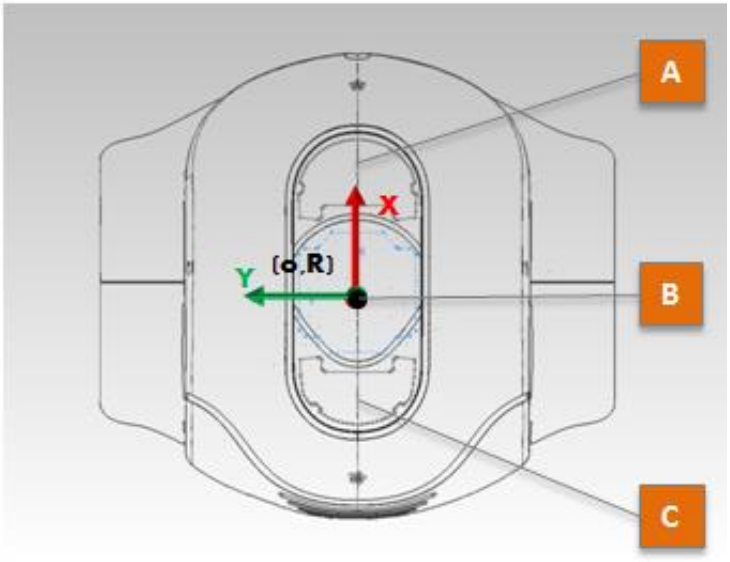
3.2 Dotyková čidla



Obrázek 4 Lokace dotykových čidel na robotovi Nao

Robot disponuje několika dotykovými čidly [9]. Tlačítka na hlavě a rukou robota (A, C) jsou aktivovaná lehkým dotykem bez potřeby na ně tlačit či je „mačkat“, dávají pocit toho, že jsou kapacitní. Tlačítkem B se robot primárně zapíná či zjišťuje internetová adresa pro připojení. Tlačítko je jako jediné třeba pro spínání znatelně stisknout. Tlačítka na nohou (D na obrázku) jsou „nárazová“ aktivovaná primárně když robot narazí na překážku na úrovni nohou. Příkladem využití může být schod, práh či zeď před robotem kde je možnost na tuto událost programově reagovat.

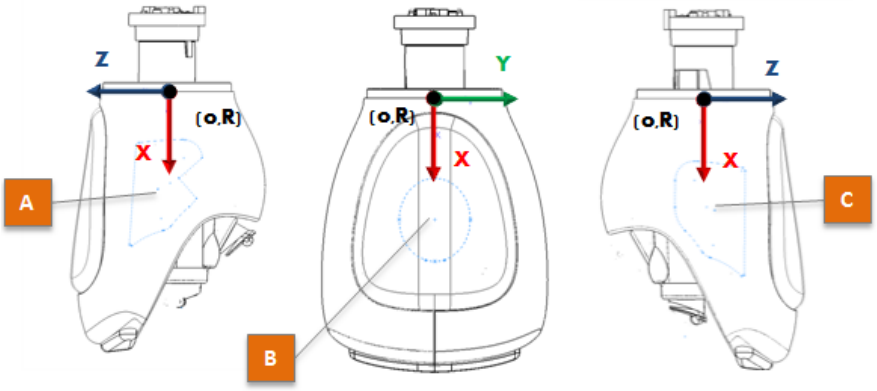
3.2.1 Čidla na hlavě



Obrázek 5 Hlava s dotykovnými tlačítky robota Nao

Na hlavě se nachází 3 dotyková čidla, všechny jsou plně programovatelná v tom, jakou funkci by mohli plnit. V rozsahu práce je přiřazena tlačítku C funkce mluvení ze seznamu faktů, které může robot povědět. Tlačítka stačí lehce podržet či stisknout pro zaznamenání události stisku.

3.2.2 Čidla na rukou



Obrázek 6 Čidla na rukou robota Nao

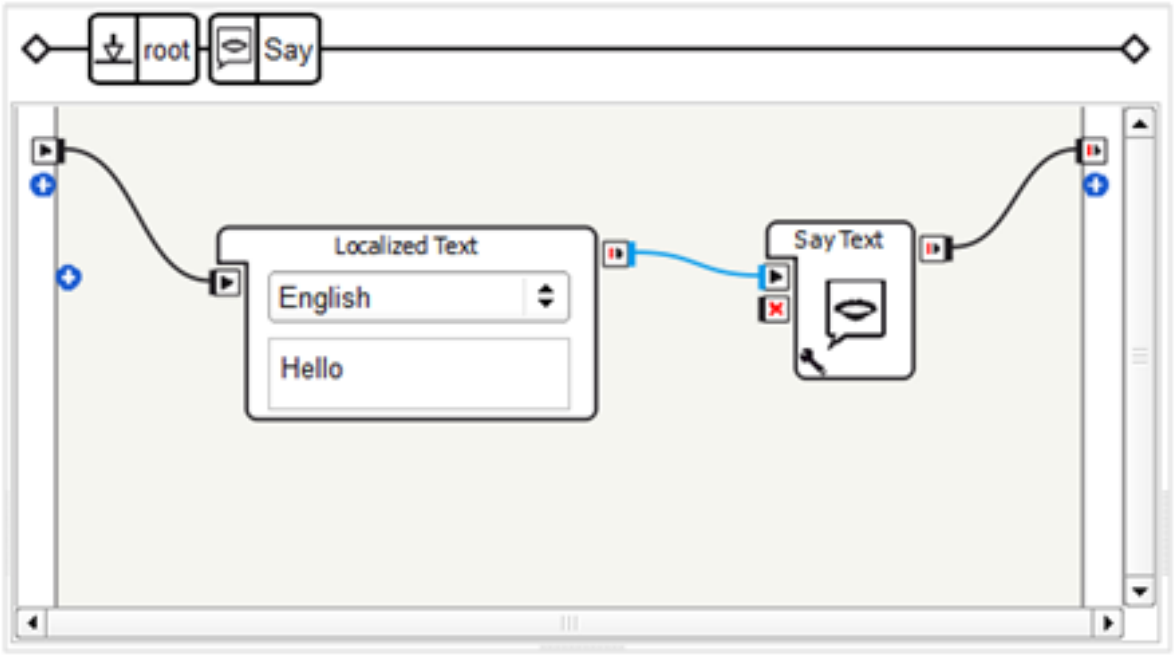
Čidla na rukou jsou obdobně stejně programovatelná. Dají se využít kdykoliv pro dodatečné funkce. Například při požadavku od programu ruku zvednout a čekat určitý čas,

při dotyku čidla na ruce se bude simulovat předpřipravený příkaz potřesení rukou. Robot také disponuje možností pro měření síly stisku (ačkoliv s malým výkonem), díky tomu je možno chytit dostatečně lehký objekt (například plyšový míč) a manipulovat s ním v prostoru. Tato možnost je poměrně omezená právě kvůli omezené síle stisku robota a malou velikostí ruky, pravděpodobně z bezpečnostních důvodů.

3.3 Programové vybavení robota

Systém, ve kterém se Nao spouští a ovládá vnitřně či se nastavuje je nazýván OpenNAO [10], jenž je distribuce GNU/Linux [11] upravená jeho specifickým hardwarovým potřebám. V systému robota je možnost nainstalovat dodatečné balíčky a úpravy dle potřeby uživatele. Tvůrčí společnost Softbank má také možnost zakoupení placených balíčků v jejich obchodě online určené přímo pro robota Nao. Příkladem takového balíčku je personalizovaný pozdrav robota či zamávání dle požadavku kupujícího.

Pro různé typy ovládání robota Nao patří programování přímo pomocí jazyka Python 2.7 či C++ s využitím NAOqi knihovny. Dalšími možnostmi je také program Choreographe [13], jenž je graficky zastoupený s možností skládání funkčních bloků jako je například promluv „Ahoj“, „Zamávej rukou“ a podobně. Díky různorodosti možností ovládání je robot vhodný i pro mladší ročníky na základních či středních školách až po výzkumné instituce dle požadovaného záměru.

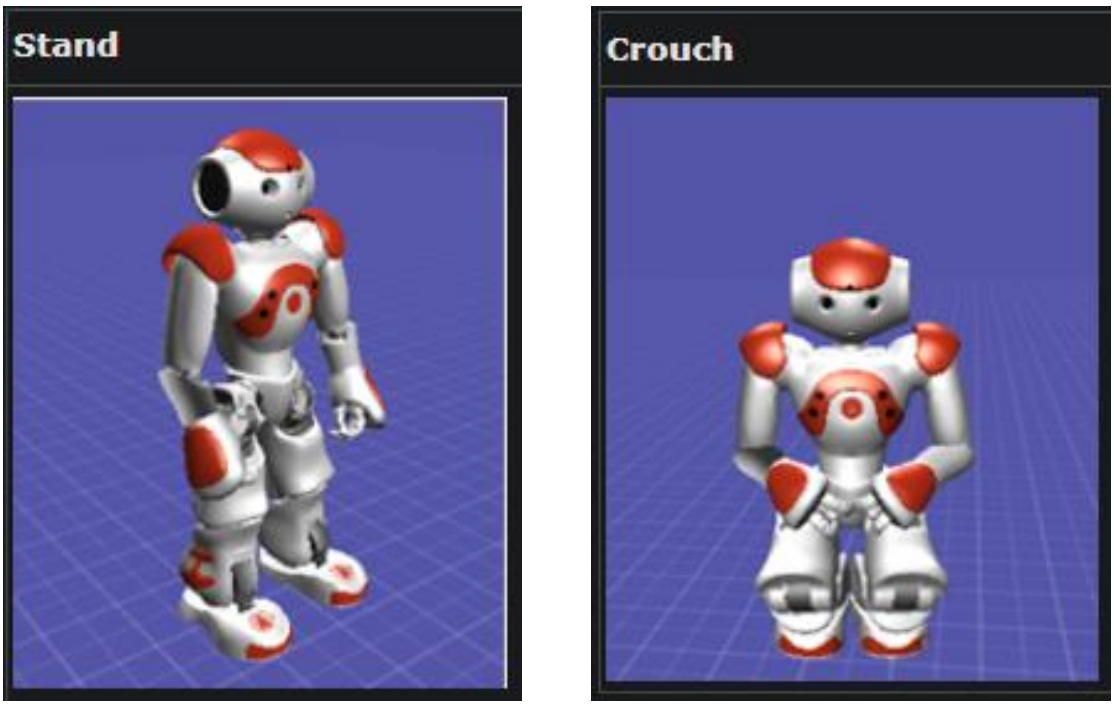


Obrázek 7 Ukázka příkazu v programu Choreographe [13]

Robot disponuje celou řadou předem nakonfigurovaných funkcí a limitů, jenž používají stávající senzory. Příkladem jsou senzory na nohou, jenž kontrolují, zda robot má pevné uzemnění a nehrozí mu riziko přepadnutí přes hranu, kdy se případně zastaví a dál nepůjde. Další bezpečnostní řízení (či reakce) je při zjištění pádu, kdy se robot pokusí před sebe nebo za sebe dát rychle ruce, ignorující nastavené limity pro uživatele, aby zabránil úderu do zbytku těla, především do hlavy, kde sídlí výpočetní jednotka.

Důležitou součástí programového vybavení je konvenční provedení určitých příkazů využitých v bakalářské práci. Například chození robota je provedeno jediným příkazem s upravením parametrů jako je rychlost a směr (při potřebě nejít pouze rovně). Dalším často použitým příkazem je text na řeč, kde je možno pouze zadat text a robot jej vysloví v jeho nastaveném jazyce.

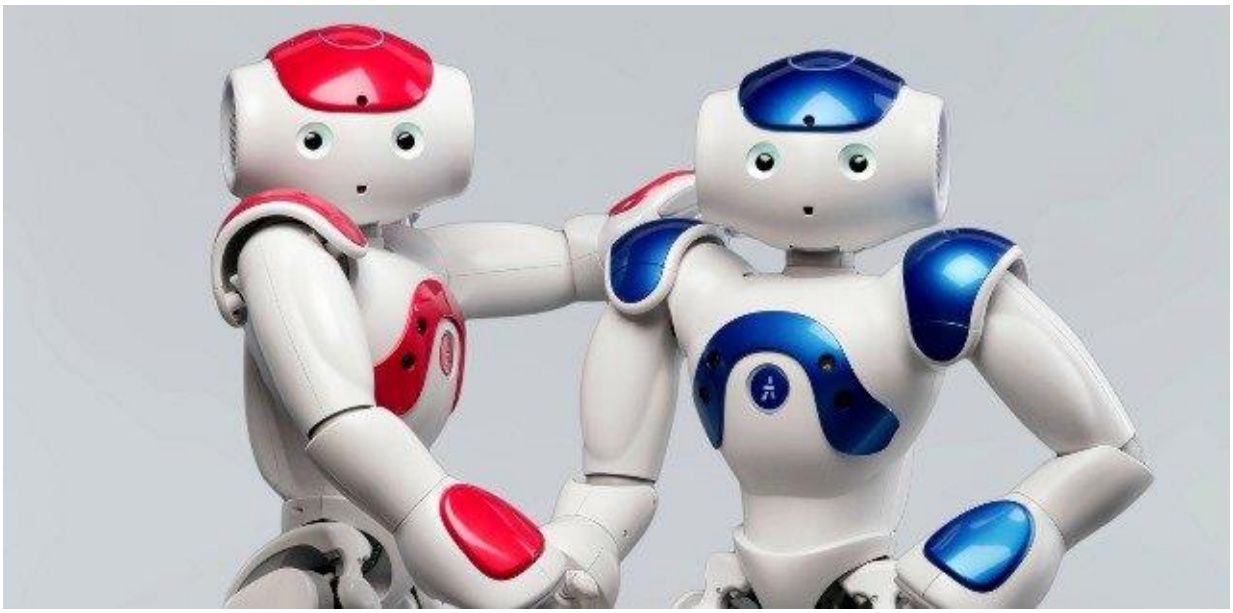
Není výjimkou plná upravitelnost jakékoliv programové části robota od naprogramování vlastního způsobu pohybu až po různá kritéria detekce obličejů, které má robot také zabudován. Robot také disponuje množstvím předdefinovaných poloh [12], které mohou být využity pro snadnější úpravu a nastavení jiných částí robota. Tyto polohy jsou nejlépe použity pro „akční“ polohu stání viz. Obrázek 8 jako „Stand“, kde je s robotem vhodné manipulovat (například chozením v prostoru) či ovládání rukou pro jiné úkony. Poloha přikrčení viz. Obrázek 8 jako „Crouch“ je nejlépe použita pro uvedení robota do stavu klidu, kde nevyužívá všechny servomotory a má výrazně menší spotřebu baterie



Obrázek 8 Předdefinované pozice "Stůj" a "Přikrčení"

3.3.1 Autonomní život

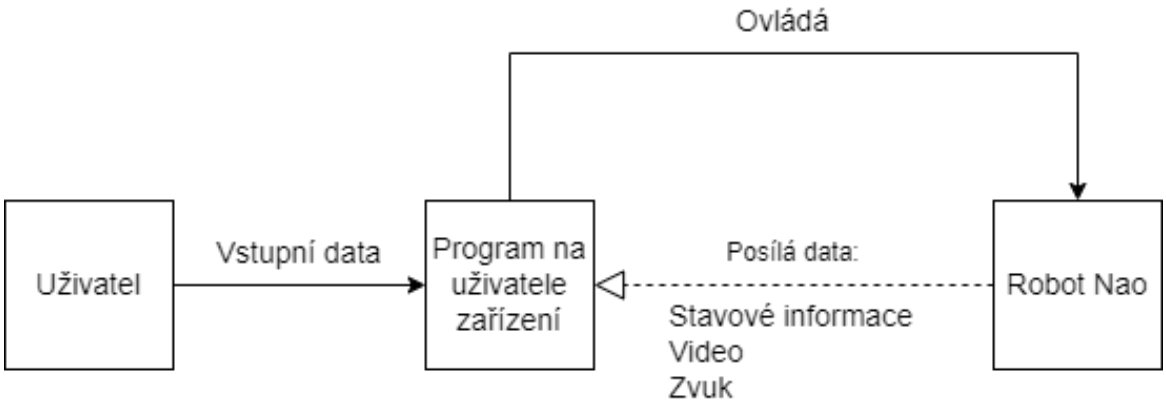
Unikátní vlastností robotů Nao je tzv. „Autonomní život“ [13]. Jedná se o originální provedení s automatizací zapínání aktivit a ostatních funkcí robota. Pokud jej uživatel neupraví, robot se bude automaticky spouštět se základním vnímáním svého okolí. V tomto režimu se bude robot při vzpřímené pozici klasicky stojící automaticky hýbat podobně člověku. To zahrnuje hýbání těla na místě a koukání se po okolí. Při spatření obličeje člověka se robot na tento obličej soustředí navazující „oční“ kontakt. Tyto aktivity se při pokročilejším nastavení a ovládnutí robota dají rozšířit či plně vypnout dle potřeby. Aktivity pro přidání jsou součástí obchodu Softbank kde jsou zpoplatněné pro nainstalování na robota. Příkladem takové aktivity může být vlastní upravený pozdrav pořízený z obchodu, který se automaticky provede při nasnímání obličeje dle nastavení robota.



Obrázek 9 Nao V5 evoluce [33]

4 NÁVRH PROGRAMU

Program psaný v jazyce Python 2.7 je třeba navrhnout pro základní ovládání robota dle požadavků. Program je třeba navrhnout s co největší přesností rozložení elementů a funkcionality, aby se předešlo časově náročnému zpětnému vracení. Z důvodu malého množství zkušeností s návrhy a rozložením grafického rozhraní se tomuto problému bude velice obtížné plně vyhnout. Pro směrodatnou myšlenku celkového návrhu byl nakreslen diagram pro orientaci směru dat a ovládání. Nechceme, aby v robotovi po manipulaci s programem zbyla nějaká data.



Obrázek 10 Diagram komunikace dat

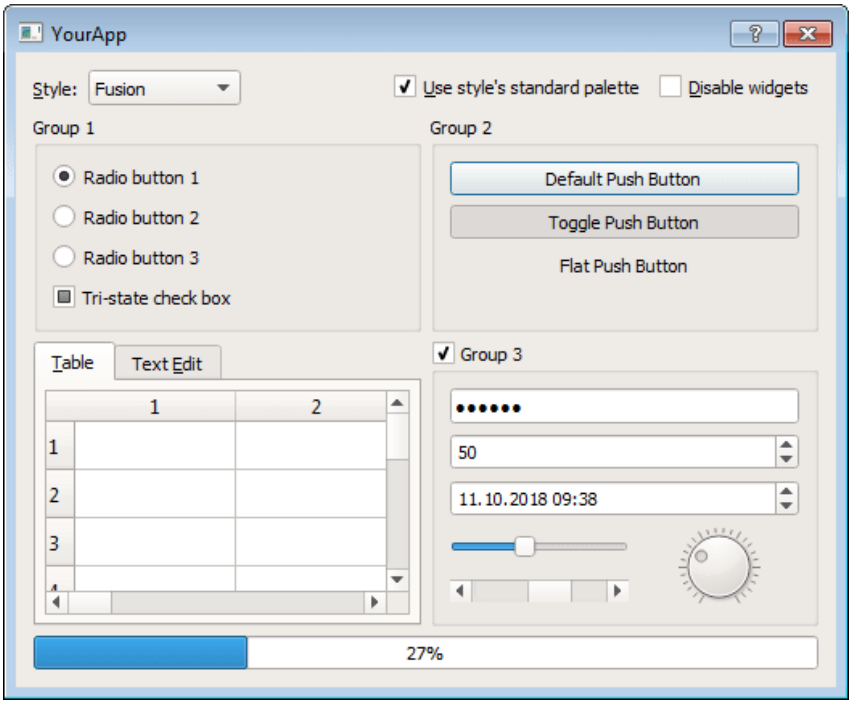
Grafický design je velice subjektivní věc, není tedy možno brát příliš velký ohled na vzhled, obzvláště u starší verze Python 2.7 kde dostupnost balíčků s „hezčím“ designem prvků je poměrně omezeně dostupná.

4.1 Návrh Grafického uživatelského rozhraní (GUI)

4.1.1 Výběr knihovny pro vývoj GUI

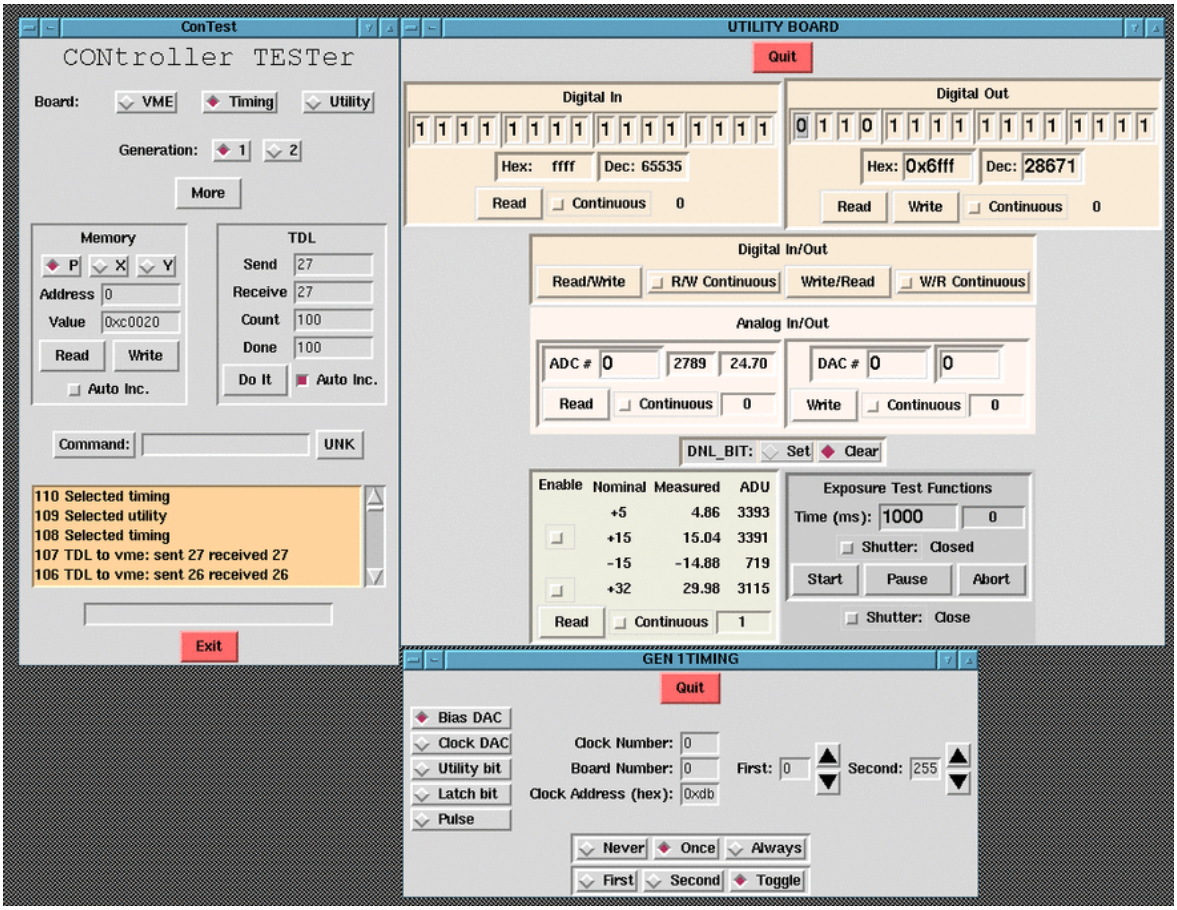
Velice důležitý aspekt vývoje jakéhokoliv rozhraní je výběr správných knihoven, pokud jsou dostupné. V případě tvorby GUI je dostupnost dostatečná (ačkoliv ne taková jako u novějších verzí Python). Pro výběr je třeba zohlednit faktory, jež jsou důležité pro poměrně jednodušší menší aplikaci jako je právě v tomto případě. Hlavními faktory k zohlednění jsou tedy: Jednoduchost, tvorba elementů pro GUI musí být jednoduchá a jasná. Rychlost naučení se práce i s pokročilejšími funkcionalitami knihovny. Dobře zpracovaná dokumentace, tím pádem i větší použitelnost mezi komunitou pro případné řešení problémů.

Do výběru tedy spadají dvě hlavní knihovny: PyQt a Tkinter. Pro porovnání mezi nimi bez přílišného se zabývání oficiální dokumentací, jenž by trvalo mnoho času, byl použit článek [14] od známé skupiny DEV community. Z článku vychází to, že PyQt je zpracováno okolo konceptu „signálů a slotů“ pro komunikaci mezi objekty, jenž má za cíl vetší flexibility s vypořádáním se s GUI událostmi a výsledky. Další „výhodou“ je dostupnosti širokého spektra API (předpřipravené funkce). Z těchto poznatků lze odsoudit, že knihovna obsahuje mnohem více funkcionalit, než jsme schopni využít. Zaměříme se tedy na Tkinter.



Obrázek 11 Vzhled grafických prvků knihovny PyQt [32]

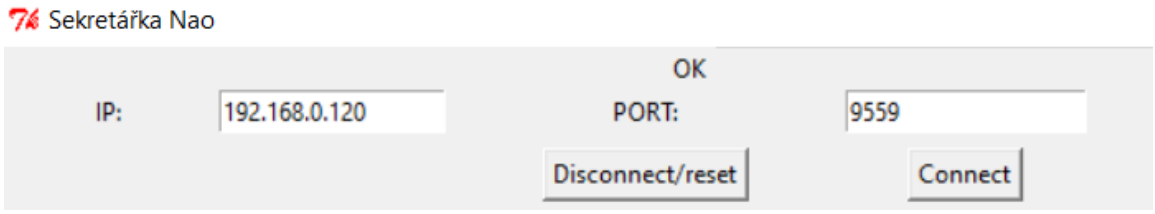
Tkinter Patří od základu do Python knihoven, tedy bez žádných dalších potřebných závislých knihoven. Tkinter je také jednoduchý na použití a porozumění. Z poznatků ze článku a menší pohled do dostupné dokumentace byla zvolena knihovna Tkinter pro její jednoduchost a rychlost naučení se práce s prvky. Nevýhoda Tkinter knihovny je nedostupnost „designeru“, jenž umožňuje prvky vkládat do okna a z toho pak následně vygenerovat „hotový“ kód na úpravu a přidání funkcionalit. Další nevýhoda je „horší“ či méně moderní vzhled.



Obrázek 12 Vzhled grafických prvků knihovny Tkinter [16]

4.1.2 Návrh uživatelského rozhraní

Grafické rozhraní je potřeba navrhnout tak, aby bylo uživateli/operátorovi příjemné, užitečné a co nejintuitivnější. Návrhů bylo několik, kde se časem vždy zjistilo, že se prvky nehodí na dané lokace nebo jsou třeba přidat úplně nové. Při přidávání se museli všechny ostatní předešlé prvku danému ručně přemístit (kód je psaný jako např. řádek: 1 sloupec: 2 -> prvek 2 apod…). V průběhu práce se grafické rozhraní muselo několikrát předělávat právě pro důvod lepší čitelnosti a orientaci v aplikaci.

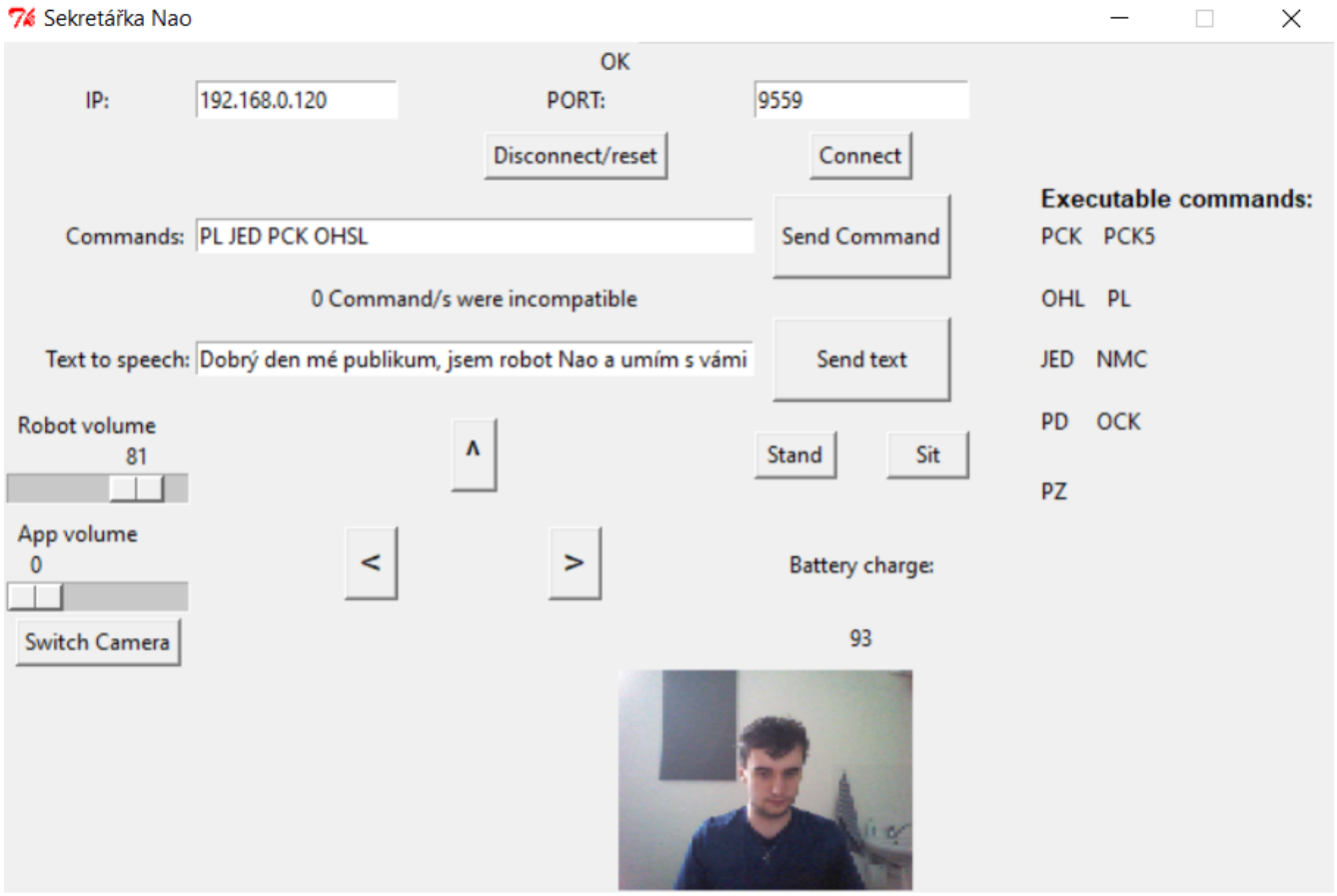


Obrázek 13 Grafické rozhraní před připojením se k robotovi

Dokud se operátor nepřipojí k robotovi pomocí uvedených kolonek (IP adresa a PORT) tak je UI zredukované pouze na tento vzhled viz. Obrázek 13 Grafické rozhraní před připojením se k robotovi.

Dále je třeba rozhraní navrhnout tak, aby nezabíralo zbytečné mnoho místa. Prvky jsou rozmístěny takovým způsobem, aby častěji využívané prvky byli lépe viditelné ve středu aplikace. Předpoklad je takový, že prvky pro text-to-speech a použití předpřipravených příkazů budou nejčastěji používány, umístíme je tedy po přihlášení do vrchní části aplikace, kde by měli být přehledně vidět pod sebou. Zpracování předpřipravených příkazů bude probíhat pomocí psaného textu do textového pole pro možnou kombinaci. Další často používaná vlastnost bude pravděpodobně (odhadem autora) možnost pohybu robota v prostoru. Pohyb robota bude docílen pomocí třech tlačítek každé indikující pohyb kupředu, doleva, doprava umístěných podobně jako šipky na klávesnicích. Toto ovládání je považováno za důležité a bude umístěno pod výše uvedenými prvky ve středu aplikace. To zanechává volný prostor nalevo a vpravo od tlačítek pro pohyb. Nalevo bude umístěno ovládání hlasitosti robota (jeho hlasitost mluvení) a zvuk aplikace. Napravo budou umístěny prvky pro ovládání poloh robota pro přepínání mezi „odpočinkovým“ stavem a aktivním. Provedení bude pomocí tlačítek „Crouch“ a „Stand“. Další velice důležitý prvek je záznam videa. Pro minimalizaci velikosti okna aplikace bude obraz umístěn v menší dostatečně vnímatelné velikosti na spodní část aplikace pod tlačítka pohybu do stran. Jako dodatečné prvky později implementované je přepínání mezi dolní (u „brady“ robota) a přední kamerou (mezi „očima“ robota), jenž bude umístěno pod ovládání hlasitosti. Důvod tohoto

umístění je předpoklad ne příliš častého využití a blízkost k pohybovým tlačítkům. Stav baterie je umístěn pod nastavováním poloh „Stand“ a „Crouch“. Tato informace je potřebná ale není dost zásadní na to, aby byla umístěna ve středu aplikace. Jako poslední dodatkové prvky jsou možnosti předpřipravených příkazů pro vykonání. Ty jsou umístěny jako nový sloupec v aplikaci napravo. Od zbytku prvků. Viz. Obrázek 14 Grafické rozhraní programu



Obrázek 14 Grafické rozhraní programu

Všechny elementy by měli být samo popisné. *Commands* pole umožňuje zadání a provedení předpřipravených příkazů, *Executable Commands* reprezentuje seznam možných použitelných příkazů. Příkazy jsou „pevně“ dány a pro změnu se musí ručně změnit obsah daných polí či je úplně odstranit. *Text to speech* slouží pro mluvení robota pomocí textu. *Robot volume* a *App volume* pro nastavení hlasitosti robota či aplikace samotné. *Switch camera* se přepíná mezi kamerami (dolní u „brady“ či přední mezi očima) a napravo je vidět stav baterie konkrétně na 93 procentech. Šipkami se ovládá otáčení či chůze robota dopředu. *Stand* a *Sit* pro změnu poloh robota. Nejvíce dole je vidět použitá přední kamera ve funkci.

Jako alternativní rozložení bylo zvažováno přidat živý záznam z video kamery do prostředku aplikace a rozvrhnout prvky tak, aby byli v okolí daného okna se záznamem. Toto řešení mohlo být optimálnější pro lepší vizuální zážitek. Aplikace by ale poté zabírala znatelně více plochy (při dodržení velikosti dosavadních prvků jako je to ve výše uvedeném *Obrázek 14 Grafické rozhraní programu*). Nevýhoda obou návrhů je nepřítomnost možnosti aplikaci zvětšit či zmenšit, jelikož postrádá vzorec potřebný pro přepočet velikostí prvků. Novější verze jazyku C# či knihovna Tkinter tuto možnost mají automatickou či lépe dostupnou pro úpravu.

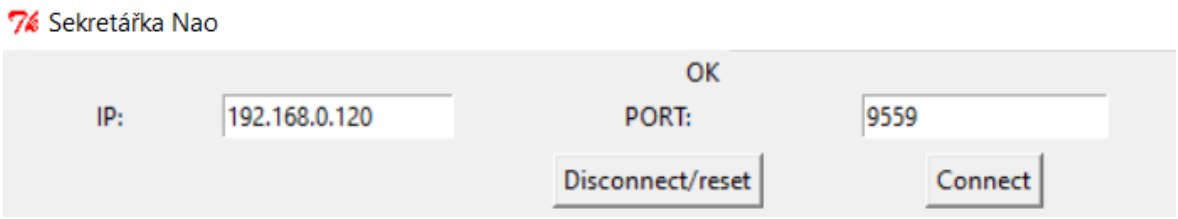
5 FUNKCE PROGRAMU

Programové funkce jsou psány do modulových částí pomocí metod, které provádějí určité zásadní funkčnosti programu odděleně od sebe. Bude tedy jednoduché metody upravit či přidat nové funkcionality připsáním nového bloku a připsání do inicializačního modulu. Například Zpracování zvuku a video jsou odděleny do plně jiných metod s upravitelností pro jednotlivé metody zpracování.

5.1 Připojení k robotovi

Připojení k robotovi Nao probíhá v programu pomocí IP adresy, kterou lze na robotovi nastavit ručně či pomocí DHCP služeb routeru ke kterému je připojen. Robota lze také přímo připojit do počítače či sítě pomocí ethernet portu jenž se nachází vzadu na hlavě.

K připojení k robotovi je také třeba znát port. Port nesmí být obsazen jiným procesem v robotovi či v počítači. Lze jej nakonfigurovat manuálně ale šance konfliktu s porty jsou poměrně malé, proto pro většinu případů nemusí být po pořízení robota měněn. Viz. Obrázek 13



Obrázek 15 Grafické rozhraní před připojením se k robotovi

5.2 Inicializace modulů

Moduly jsou programovány do jednotlivých metod. Je třeba je potom inicializovat co nejpřehlednějším a nejjasnějším způsobem. Pro to je nejvíce vhodná další metoda, jenž všechny požadované moduly inicializuje a připraví na použití. Zavede jejich funkci do paměti programu a provede jejich kód při zavolání či potřebě využití. Program také inicializuje moduly knihovny Naoqi, jenž jsou potřebné pro umožnění propojení a ovládání robota z místa spuštění programu [15]. Na pořadí inicializace modulů v určitých případech záleží. Jako první náležitost je třeba inicializovat všechny moduly Naoqi knihovny, jež budou použity zbytkem přidaných metod. Příkladem je pro použití chodících (nebo hýbacích) možností robota potřeba inicializovat nejdříve z knihovny Naoqi modul pro ovládání pohybů před použitím tohoto modulu v ovládání programu. Dalším úkolem inicializace je mít lepší přehled o závislostech mezi částmi metod programu, které nemusí být jinak jasné. Například pro ovládání hlasitosti robota je třeba nejdříve mít inicializovaný modul pro ovládání hlasitosti. Inicializace modulů z knihovny Naoqi probíhá následovně pomocí Pythonu:

```
battery = ALProxy("ALBattery", nao_ip, nao_port)
get_battery_charge()
```

Obrázek 16 Inicializace modulu pro stav baterie

Kde Obrázek 16 zavádí část programu pro práci s baterií robota. Následně se zavolá metoda *get_battery_charge()* pro zpracování stavu baterie z výše uvedeného modulu.

5.3 Chození

Robot má možnosti chůze nastavené na levou, pravou a přední stranu. Možnost chodit pozadu z bezpečnostních důvodů nemá, jelikož operátor nemá možnost vidět za robota bez své přítomnosti u něj.

Chůze je zařízena pomocí předem dostupných funkcí dostupné na robotovi. Předpřipravené možnosti pohybu [16] jsou dle dokumentace celkem tři. Ze tří možností byla vybrána metoda možnosti pohybu v určitém směru (za použití proměnných x a y jako osy) a za určité rychlosti. Konkrétně je rychlost omezena na přibližně 60 % z důvodu bezpečnosti robota, aby neměl tendenci se při rychlém pohybu převrátit a poškodit se. Obzvláště v nerovném terénu. Při zkoušení efektivity chůze byl robot schopen přejít přes pár kabelů na zemi. Avšak udržuje stejný úhel nohou a těla při chůzi, proto při pohybu do kopce či z kopce je vyšší tendence robota přepadnout kvůli neudržování optimálního těžiska. Robot svou stabilizaci udržuje za pomoci zpětné vazby z jeho kloubů a senzorů.

Robot má daleko rozsáhlejší možnosti úpravy pohybu jako jsou mnohé parametry a nastavení. Dá se i nadefinovat vlastní styl chůze či běhu, ačkoliv tyto způsoby jsou už poměrně pokročilé a je třeba znát meze robota, aby se nedokázal převrátit a poškodit. V tomto případě bude stačit klasický režim chůze bez potřeby jakýchkoliv úprav tohoto pohybu.

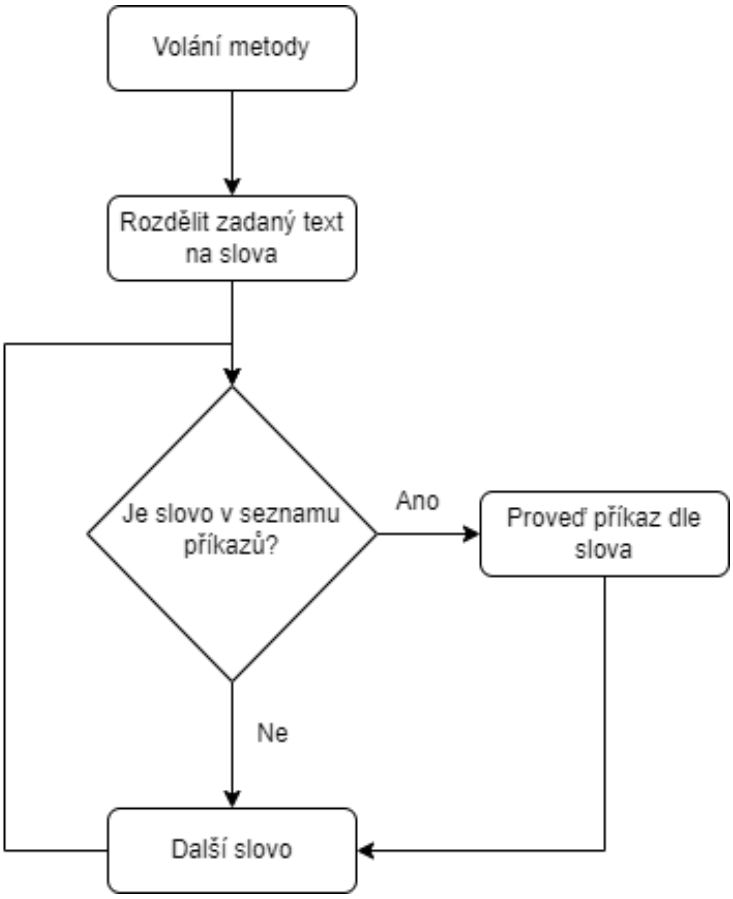
5.4 Text-to-speech

Text na řeč je opět prováděno pomocí předpřipravených funkcí robota. Je několik modulů pro jejich zpracování (*ALTextToSpeech, ALDialog, ALAnimatedSpeech*) a v tomto případě bude pro lepší interaktivní zážitek použit modul *ALAnimatedSpeech* [17], jenž dává možnost nastavení gestikulací při mluvení. Gestikulace mohou být nastaveny na mnoho režimů, například při mluvení určitých slov bude robot dělat konkrétní pohyby. Má také možnost obsahovat seznam určitých slov (omezeno zvoleným jazykem, čeština by měla být podporována ale ne plně), na který při vyslovení slova ze seznamu bude také provádět uložené gesto či pohyb. Možností upravitelnost je tedy mnoho, dokonce má i možnost vybírat pro jedno slovo náhodně z více gest pro vykonání dle nastavení. V tomto případě, aby robot nedělal stále stejné pohyby při řeči, je nastaveno provádění náhodných gest. Při mluvení robot kontroluje svou polohu a pokud je v poloze jiné než stojící, tak gestikulace nebude provádět. Důvod zákazu gestikulací v jiných polohách je riziko ztráty rovnováhy s následným možných pádem robota, k čemuž během testování došlo. Dalším důvodem volby tohoto způsobu zpracování řeči je také možnost provádění více akcí než jen mluvení robota. Gestikulacemi robot nabývá osobnějšího a lidštějšího dojmu. Operátor má také možnost volby určitých pohybů, pokud bude vědět, jak je provést pomocí dokumentace na [17]. Například při konkrétním zadání tohoto textu: *"Hello! ^start(animations/Stand/Gestures/Hey_1) Nice to meet you!"*. Se navíc kromě promluvení „*Hello! Nice to meet you!*“ provede mezi větami animace *Hey_1*. To dodává vysoké univerzálnosti i bez jiných podnětů jako jsou předpřipravené příkazy. Jediné omezení využití tohoto způsobu zpracování zvuku je zavilost na knihovně dostupných gest. Knihovnu lze ale rozšířit uživatelem dle dokumentace. Práce potřebná pro tyto rozšíření je ale náročnější s menším příspěvkem k zážitku.

5.5 Předpřipravené příkazy

Metoda pro vykonávání předpřipravených příkazů je řízena pomocí seznamu příkazů, které jsou v programu pevně (neměnitelné mimo úpravu kódu programu) zapsány. Program při požadavku provedení příkazu rozdělí zadaný text dle mezer na slova, která se porovnají se seznamem příkazů.

Pokud je slovo obsaženo v seznamu, provede se metoda daného příkazu. Tento postup se opakuje, dokud se neprovedou všechny příkazy obsažené na vstupu volání. Slova, která nejsou v seznamu příkazů budou porovnána a potom vynechána/ignorována.

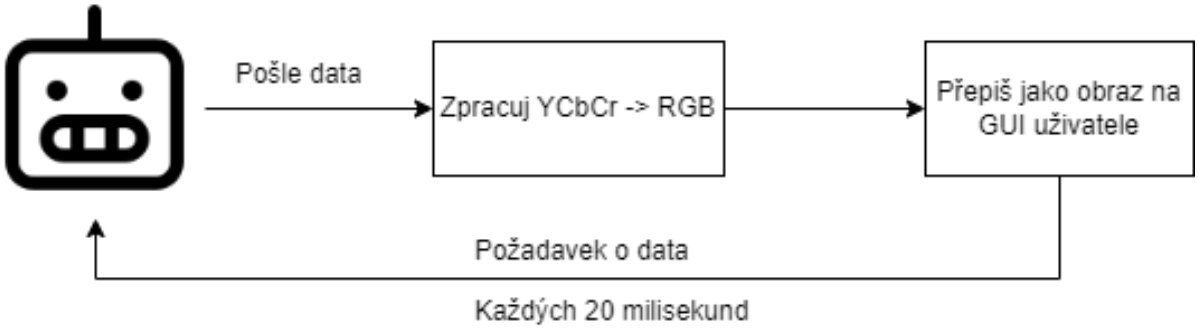


Obrázek 17 vývojový diagram předpřipravených příkazů

Příkazy mohou být dle limit jazyka Python jakékoliv. Příkaz může být ovládací sekvence pro robota či i dokonce jiná práce či program, pokud je spustitelný jazykem Python.
Při brzkých fázích vývoje program byla metody pro předpřipravené příkazy

.

5.6 Přenos videa z kamery



Obrázek 18 Diagram získávání a zpracování dat k videu

Pro získávání informací od robota je kvůli omezenému hardware třeba informace posílat v co nejméně zpracované podobě. Robot Nao umožňuje pořizování videa v několika rozlišeních i barevných formátech. Jelikož robot pořizuje video nativně ve formátu YCbCr422, bude nastaven režim posílání videa [18] na YCbCr. Tento formát je poté možno jednoduše převést na RGB, jenž je potřeba pro správné zobrazení barev na zařízení operátora. Rozlišení videa lze zvolit maximálně HD+ kvalitu 1280x960 pixelů, kvůli omezenému výkonu procesoru bude režim rozlišení obrazu nastaven na 640x480 pixelů.

Jelikož knihovna pro ovládání robota neumožňuje přístup k automatickému posílání videa pro živý přenos, bude třeba použít nepřímého způsobu získávání živého přenosu videa. To je provedeno pomocí požadavku na robota, aby zasílal jednotlivé obrazy. Robot zašle požadovaná data dle nastavení režimů uvedených výše. Z dat se uloží šířka, výška obrazu a samotná data o barvě jednotlivých pixelů. Tyto data v barevném schématu YCbCr se v programu na počítači operátora zpracují na RGB pro správné a přesné zobrazení barev. Požadavek o data probíhá dle nastavení v programu každých 20 milisekund, jenž by odpovídalo 50 snímků za sekundu. Obraz operátora je ale omezen na 30 snímků za sekundu kvůli frekvenci obnovování obrazů na robotovi. Tento způsob zpracování video je tedy přímo ovlivněn prodlevou v připojení k robotovi, kde při větší odezvě bude přímo ovlivněn počet snímků za sekundu. Při testování připojení se před Eduroam na robota byla odezva mnohem větší (+2-3 sekundy) a obnova obrazu byla zmenšena na přibližně 4 snímky za sekundu.

5.7 Přijímání zvuku

Zařízení zvuku na roboti je docílit přenášení dat živě s co nejmenší prodlevou. K tomu je třeba prozkoumat dokumentaci knihovny pro možné způsoby. Způsob, který popisuje knihovna *Naoqi* zobrazuje pouze jeden [19]. Tento způsob se nepodařilo i přes mnohé týdny snažení zprovoznit pravděpodobně kvůli technickým detailům, které v dokumentaci nejsou zmíněny nebo nejsou udělány pro jazyk Python.

Jako alternativní způsob byl nalezen modul v robotovi Nao s názvem *Gsteamer* [20]. Tento modul umožňuje přístup k mnoha nástrojům pro úpravu a managment zvuku. Zahrnuje to přehrávání, nahrávání, upravování či streamování zvuku. V tomto případě bude potřeba jeho možnost živého vysílání a zpracování zvuku. Cílem tohoto způsobu bude vytvořit v robotovi server, na kterém se bude vysílat jeho audio z mikrofonu s co nejmenší prodlevou určenou výpočetní kapacitou robota. Jako zásadní modul obsažen v rámci *Gsteamer* pro požadovaný výsledek je *PulseAudio* [21] neboli *pulsesrc* [22], jenž umožňuje vytvoření serveru na zařízení (v našem případě na robotovi) s možností připojení se pomocí IP adresy, komunikačního protokolu TCP [23] a portu.

Pomocí knihovny Paramiko [24], jenž slouží k implementaci SSH protokolu do jazyka Python se připojíme k roboti Nao a sestavíme příkaz pro aktivaci vysílání zvuku pro operátora k odchycení.

Použitý příkaz je následující: *gst-launch-0.10 pulsesrc ! audioconvert ! vorbisenc ! oggmux ! tcpserversink port=1234*. *Gst-launch* slouží jako báze pro použití příkazu *pulsesrc*, jenž vytvoří na robotovi server pro vysílání zvuku zachyceného jeho mikrofony. *Audioconvert* přetransformuje audio data z mikrofonu na specifické zvukové formáty. *Vorbisenc* enkóduje získaná data na formát *Vorbis* [25], jenž je neproprietární nepatentovaný audio formát. Použitím tohoto formátu se data zkomprimují a šetří se tím vytížení sítě. *Oggmux* ke konci slouží pro sloučení video a audio (obecně, v konkrétní situaci chceme pouze audio) vysílání do .ogg souborů, to slouží pro konečné přijímání dat na operátorském zařízení pomocí knihovny VLC.

Nevýhoda tohoto řešení je závislost na modulech *Gstreamer* a VLC, jenž při již nepodporované verzi Python 2.7 (a tím i starým verzím rozšíření) bude mít limitovanou životnost či chyby, které již byli opraveny v novějších verzích.

5.8 Reakce na stisk tlačítka

Obrázek 19 Vývojový diagram stisku tlačítka na hlavě

Robot má možnost interakce s jeho okolím pomocí stisku zadního tlačítka na jeho hlavě mimo kontrolu operátora. Po stisknutí daného tlačítka na hlavě robot přednese jeden ze seznamu faktů. Fakty jsou obsaženy jako součást programu a je jich konkrétně 5. Po přednesení všech faktů robot řekne, že již žádný fakt nemá a bude povídat ty předchozí

5.9 Ovládání hlasitostí

Ovládání hlasitosti zvuků, co vydá robot je poměrně přímočaré. Pro inicializaci je třeba použít modul ALAudioDevice a jeho volací metody a funkce [26]. V tomto modulu je obsáhlá metoda pro změnu hlasitosti v měřítku 0 až 100 %. Změna zvuku, co vydává aplikace operátorovi je pro změnu závislá na modulu, jenž vysílá zvuk z robota mikrofonů. Použitá knihovna VLC pak obsahuje metodu pro změnu zvuku obdobně v měřítku 0 až 100 %. Změny zvuku jsou v uživatelském rozhraní prováděny pomocí posuvných tlačítek (slidebarů), jenž jsou dostupné z Tkinter knihovny. Při posunu tlačítka dojde k automatickému volání metody pro změnu hlasitosti. Frekvence volání metody probíhá vnitřně při téměř každé změně hodnoty (neprovádí se v případě, kdy uživatel posouvá tlačítkem velice rychle a program by pak měl problém obsluhovat každé volání metody).

5.10 Stavové informace

Robot disponuje širokým spektrem dostupných informací o sobě. Mnoho z těchto informací (například napětí v motorech, natočení jednotlivých končetin) nejsou pro operátora relevantní. Téměř všechny informace o roboti je možno snímat pomocí modulu *ALMemory* [27], jenž poskytuje způsob pro přístup i úpravu (ačkoliv to může být nebezpečné pro robota) ke všem podstatným informacím robota. Informace o robotovi lze získávat voláním určitých funkcí, které poté navrátí požadované hodnoty zpět. Tyto funkce ale také pracují se zmiňovaným modulem [27]. Výhodou těchto funkcí je menší pravděpodobnost poškození či provedení nechtěných změn na robotovi jenž je velmi kladná výhoda a preferovaný přístup, pokud je tak možno provést operace.

Aplikace po robotovi pravidelně požaduje informaci o jeho stavu baterie (přibližně každou sekundu) pro informaci operátora o časových možnostech robota, kdy může být odpojen od zdroje nabíjení a být pak plně bez připojených kabelů. Robot také snímá obličeje v jeho zorném poli, pokud obličej najde, vypíše informaci o tom, že detekuje obličej a přiradí jemu unikátní ID.

Program také disponoval řízením tzv. zón *ALEngagementZones* [28], na které se by měl možnost reagovat bez podmětu operátora. Příkladem takové interakce by bylo při vkročení člověka do jeho zóny a pozdrav s pohledem na účastníka v jeho blízkosti. Modul nakonec nebyl použit z důvodu nepotřeby pro splnění zadání a časového vytížení řešení jiných problematik.

Dalším nevyužitým modulem bylo využití ultrasonických senzorů, jenž by snímali vzdálenost objektů v blízkosti robota. Využitím tohoto modulu mělo být obdobné jako výše zmíněná také nevyužitá část. Vzhledem k větší kompletnosti a využitelnosti jiných modulů nebyla tato část programu využita. Senzory se také dalo využít na zastavení robota při riziku nárazu do překážky před ním, robot by ale měl sám disponovat takovou možností automaticky reagující na danou situaci.

6 ZÁVĚR

V rámci bakalářské práce bylo splněno všech požadavků zadání. Rešerše ukázala, že v českém a anglickém jazyce (především na západní straně Země) je malá dostupnost obdobných úloh, žádná úloha není zpracováním či obsahem shodná. Vybavení robota Nao je rozsáhlé a bohaté funkcemi jak hardwarově, tak softwarově. Bohužel kvalita (spíše stáří) hardware komponent robota omezuje od využití jeho plného potenciálu, obzvláště ve výpočetní oblasti. V brzkých fázích byl program navrhnut poměrně rychle a bez větších obtíží. Po implementaci více funkcionalit a grafických elementů byla nezkušenost s návrhem obdobných aplikací pro dynamické ovládání znát, jelikož bylo mnoho instancí, kde se rozdělaná aplikace musela pracně měnit a přemisťovat grafické elementy. Kdyby byl návrh na začátku lépe provedený, tak by byl průběh práce rychleji proveden. Ovládání a získávání informací robota je provedeno úspěšně i přes poměrně omezenou dostupnost dokumentace k dynamickému ovládání robota Nao. Primárním zdrojem informací byla samotná dokumentace. Robot je responzivní, odezva videa a zasílání příkazů je téměř instantní (závislé na kvalitě vzdáleného připojení). Předpřipravené příkazy fungují, jak mají a program je jednoduše rozšiřitelný. Při zpracovávání zvuku nebylo docíleno zpracování způsobem, jaký je uveden v dokumentaci výrobce. Bylo proto nalezeno jiné řešení, kde se vytvoří na robotovi server, na kterým je zvuk zpracován. Ten se následně vysílá do jeho okolí a operátora program jej zachytí a následně zpracuje. Program disponuje více funkcemi, než jsou zobrazené na uživatelském rozhraní. Jednou z těchto funkci je detekování obličejů a vypsáním jeho identifikačního kódu. Detekce obličejů nakonec není nijak využita kvůli změně pohledu na chtěnou interakci robota s jeho protějškem (původním cílem byl pozdrav při detekci obličeje).

Jako možné vylepšení by mohl být navrhnut algoritmus, který dokáže robotovi volně hýbat s hlavou dle požadavků operátora pro lepší orientaci v prostoru. Dalším vylepšením by bylo jiné provedení zpracování zvuku mimo robota (tedy na počítači operátora), to by eliminovalo zpoždění, jenž je zaviněno omezeným výpočetním výkonem. Jako poslední větší vylepšení by mohla být možnost robota ovládat vzdáleným zařízením jako je ovladač, virtuální realita či propojení obou těchto možností.

POUŽITÁ LITERATURA

[1] NAOqi Framework. *http://doc.aldebaran.com/.* [Online] [Citace: 10. 05 2022.] http://doc.aldebaran.com/1-14/dev/naoqi/index.html.

[2] Bremmer, Paul a Leondars, Ute. Iconic Gestures for Robot Avatars, Recognition and Integration with Speech. 17. 2 2016, Sv. 7, 183.

[3] OlivierMichel. Aldebaran Nao. *ros.org.* [Online] 27. 3 2019. [Citace: 19. 5 2022.] http://wiki.ros.org/nao.

[4] Cocks, Naomi, Morgan, Gary a Kita, Sotaro. Iconic gesture and speech integration in younger and older adults. Gesture, 1 2011, Sv. 11, 1.

[5] Aldebaran Nao vzdálené ovládání pomocí Dell Latitude XT3. [Video]. místo neznámé, Itálie : Notebook Italia, 2011.

[6] Muhammet Yunus Şeker; Mehmet Özdemir. Nao Robot Avatar. *Department of Computer Engineering.* [Online] Vysoká škola Bogazici, 17. 7 2017. [Citace: 10. 5 2022.] https://www.cmpe.boun.edu.tr/content/nao-robot-avatar-0.

[7] Aldebaran Softbank Group. NAO H25. *Aldebaran documentation.* [Online] [Citace: 11. 5 2022.] http://doc.aldebaran.com/2-1/family/nao_h25/index_h25.html.

[8] —. Motherboard. *Aldebaran Documentation.* [Online] [Citace: 11. 5 2022.] http://doc.aldebaran.com/2-1/family/robots/motherboard_robot.html#robot-motherboard.

[9] —. Nao - Video Camera. *Aldebaran Documentation.* [Online] [Citace: 11. 5 2022.] http://doc.aldebaran.com/2-1/family/robots/video_robot.html#robot-video.

[10] —. H25 - Contact and tactile sensors. *Aldebaran documentation.* [Online] [Citace: 11. 5 2022.] http://doc.aldebaran.com/2-1/family/nao_h25/contact-sensors_h25.html.

[11] —. OpenNAO - NAO OS. *NAO Software 1.14.5 documentation.* [Online] [Citace: 11. 5 2022.] http://doc.aldebaran.com/1-14/dev/tools/opennao.html.

[12] Stallman, Richard a Kučera, František. Linux a systém GNU. *Operační systém GNU.* [Online] gnu.org, 2. 11 2021. [Citace: 11. 5 2022.] https://www.gnu.org/gnu/linux-and-gnu.cs.html.

[13] Aldebaran Softbank Group. Your first steps in Choregraphe. *NAO Software 1.14.5 documentation.* [Online] [Citace: 13. 5 2022.] http://doc.aldebaran.com/1-14/software/choregraphe/choregraphe_first_steps.html.

[14] —. Predefined postures. *Aldebaran documentation.* [Online] [Citace: 11. 5 2022.] http://doc.aldebaran.com/2-1/family/robots/postures_robot.html#robot-postures.

[15] Group, Aldebaran Softbank. ALAutonomousLife - Advanced. *Aldebaran documentation.* [Online] Softbank Robotics. [Citace: 11. 5 2022.] http://www.bx.psu.edu/~thanh/naoqi/naoqi/core/autonomouslife_advanced.html.
[16] Python GUI, PyQt vs TKinter. *DEV community.* [Online] 31. 10 2019. https://dev.to/amigosmaker/python-gui-pyqt-vs-tkinter-5hdd.
[17] Robotics, Softbank. Python SDK - Overview. *Softbank Robotics Documentation.* [Online] Softbank Robotics. [Citace: 11. 5 2022.] http://doc.aldebaran.com/2-5/dev/python/intro_python.html.
[18] Group, Aldebaran Softbank. Locomotion control. *Aldebaran Documentation.* [Online] [Citace: 11. 5 2022.] http://doc.aldebaran.com/2-1/naoqi/motion/control-walk.html.
[19] —. ALAnimatedSpeech. *Aldebaran documentation.* [Online] [Citace: 11. 5 2022.] http://doc.aldebaran.com/2-1/naoqi/audio/alanimatedspeech.html.
[20] Robotics, Softbank. ALVideoDevice. *Softbank Robotics Documentation.* [Online] [Citace: 11. 5 2022.] http://doc.aldebaran.com/2-5/naoqi/vision/alvideodevice.html.
[21] —. ALAudioDevice. *Softbank Robotics Documentation.* [Online] [Citace: 11. 5 2022.] http://doc.aldebaran.com/2-5/naoqi/audio/alaudiodevice.html.
[22] Gstreamer. GStreamer. *Gstreamer.* [Online] 11. 1 2001. [Citace: 11. 5 2022.] https://gstreamer.freedesktop.org/documentation/gstreamer/gst.html?gi-language=c.
[23] Poettering, Lennart, a další. Pulse Audio. *freedesktop.org.* [Online] 2004. [Citace: 11. 5 2022.] https://www.freedesktop.org/wiki/Software/PulseAudio/.
[24] Gstreamer. pulsesrc. *Gstreamer.freedesktop.org.* [Online] [Citace: 11. 5 2022.] https://gstreamer.freedesktop.org/documentation/pulseaudio/pulsesrc.html?gi-language=c.
[25] Cerf, Vint, Tomlinson, Ray a Kahn, Bob. Transmission Control Protocol. *Wikipedia.* [Online] 1989. [Citace: 11. 5 2022.] https://cs.wikipedia.org/wiki/Transmission_Control_Protocol.
[26] Bitprophet. Paramiko. *Paramiko.org.* [Online] 2022. [Citace: 11. 5 2022.] https://www.paramiko.org/.
[27] Xiph.org Foundation. Vorbis audio compression. *Xiph.org.* [Online] 1994-2016. [Citace: 11. 5 2022.] https://xiph.org/vorbis/.
[28] Softbank Robotics. ALAudioDevice API. *Softbank Robotics Documentation.* [Online] [Citace: 11. 5 2022.] http://doc.aldebaran.com/2-5/naoqi/audio/alaudiodevice-api.html.
[29] Aldebaran Softbank Group. ALMemory. *Aldebaran Documentation.* [Online] [Citace: 11. 5 2022.] http://doc.aldebaran.com/2-1/naoqi/core/almemory.html.
[30] —. ALEngagementZones. *Aldebaran Documentation.* [Online] [Citace: 11. 5 2022.] http://doc.aldebaran.com/2-1/naoqi/peopleperception/alengagementzones.html.

[31] M15000089, Miroslav Eichler:. Využití distupných senzorů robota Nao pro detekci objektů a mapování okolí. Liberec : Technická univerzita V Liberci, 2018.
[32] Herrmann, Michael. PyQt5 tutorial. *fman build system.* [Online] 2016. [Citace: 13. 5 2022.] https://res.cloudinary.com/practicaldev/image/fetch/s--Xp47dkfA--/c_limit%2Cf_auto%2Cfl_progressive%2Cq_auto%2Cw_880/https://external-content.duckduckgo.com/iu/%3Fu%3Dhttps%253A%252F%252Fbuild-system.fman.io%252Fstatic%252Fpublic%252Fimg%252Fpyqt5-example.
[33] Inbar, Elad. *NAO Evolution V5.* [Online] místo neznámé : RobotLAB, 2014.

Příloha 1: Návod k obsluze

Příloha 2: USB Flash disk

Návod k obsluze:

Program je třeba spustit ve vývojovém prostředí (například Visual Studio Code či Pycharm) s nainstalovaným jazykem Python 2.7 32bit a k němu (v hlavičce souboru) nainstalovat použité knihovny. Po spuštění skriptu se otevře připojovací okno pro operátora. Po zadání správné IP adresy a portu se operátor připojí k robotovi. Má pak k dispozici všechny ovládací prvky, jenž aplikace nabízí.

Obsah přiloženého USB Flash disku

- Program bakalářské práce (python - .py soubor)
- Bakalářská práce ve formátu .pdf