VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY
DEPARTMENT OF CONTROL AND INSTRUMENTATION

PLÁNOVÁNÍ TRAJEKTORIE ROBOTICKÉHO RAMENE
PLANNING OF ROBOTIC MANIPULATOR TRAJECTORY

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR
Ondra Zbožínek

VEDOUCÍ PRÁCE
SUPERVISOR
Ing. Adam Chromý

BRNO 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Automatizační a měřicí technika**
Ústav automatizace a měřicí techniky

Student: Ondra Zbožínek ***ID:*** 174432
Ročník: 3 ***Akademický rok:*** 2016/17

NÁZEV TÉMATU:

Plánování trajektorie robotického ramene

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je vytvořit software pro rychlou, jednoduchou a názornou definici trajektorie robotického manipulátoru. Program bude umožňovat několik způsobů zadávání, vizualizaci a testování přímo s robotickým ramenem.

1. Seznamte se se stávajícím C# driverem pro šestiosý manipulátor EPSON C4 a s možnostmi jeho řídicí jednotky.
2. Oživte nový manipulátor EPSON C4 a zprovozněte na něm tento driver.
3. Rozšiřte stávající driver o další způsoby pohybu dle zadání vedoucího.
4. Navrhněte a vytvořte softwarový nástroj pro rychlé a názorné definování trajektorie pohybu robotu různými způsoby (skriptování, 3D myš, apod.) umožňující vizualizovat navrženou trajektorii nebo ji přímo realizovat robotickým manipulátorem.

DOPORUČENÁ LITERATURA:

[1] A. Hejlsberg, M. Torgersen, S. Wiltamuth, and P. Golde, The C# Programming Language (Covering C# 4.0). Addison-Wesley Professional, 2010.

[2] R. P. Paul, Robot Manipulators: Mathematics, Programming, and Control : the Computer Control of Robot Manipulators. Richard Paul, 1981.

Termín zadání: 6.2.2017 ***Termín odevzdání:*** 29.5.2017

Vedoucí práce: Ing. Adam Chromý
Konzultant:

doc. Ing. Václav Jirsík, CSc.
předseda oborové rady

Abstrakt

Tato bakalářská práce se zabývá oživením robotického manipulátoru EPSON C4 a jeho řídící jednotky RC700-A. Zprovoznění komunikačního protokolu TCP/IP pro komunikaci řídící jednotky s knihovnou, implementovanou v jazyce C# .NET. Vytvoření SW nástroje pro rychlé a jednoduché plánování trajektorie pomocí ovladačů Razor Hydra.

Klíčová slova

robotický manipulátor EPSON C4, řídící jednotka RC700-A, vývojové prostředí EPSON RC+ 7.0, komunikační protokol TCP/IP, programování C# .NET, plánování trajektorie robotu, ovladače Razor Hydra

Abstract

This bachelors work deals with commissioning of a robotics manipulator EPSON C4 and a controller RC700-A. The work also deals with commissioning of a communications protocol TCP/IP for communication between a controller and driver which is programmed in C# .NET. Creation of SW tool for fast and simple planning trajectory, using Razor Hydra controllers.

Keywords

robotic manipulator EPSON C4, controller RC700-A, IDE EPSON RC+ 7.0, comunication protocol TCP/IP, programming C# .NET, robots trajectory planning, Razor Hydra controllers

Bibliografická citace:

ZBOŽÍNEK, O. Plánování trajektorie robotického ramene. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2017. 57 s. Vedoucím bakalářské práce Ing. Adam Chromý.

Prohlášení

„Prohlašuji, že svou závěrečnou práci na téma Plánování trajektorie robotického ramene jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne 29. května 2017 .....

podpis autora

Poděkování

Děkuji vedoucímu bakalářské práce Ing. Adamu Chromému za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé bakalářské práce.

V Brně dne 29. května 2017 ……

podpis autora

Obsah

ÚVOD

Obsahem bakalářské práce je seznámení se s celkovým konceptem robotického manipulátoru Epson C4 a jeho řídící jednotky Epson RC700-A. Manipulátor uvolněn pro tuto práci nebyl dříve použit v žádném provozu a tudíž je potřeba projít celkovou instalací produktu (od navrhnutí bezpečnostních okruhů až po vytvoření komunikace se C# driverem použitém na stávajících modelu Epson C3).

Vzhledem k vytíženosti robota je potřeba vytvořit ucelený návod na instalaci a vytvoření spojení mezi robotem a vývojovým prostředím EPSON RC+. Také zprovoznění komunikace s vnější řídící knihovnou EpsonC3Com využívanou i jako stejnojmenný SW nástroj pro ovládání robota.

Práce se zabývá také programováním v jazyce SPEL+, kde je potřeba naimplementovat rozšiřující funkce k dosažení hladšího a kontinuálního pohybu robota řízeného přes výše zmíněnou komunikační knihovnu.

Dalším řešeným problémem v bakalářské práci je plánování trajektorie manipulátoru. V současné době aktuální SW nástroj je schopen jednoduchého pohybu z aktuálního bodu do bodu předepsaného souřadnicemi nebo otočením kloubu/kloubů. Tuto omezenost je potřeba eliminovat v podobě upravení SW nástroje, který bude schopen plánování delších mnohobodových trajektorií a bude je schopen ukládat do nějaké dlouhodobé paměti.

V neposlední řadě práce rozebírá rozšiřující možnosti plánování trajektorie v podobě např. 3D myši, skriptování atp. Tato aplikace by měla uživateli umožnit, buď vizualizaci nebo přímé realizování navržené trajektorie. Zároveň by tato aplikace měla být opakem pracného zadávání či manuálního inkrementování hodnot k pohybu manipulátoru.

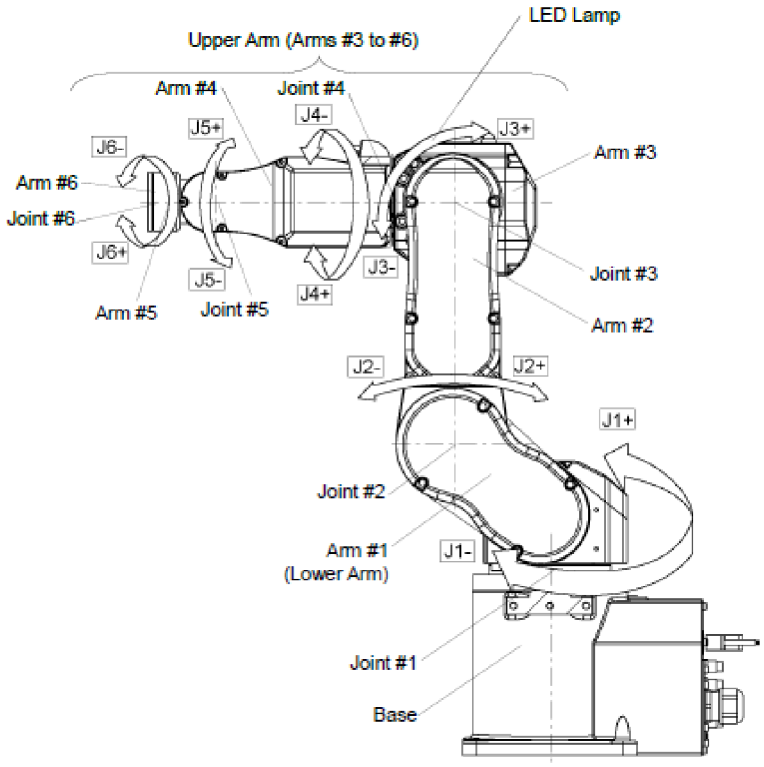
1. ROBOT EPSON C4

1.1 Základní vlastnosti

Jedná se o model C4-A601S, robotický manipulátor, který je dalším nástupcem šestiosých robotů od společnosti Epson a tím nahrazuje starší model Epson C3. Na první pohled se vyznačuje svými menšími rozměry, které ho právě odlišují od běžných průmyslových robotů a umožňují aplikaci v těsných prostorech. I přes jeho malou velikost je robot schopen zdvihnout břemena do hmotnosti 4kg (v určitém stavu až 5kg), a pracovat s nimi s poměrně velkou rychlostí a přesností.[8]

Robot se vyznačuje především svou relativní opakovatelností ±0,02mm a rychlou průměrnou dobou cyklu, která dosahuje až 0,37s, což je např. rychlejší než mnoho jiných SCARA (G/RS/LS Series) konkurentů od společnosti Epson. Rychlosti jednotlivých kloubů můžeme hodnotit také nadprůměrně, kde jsme schopni dosáhnout rychlostí 450-750˚/s (viz tabulka č.1).[8]

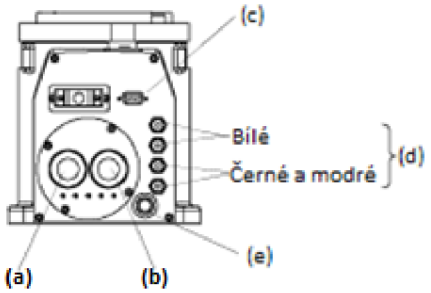
1.2 Popis modelu C4-A601S



Obrázek 1: Popis robotického manipulátoru Epson C4-A601S

	Max. rychlost [˚/s]	Max. pohybový rozsah [˚]	Rozlišení [˚/puls]	Výkon motoru [W]
Joint #1	450	170	0,0000343	400
Joint #2	450	-160 do +65	0,0000343	400
Joint #3	514	-51 do +225	0,0000392	150
Joint #4	555	±200	0,0000423	50
Joint #5	555	±135	0,0000423	50
Joint #6	720	±360	0,0000549	50

Tabulka 1: Specifické vlastnosti modelu Epson C4-A601S [8]



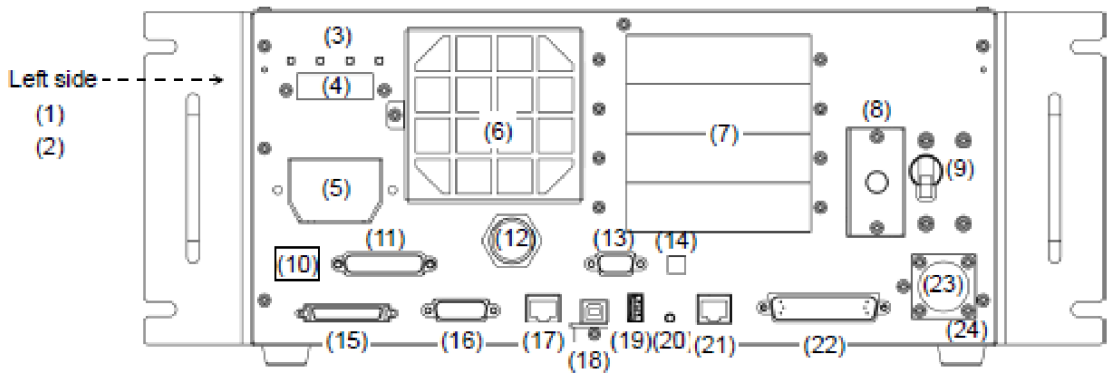
Obrázek 2: Zadní panel robotického manipulátoru Epson C4-A601S

Na zadním panelu můžeme najít porty pro připojení:

a) Signálový kabel např. pro detekci pozice motorů.
b) Napájecí kabel pro napájení manipulátoru.
c) Uživatelský kabel (9-pin D-sub konektor) pro připojení I/O funkcí v reálném čase.
d) 2 bílé a 2 černé nebo modré porty pro připojení pneumatický kabelů o průměru 4mm (barevné rozlišení závisí na přístupovém čase).
e) Vyfukovací port pro pneumatické kabely o průměru 8mm.

1.3 Popis řídící jednotky RC700-A

Řídící jednotka RC700-A se vyznačuje svou otevřenou architekturou a snadným použitím v průmyslu. Stará se o plynulý pohyb manipulátoru, rychlost zrychlovací a zpomalovací odezvy a v neposlední řadě se zasluhuje o průměrnou dobu cyklu. Řídící jednotka je schopna bezproblémové komunikace s PC prostřednictvím vývojového prostředí EPSON RC+ 7.0. Připojení řídící jednotky k PC je možné přes USB kabel a po následné konfiguraci prostředí i přes Ethernet rozhraní.[1]



Obrázek 3: Zadní panel řídící jednotky RC700-A

1) Sériové číslo jednotky.
2) Modelová čísla robotických manipulátorů, která jsou kompatibilní s jednotkou.
3) **LED** - Indikuje stávající operační mód (TEST, TEACH, AUTO a PROGRAM mód).
4) **7-segmentový display** – Obsahuje 4 číslice, které zobrazují stav jednotky (chybové číslo, stav nouzového tlačítka nebo bezpečnostních dveří).
5) **M/C POWER konektor** – Napájecí konektor řídící jednotky.
6) **Ventilátorový filtr** – Ochranný filtr v přední části ventilátoru k prachové ochraně.
7) **Rozšiřovací sloty** – Je možné použít pro rozšiřovací karty (I/O, RS-232C, PG).
8) **Baterie** – Lithiová baterie pro případné datové zálohování.
9) **Vypínač** – Pro zapnutí či vypnutí jednotky.
10) Ukazatel kompatibilního manipulátoru.
11) **Nouzový konektor** – Tento konektor je používán pro vstupní/výstupní signály pro nouzové tlačítko a spínače ochranných dveří.
12) **TP Port –** Připojení ručního ovládacího panelu
13) **RS-232C port** – Používán pro RS-232C komunikaci s externími zařízeními.
14) **Přepínač enkodéru napěťové úpravy** – Přepínač ke správné úpravě napětí podle délky napájecího kabelu.
15) **M/C signálový konektor** – Používán např. pro detekci pozice motorů.
16) **R I/O konektor** – Používán pro vstupní signály pro I/O funkce v reálném čase.
17) **OUT konektor** – Pro připojení pohonné jednotky.
18) **USB port** – Slouží k **offline** připojení k PC.
19) **Paměťový port** – Připojení běžného paměťového USB pro zálohování.

20) **Spouštěcí spínač** – Tento spínač je pro zálohovací funkci jednotky, která využívá paměťové USB.
21) **LAN port** –Propojuje řídící jednotku a PC přes ethernetový kabel.
22) **I/O konektor** – Je možné připojení vstupních/výstupních zařízení (24 vstupů a 16 výstupů)
23) **AC IN** – Konektor pro napájení ze střídavé sítě 200VAC.
24) Informační štítek řídící jednotky (např. sériové číslo).

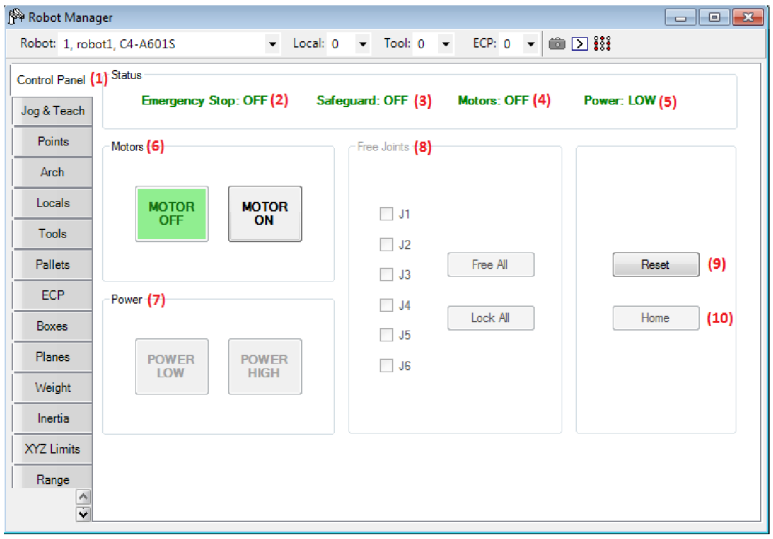
1.4 EPSON RC+ 7.0

EPSON RC+ 7.0 Project Management and Development Environment je nezbytný stavební prvek pro vývoj a implementaci programů pro řídící jednotku. Prostředí je kompatibilní se systémy Microsoft Windows XP, Vista, 7 a 8.1. Komunikace probíhá prostřednictvím USB nebo Ethernet. Software umožňuje propojení více řídících jednotek k jednomu PC a současně je spravuje. V prostředí je možnost použití přednastavených funkcí a nástrojů, které výrazně ulehčují práci s robotem a příslušnou řídící jednotkou.

1.4.1 Robot Manager

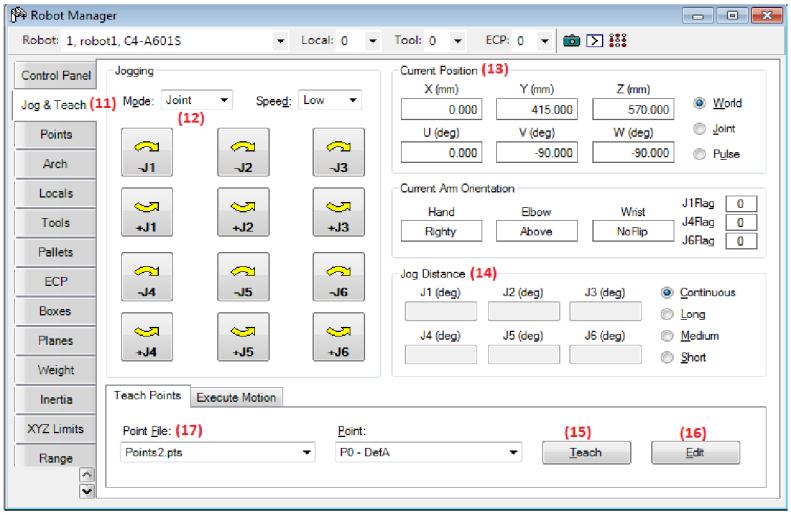
Je nástroj, který zpřehledňuje práci s robotem a umožňuje uživateli použití základních funkcí bez implementace jakéhokoli kódu nebo příkazu (eliminuje použití příkazové řádky vývojového prostředí).

Ve výchozí kartě **„Control Panel“**(1) dostáváme informace o bezpečnostním okruhu (2), ochranných prvcích robota (3), stavu motorů (4) a aktuálního výkonového módu (5). Je umožněno pomocí přepínacích tlačítek přímo měnit stav motorů (6) či výkonového módu (7). Dále je možné individuální nebo celkové uvolnění/sepnutí elektromagnetických brzd kloubů robota (8). V neposlední řadě karta obsahuje tlačítko k resetování všech stavů jednotky (9) a tlačítko k pohybu do přednastavené domovské pozice (10).



Obrázek 4: Robot manager - Control panel

V kartě **„Jog & Teach“** (11) se manuálně ovládá pohyb robota a učí se novým pozicím, které je možné si následně uložit a pracovat s nimi později v programu. Ovládání pohybu je možné dvěma způsoby. Prvním způsobem **(Mode → World)** (12) je robot ovládán pomocí **X, Y, Z, U, V, W** souřadnic v 3D prostoru (pracovní prostor manipulátoru). Druhý způsob **(Mode → Joint)** (12) umožňuje individuální natočení jakéhokoli kloubu manipulátoru. Při jakémkoli pohybu manipulátoru jsou sledovány reálné hodnoty jeho pozice a to jak v souřadnicovém tvaru (13), tak v úhlu natočení jednotlivých kloubů (14). Pro naučení nového bodu stačí stisknout tlačítko **Teach** (15) nebo editovat (16) již uložený bod, kde se pozice pojmenuje a přidá se případně popis. Tyto pozice se ukládají do speciálního souboru **pts** (17).

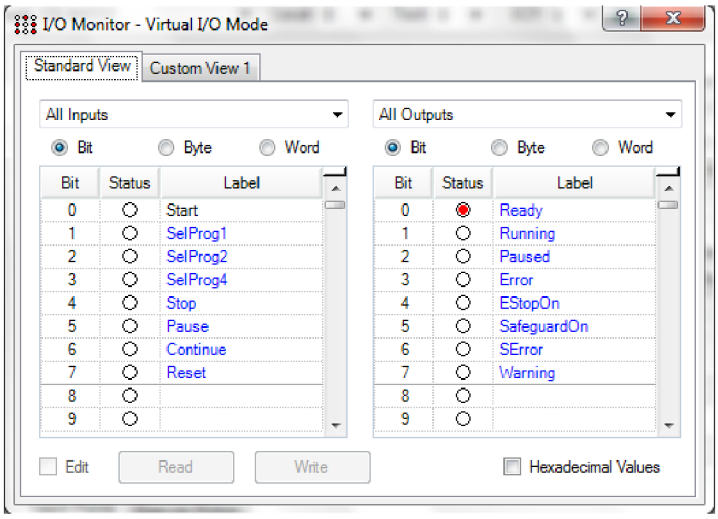


Obrázek 5: Robot Manager - Jog & Teach

Nástroj umožňuje mnoho dalších funkcí jako např. správu již vytvořených pracovních pozic, převod stupňů natočení kloubů na inkrementy, nastavení hmotnosti břemene, nastavení offset pozicí pro různé velikosti chapadel aj.

1.4.2 I/O Monitor

Jestliže jsou k řídící jednotce připojeny vstupní/výstupní zařízení nebo je potřeba robota řídit přes virtuální I/O zařízení, tak je potřeba nejdříve jejich konfigurace přímo v řídící jednotce pomocí prostředí a následně použitím nástroje **„I/O Monitor“** je možné tyto zařízení monitorovat v reálném čase. V tomto nástroji je také možné si jednotlivé I/O zařízení pojmenovat a měnit jejich digitální hodnotu.

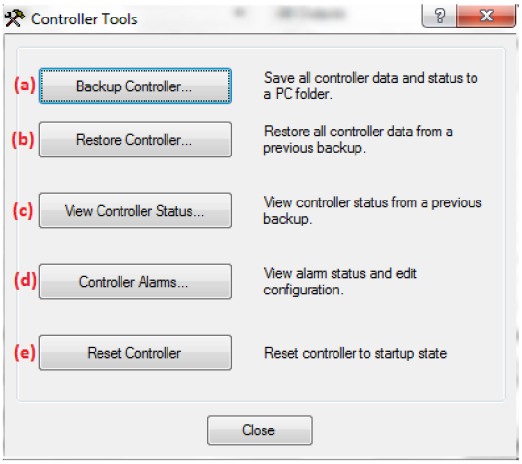


Obrázek 6: Nástroj I/O Monitor

1.4.3 Controller tools

Tento nástroj obsahuje 5 důležitých funkcí pro bezpečný chod robota.

a) **Záloha řídící jednotky** – Uloží všechna data a stav řídící jednotky do PC souboru.
b) **Obnova řídící jednotky** – Obnoví data a stav řídící jednotky z předem zálohovaného PC souboru.
c) **Stav řídící jednotky** – Zobrazí stav řídící jednotky z předchozí zálohy.
d) **Varování řídící jednotky** – Zobrazí varovný stav a upraví konfiguraci jednotky.
e) **Resetování řídící jednotky** – Restartuje řídící jednotku.

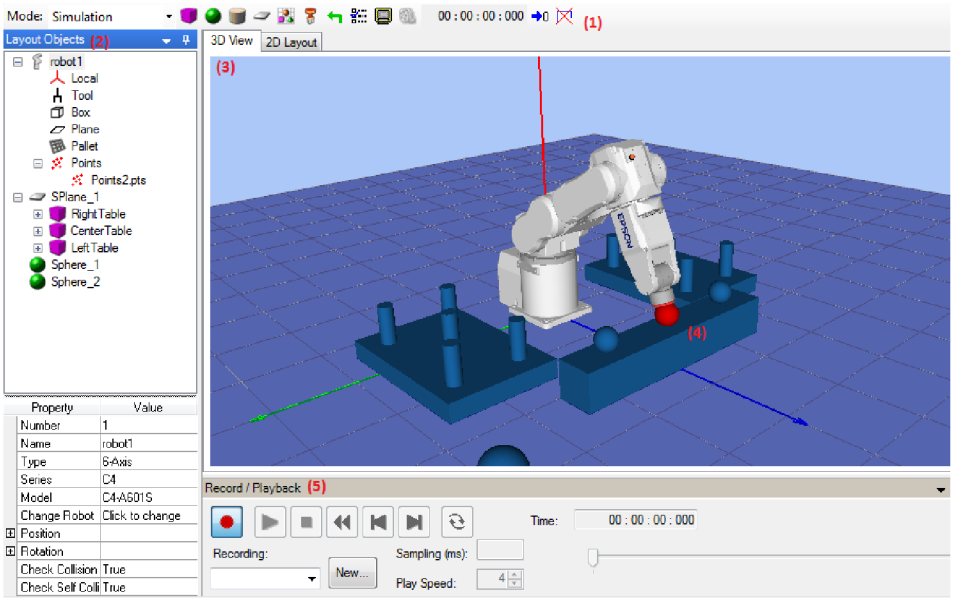


Obrázek 7: Nástroj Controller Tools

1.4.4 Simulátor

V nástroji je umožněno nahrát a vizualizovat model kteréhokoli robotického manipulátoru od společnosti Epson. S vizualizovaným robotem je možné snadno pohybovat pomocí příkazového řádku, nástroji **Robot Manager** (kap. 1.4.1) nebo samotného simulátoru. Simulátor je primárně navržen pro kontrolu pohybu manipulátoru před reálným spuštěním.

Grafické rozhraní je rozvrženo do několika sekcí. V první sekci (1) jsou umístěny hlavní funkce simulátoru jako např. změna způsobu simulace (**Simulation** – manuální simulace nových pohybů, **Playback** – přehrání již nasimulovaného pohybu, ke kterému patří i sekce standardního přehrávače), nákres jednotlivých předmětů, import speciálních chapadel, import výkresu CAD, atd… V sekci rozvržení předmětů (2) je stromový přehled robotů, chapadel a všech předmětů obsažených v simulaci. Hlavní zobrazovací sekce (3) potom umožňuje jak 2D tak i 3D zobrazení nadefinovaných předmětů, chapadel, robotů na definovaných místech. Jedna z důležitých funkcí simulátoru spočívá ve vykreslení červené barvy v místě kolize robotu s pracovním nebo jakýmkoli jiným předmětem (4). K nahrávání/přehrávání těchto simulací slouží poslední sekce tohoto nástroje, která pracuje stejně jako standardní nahrávač/přehrávač (5).



Obrázek 8: Simulátor

Programovací jazyk SPEL+ je jediný kompatibilní jazyk s řídícími jednotkami od společnosti Epson. Je postavený na bázi jazyku BASIC a podporuje důležité funkce jako např. multitasking, řízení pohybu robota a práci s I/O zařízeními.[4]

Programy jsou psány v ASCII znakové sadě a následně jsou kompilovány do spustitelných objektových souborů. Několik z jazykových instrukcí je možné spustit přímo z příkazové řádky vývojového prostředí, která nahrazuje funkce nástroje **„Robot Manager“**. [4]

1.4.5 Struktura programu

Každý projekt musí povinně obsahovat alespoň jeden program a v něm definovanou funkci **„main“**, jestliže tomu tak není, program není spustitelný. Jednotlivé programy projektu jsou vytvářeny v souboru **„prg“** a automaticky se ukládají do složky projektu.

Celkově je možné vytvořit až 63 **„main“** funkcí v jediném projektu. Každý z programů má vlastní startovací funkci: **main1, main2…, main63** a mohou být odděleně spuštěny. Každá z funkcí včetně funkcí **„main“** musí začínat počáteční deklarací a končit **„Fend“** příkazem. Jednotlivé funkce je možné mezi sebou volat příkazem **„Call“** a tím docílit dynamičnosti kódu. [4]

1.4.6 Vybrané příkazy jazyku SPEL+

Motor *On/Off* – Příkaz **„Motor On“** aktivuje napájení motorů a odbrzdí brzdy pro všechny osy. Naopak příkaz **„Motor Off“** napájení deaktivuje a zapne brzdy. Před jakýmkoli pohybem s manipulátorem je potřeba použít příkaz **„Motor On“.**

Power *Low/High* – Přepíná mezi napájecími módy **„Low“** (rychlost do 250mm/s) a **„High“** (plná rychlost manipulátoru). Výchozí napájecí mód při startu je nastaven na hodnotu **„Low“.**

Speed – Nastavuje nebo zobrazuje rychlost manipulátoru (v procentech 1-100). Ovlivňuje pohybové příkazy **„Go“, „Jump“ a „Pulse“**

SpeedS – Nastavuje nebo zobrazuje rychlost manipulátoru (v jednotkách 0,1 – 2000mm/s). Ovlivňuje přímočaré pohyby příkazů **Move, Arc, Arc3, Jump3 a Jump3CP**.

Accel *accel, decel* – Nastavuje rychlost zrychlení a zpomalení (v procentech 1-100) pro PTP pohyby (příkazy **Go, Jump a Pulse**).

AccelS *accel [decel]* – Nastavuje rychlost zrychlení a zpomalení (v jednotkách 0,1 – 25000mm/s) pro přímočaré pohyby (příkazy **Move, Arc, Arc3, Jump3, CVMove** atd.).

Print *expression* – Tiskne zprávu a výstupní data do aktuálního pracovního okna (např. **Run Window, Operator Window, Command Window, Macro Window**).

JTran *jointNumber, distance* – Provede pohyb na specifickém kloubu o specifickou hodnotu ve stupních.

Go *destination* – Přemístí manipulátor PTP pohybem z aktuální polohy do polohy specifikované v příkazu (zadají se souřadnice **X, Y, Z, U, V, W**). Tento příkaz dokáže současně hýbat s klouby 1–6 v jakékoli kombinaci.

Move *destination* – Přemístí manipulátor přímočarým pohybem (pomocí lineární interpolace) z aktuální polohy do polohy specifikované v příkazu (zadají se souřadnice **X, Y, Z, U, V, W**).

HomeSet *J1Pulses, J2Pulses, J3Pulses, J4Pulses, J5Pulses, J6Pulses* – Nastaví a zobrazí domovskou pozici v inkrementech.

Home – Přemístí manipulátor do přednastavené domovské pozice.

On *bitNumber/outputLabel, [time]* – Zapne specifikovaný výstup a po uplynutí specifikované doby, výstup vypne.

Off *bitNumber/outputLabel, [time]* – Vypne specifikovaný výstup a po uplynutí specifikované doby, výstup zapne.

Sw *bitNumber/outputLabel* – Přečte hodnotu na specifikovaném výstupu.

GoTo *{label}* – Bezpodmínečně přemístí vykonávání programu do specifického označení.

OpenNet *{portNumber}* **As** *{Client/Server}* – Otevře specifikovaný TCP/IP síťový port.

ChkNet *(portNumber)* – Vrátí kladnou celou hodnotu, jestliže navázal spojení. V opačném případě zápornou.

WaitNet *#portNumber* – Čeká na vytvoření spojení TCP/IP portu.

Call *funcName* – Zavolá uživatelskou funkci.

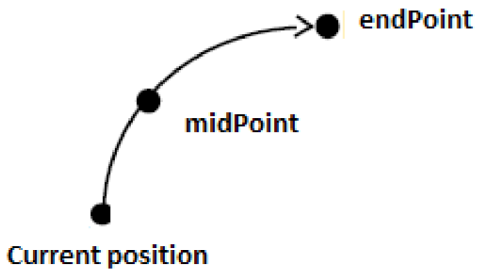
Tool *#toolNumber* – Vybere nebo zobrazí aktuální nástroj.

TLSet *toolNum, toolDefPoint* – Definuje nebo zobrazí souřadnicový systém nástroj. **Tool 0** je výchozí a nedá se měnit. Je možné nadefinovat až 16 různých nástrojů (offsetů).

Input *varName* – Přijímá vstupní data a ukládá je do dané proměnné.

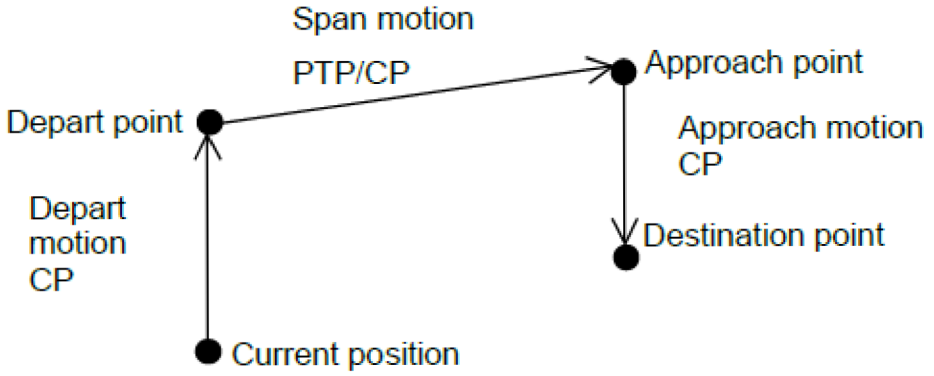
Error *errorNumber* - Generuje uživatelskou chybu.

Arc/Arc3 *midPoint, endPoint* – Přesune manipulátor do specifikovaného bodu pomocí kruhové interpolace. V případě příkazu **Arc** v rovině **XY** a **Arc3** ve 3D.



Obrázek 9: Trajektorie pro příkaz **Arc**, kruhová interpolace

Jump3/Jump3CP *depart, approach, destination* – 3D jeřábový pohyb, který se specifikuje třemi body. Manipulátor těmito body prochází v přímočarém. Příkaz **Jump3CP** při pohybu **„Span motion“** využívá pohybu PTP.



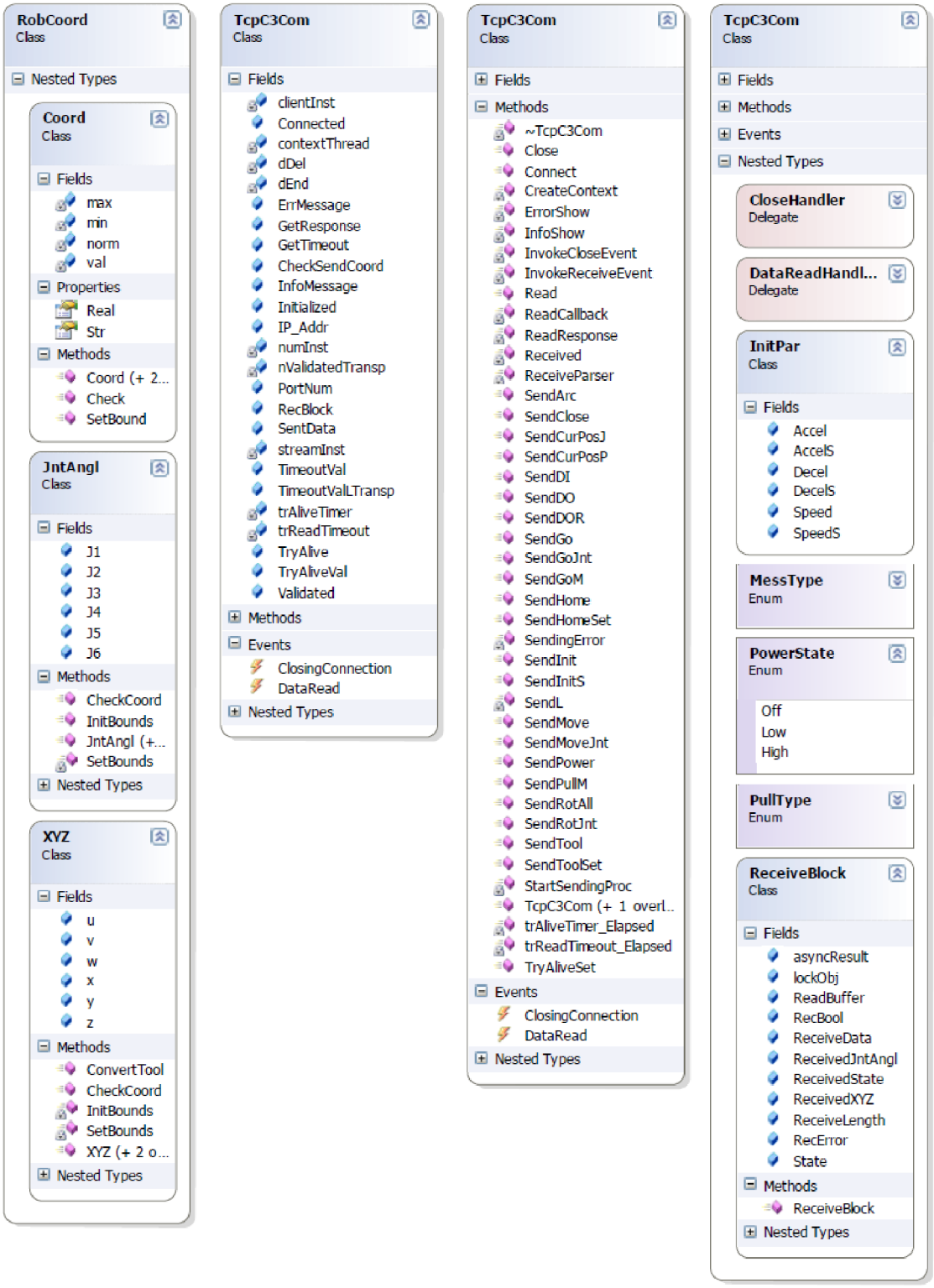
Obrázek 10: Trajektorie pro příkaz **Jump3/Jump3CP**, lineární interpolace

2. C# KNIHOVNA PRO VNĚJŠÍ ŘÍZENÍ

Pro komunikaci s jednotkou RC180/RC700-A byla vytvořena knihovna v jazyce C# jako Diplomová práce pana Ing. Martina Fireše. Knihovna komunikuje pomocí standardní třídy **TcpClient** ze jmenného prostoru **System.Net.Sockets** založená na soketovém přístupu. Na jednotce se pomocí ovládacího panelu spouští zaváděcí program. V okamžiku, kdy uživatel stiskne tlačítko Prog1, se spustí komunikační funkce. Uvnitř funkce se spustí v jednotce TCP server, který pak naslouchá a čeká na připojení TCP klienta přes rozhraní Ethernet.[7]

2.1 Komunikace

Funkce knihovny se odvíjí právě od vytvořených funkcí v jazyce SPEL+, se kterým knihovna úzce spolupracuje. Všechny zmíněné funkce a příkazy jazyku SPEL+ (kap. 3.5.2) odpovídají svým metodám v C# knihovně, které se starají o komunikaci a přípravu dat pro vykonání příkazů jazyka SPEL+. Detailnější popis těchto metod a funkcí je v Diplomové práci Ing. Martina Fireše [8]. V knihovně je vyřešeno mnoho kolizních stavů, které mohou nastat, jako např. nepřipojení TCP serveru do konce přednastaveného časového limitu, nedošlo k parsování, atd. O všech známých případných kolizních stavech je uživatel informován ve formě tisknuté chybové hlášky na obrazovku.

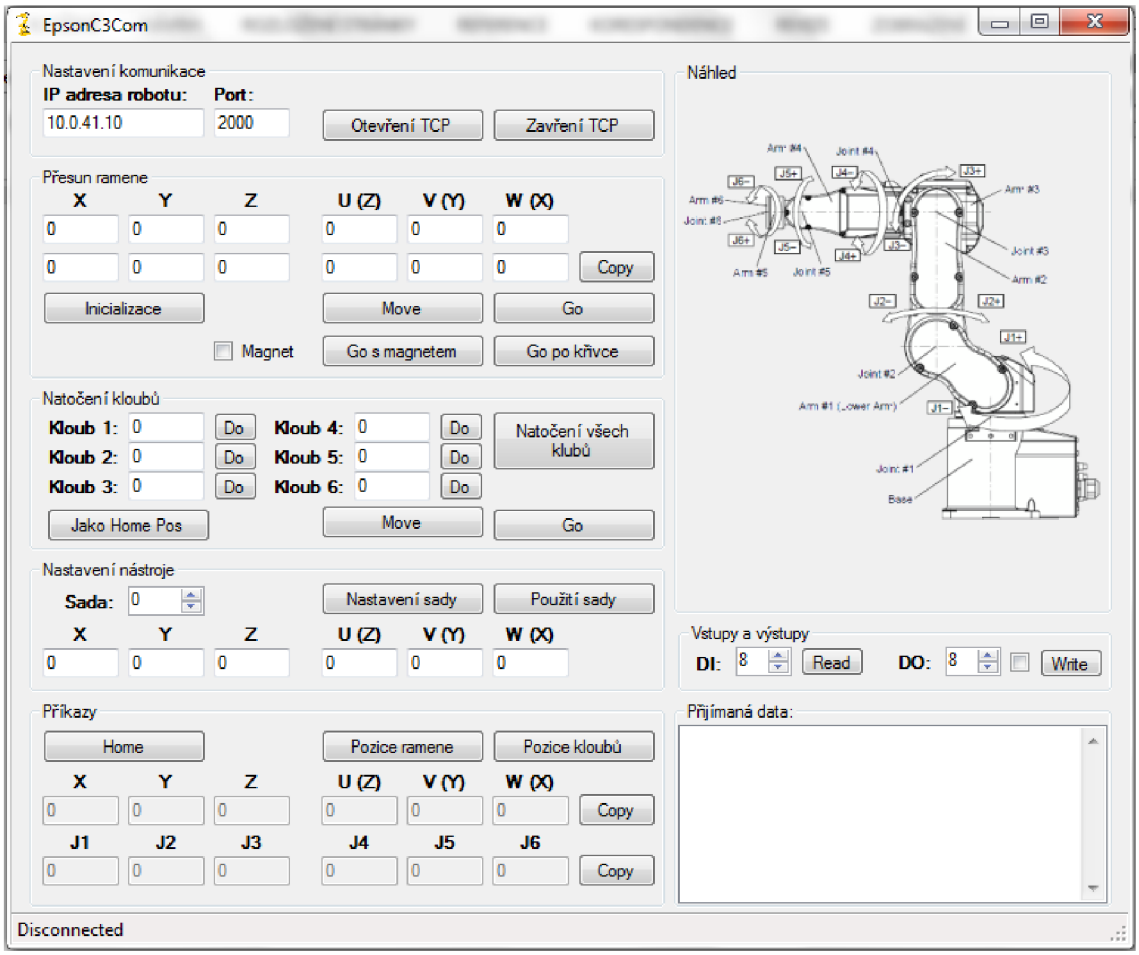


Obrázek 11: Diagram tříd ve jmenném prostoru EpsonC3Com

2.2 Testovací program EpsonC3Com

Pro otestování funkcí knihovny v C# byl vytvořen uživatelský program, jehož hlavní částí je uživatelské grafické rozhraní. Otestovat lze téměř všechny poskytované metody a funkce.

Po zadání správných parametrů komunikace (IP adresa a číslo portu serveru) a klepnutí na tlačítko **Otevření TCP**, by se ve spodním status baru měl změnit stav **Disconected** na **Connected**. Hodnota se získává z flagu **Connected** třídy **TcpC3Com**. Aby bylo možné hýbat s robotem, je nutné provést Inicializaci – pošle **SendInit** bez parametrů, použijí se výchozí hodnoty. Dále je už možné plně s robotem manipulovat. V pravé části uživatelského okna se zobrazují data s časovou hlavičkou, které přicházejí směrem od manipulátoru. Při jakýchkoliv poruchách či chybách se zobrazují varovná hlášení formou vyskakovacích oken.[7]



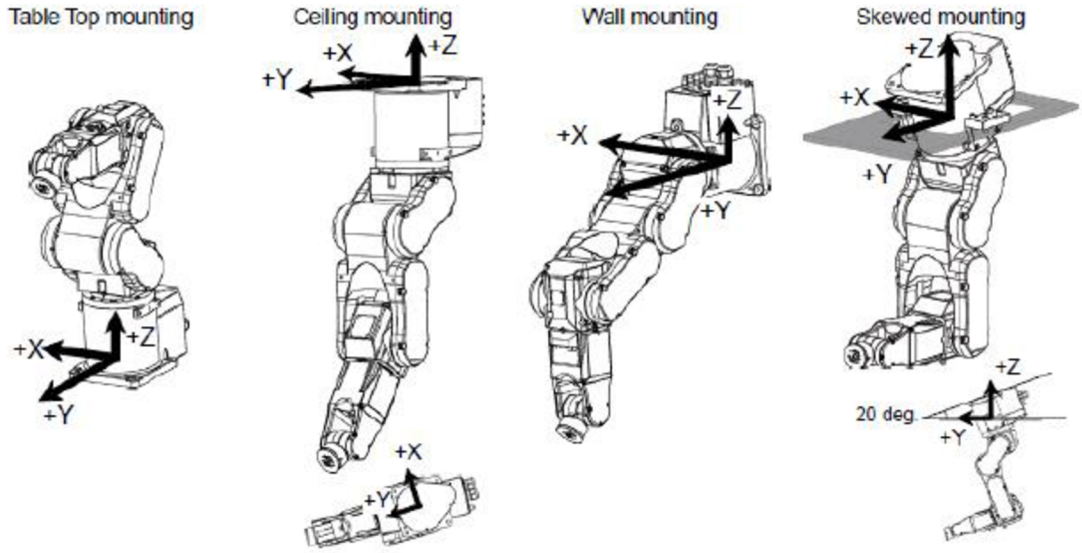
Obrázek 12: Testovací program EpsonC3Com

3. OŽIVENÍ MANIPULÁTORU EPSON C4

Při oživení manipulátoru je třeba dbát všech bezpečnostních pravidel, které jsou spojeny především s osobní bezpečností, ale také bezpečností manipulátoru. S tím také souvisí dosáhnutí tabulkových vlastností robotu a řídící jednotky. K seznámení s těmito pravidly je možné použít dokumentace přiložené k řídící jednotce a robotu nebo použít ucelenou verzi v dokumentaci k instalaci jednotky.

3.1 Umístění manipulátoru

V první fázi instalace je potřeba naplánovat umístění manipulátoru, bezpečnostních zábran, řídící jednotky a popř. ovládacího panelu. Manipulátor je možné umístit stolově, svisle přichycením na zeď nebo zavěšením na strop. Zmíněné umístění je možné kombinovat s různými pracovními úhly, které je potřeba později řádně kalibrovat. Uchycení podstavce je realizováno čtyřmi M8 šrouby, které splňují silovou normu ISO898-1 a spadají do třídy vlastností 12.9.[2]



Obrázek 13: Možné umístění manipulátoru

Řídící jednotku lze umístit vodorovně i svisle. S jednotkou je dodávána příchytná lišta, která usnadňuje instalaci řídící jednotky bez možnosti postavení.[2]

Umístění ochranných zábran musí být nejméně tak daleko od středové osy manipulátoru, aby manipulátor, chapadlo či zdvihaný předmět nebyl v kontaktu s touto

zábranou v plném rozpětí ramene manipulátoru (mimo pracovní prostor manipulátoru). Ovládací panel je potom zpravidla umístěn za touto ochrannou zábranou. [2]

Vzhledem k laboratorním podmínkám, které byly umožněny k oživení robota, tak není možné splnit všechny bezpečnostní pravidla doporučené v dokumentaci. Robot je umístěn ve stolové poloze podle obr. 13. Ovládací panel je umístěn mimo pracovní prostor manipulátoru, ale nikoliv za ochrannými zábranami, které nejsou z prostorových důvodů zhotoveny.

3.2 Bezpečnostní okruh a zapojení

K bezpečnému provozu a obsluze robota je potřeba vytvořit bezpečnostní okruh, který vychází ze zapojení **Emergency** konektoru na zadním panelu řídící jednotky (kap. 1.3).

3.2.1 Zabezpečení

K tomu, aby byl zachován bezpečný pracovní prostor, musí být kolem robota vybudována bezpečnostní zábrana v podobě klece nebo plotu. Tato zábrana musí obsahovat vstupní dveře do pracovního prostoru s bezpečnostními spínači. Tyto bezpečnostní spínače jsou v dokumentaci nazývány jako **„Safety door switches“** a signál z těchto spínačů je zapojen do **Emergency** konektoru. Tento okruh je navržen k použití dvou nezávislých signálů. Jestliže signál z těchto dvou okruhů je rozdílný 2 nebo více sekund, tak se vyhodnocuje jako kritická chyba. Vstupní dveře musí být navrženy, tak aby se nemohly jednoduše zavřít. Rozpojením jakéhokoli z těchto dvou kontaktů způsobíme okamžité vypnutí vykonávaného procesu. V případě startu nám řídící jednotka nedovolí robota zapnout.

Z kapitoly 3.1 je zřejmé, že v našem případě bezpečnostní zábrany nebyly zhotoveny a signály předurčené ke spínačům vstupních dveří byli přemostěny (obr. 15). Tuto skutečnost je možné akceptovat, jestliže se respektuje pracovní prostor manipulátoru a manipulátor bude vždy spuštěn jen v pomalém resp. studijním módu. K manipulátoru je možné přistupovat jen při vypnutých motorech s aktivovanou elektromagnetickou brzdou.

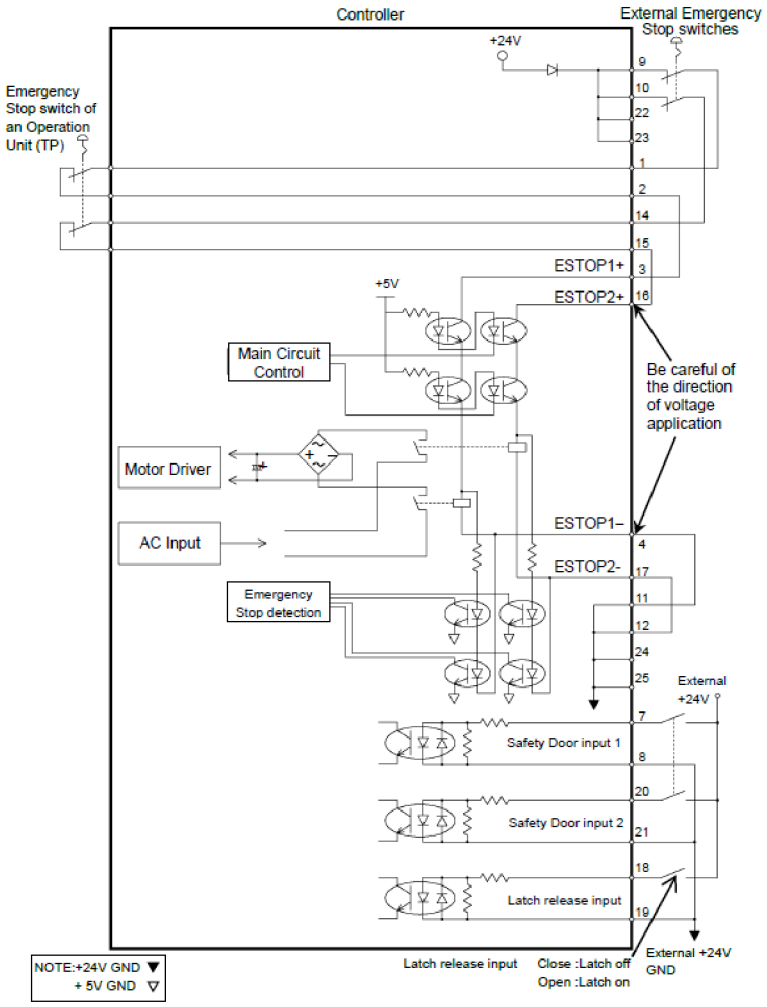
3.2.2 Bezpečnostní tlačítko

Jeden z dalších bezpečnostních prvků je instalace bezpečnostního tlačítka, které splňuje normu IEC60947-5-5. Bezpečnostní tlačítko je umístěno na ovládacím panelu (kap. 3.3) společně s ostatními volně programovatelnými tlačítky (obr. 19).

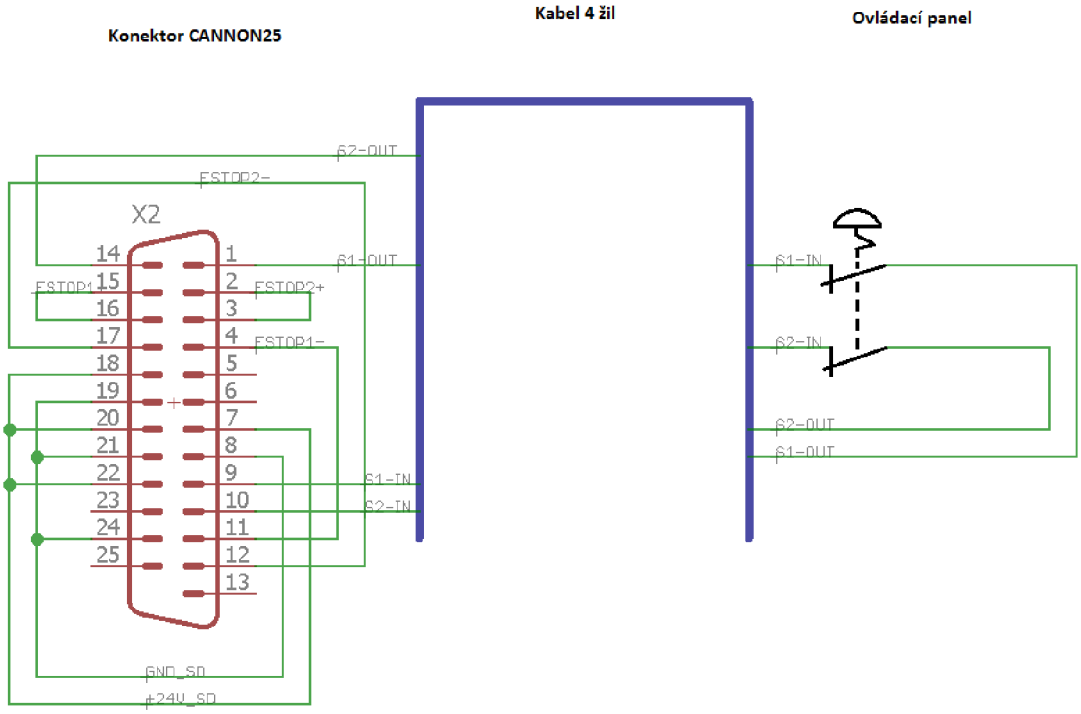
- Musí být s aretací a obsahuje rozepínací kontakt.

- Musí být hřibového tvaru a červené barvy.
- Musí obsahovat dvojitý rozepínací kontakt.

Bezpečnostní tlačítko je stejně jako bezpečnostní spínač navržen k použití dvou nezávislých okruhů. Jestliže signál z těchto dvou okruhů je rozdílný 2 nebo více sekund, tak se vyhodnocuje jako kritická chyba.[2]



Obrázek 14: Vnitřní a vnější zapojení bezpečnostních obvodů

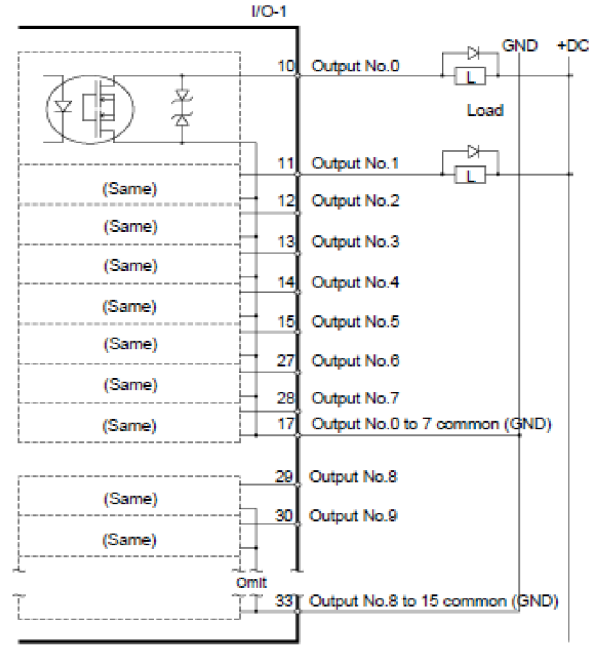


Obrázek 15: Schéma zapojení bezpečnostního konektoru

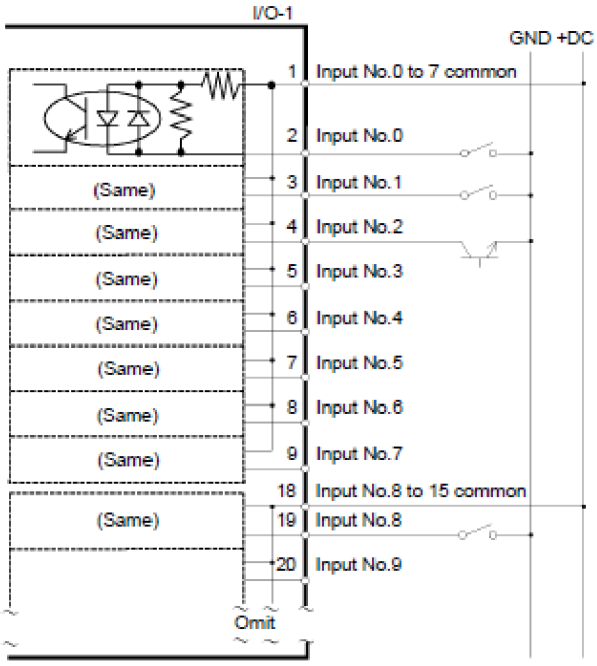
3.3 Ovládací panel

K vnějšímu řízení robota byl navržen ovládací panel se 7 funkčními tlačítky. Tlačítka jsou navržena tak, aby mohla být kdykoli volně programovatelná. Pro rozšířené funkce robota je potřeba využít 5 z těchto tlačítek:

- Start: Spuštění nahraného a zkompilovaného programu.
- Stop: Vypnutí právě probíhajícího programu či úkolu.
- Reset: Resetuje právě probíhající úkoly a stavy.
- Program 1(Prog1): Spuštění programu číslo 1.
- Program 2(Prog2): Spuštění programu číslo 2.



Obrázek 17: Vnitřní a možné vnější zapojení vstupů

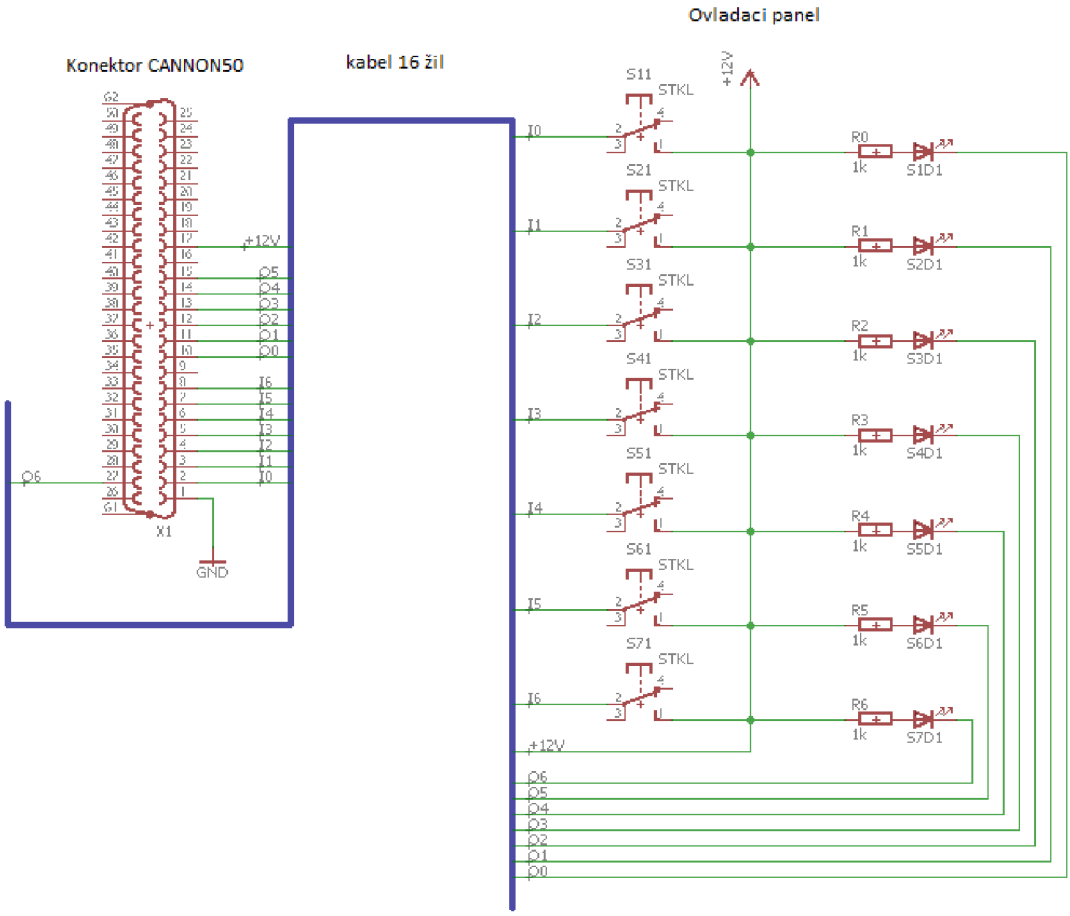


Obrázek 16: Vnitřní a možné vnější zapojení výstupů

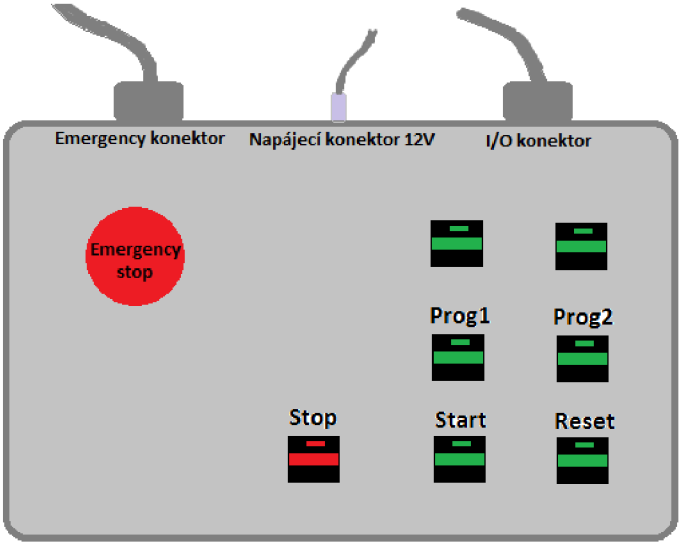
Vstupy a výstupy na ovládacím panelu jsou zapojeny dle schématu obr. 16 a obr. 17. Jsou sdruženy do 16 žilového vodiče, který je na jednom konci zakončen konektorem CANNON50 (I/O konektor řídící jednotky) a druhý konec spojen se samotným ovládacím panelem (obr. 18,19).

Barva	Označení vstupu/výstupu	Číslo pinu	Funkce
zelená	I0	2	TL. Start
zeleno-bílá	O0	10	LED Start
bílá	I1	3	TL. Stop
černá	O1	11	LED Stop
šedo-růžová	I2	4	TL. Reset
fialová	O2	12	LED Reset
hnědá	I3	5	TL. Prog1
šedá	O3	13	LED Prog1
žlutá	I4	6	TL. Prog2
modro-červená	O4	14	LED Prog2
hnědo-žlutá	I5	7	TL. Nastavitelné1
zeleno-hnědá	O5	15	LED Nastavitelné1
bílo-žlutá	I6	8	TL. Nastavitelné2
růžová	O6	27	LED Nastavitelné2
modrá	GND	1	-
červená	+12V	17	-

Tabulka 2: Přehled zapojení vstupů a výstupů



Obrázek 18: Schéma zapojení vstupů a výstupů



Obrázek 19: Rozvržení tlačítek a konektorů na ovládacím panelu

3.4 Instalace a nastavení EPSON RC+ 7.0

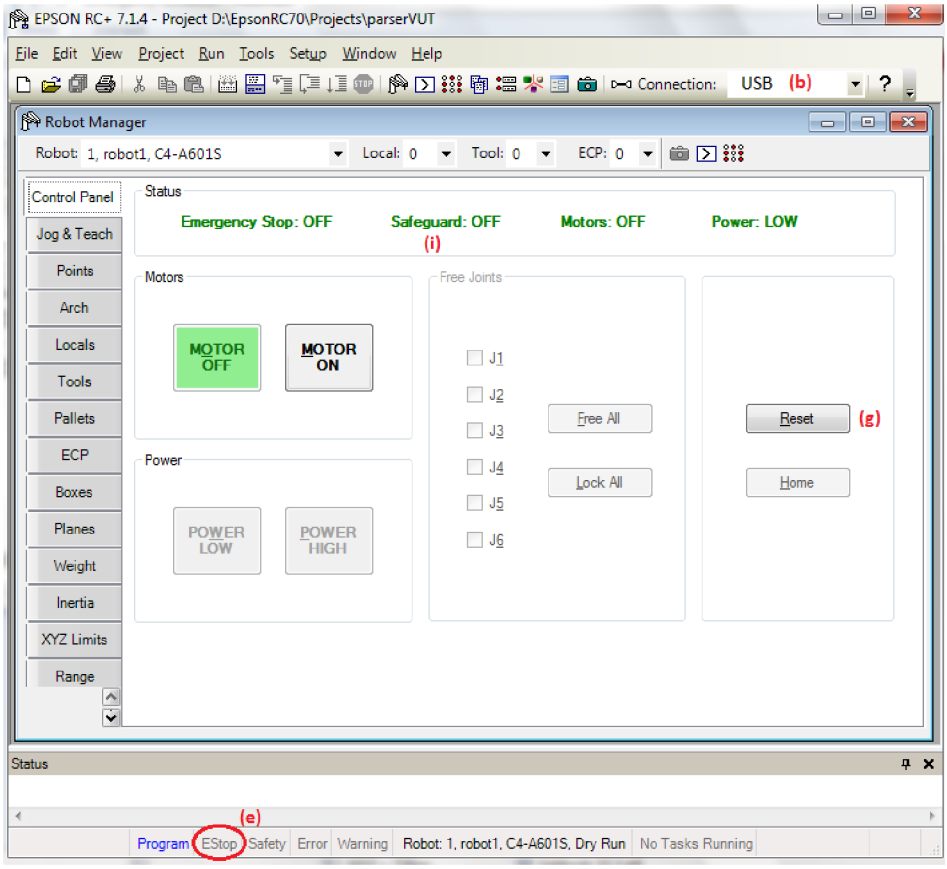
Pomocí instalačního CD od firmy Epson spustíme instalační soubor vývojového prostředí EPSON RC+ 7.0 a software nainstalujeme. Po instalaci vývojové prostředí spustíme a ověříme si funkčnost jednotlivých zapojení.

3.4.1 Kontrola bezpečnostních okruhů

a) Připojení navrženého bezpečnostní konektoru na zadní panel řídící jednotky.
b) Připojení PC k řídící jednotce pomocí USB.
c) Ve vývojovém prostředí vybrat typ připojení jako USB.
d) Zapnout řídící jednotku hlavním vypínačem na zadním panelu a ujistit se, že bezpečnostní tlačítko na ovládacím panelu je v poloze „sepnuto“.
e) Ujistit se, že 7-segmentový display na zadním panelu řídící jednotky zobrazuje:

f) Po nastartování řídící jednotky zkontrolovat, že na dolní stavové liště ve vývojovém prostředí je zobrazeno červeně **„E.Stop“**, což nám indikuje sepnuté bezpečnostní tlačítko.
g) Uvolnit bezpečnostní tlačítko do polohy „rozepnuto“.
h) Ve vývojovém prostředí **[Tools]-[Robot Manager]-[Control Panel]** kliknout na tlačítko **„Reset“**, které resetuje všechny stavy.
i) Ověříme, že 7-segmentový display již nic nezobrazuje a stav **„E.Stop“** v dolní stavové liště vývojového prostředí také není viditelné.
j) Ve vývojovém prostředí **[Tools]-[Robot Manager]-[Control Panel]** ověříme, že **Safeguard** indikuje stav **OFF**.[2]

Tímto je úspěšně ověřeno zapojení bezpečnostního tlačítka a bezpečnostních spínačů ve vstupních dveřích do pracovního prostoru robota.



Obrázek 20: Ověření funkčnosti zapojení

3.4.2 Nastavení I/O zařízení

a) Připojit I/O konektor na zadní panel řídící jednotky

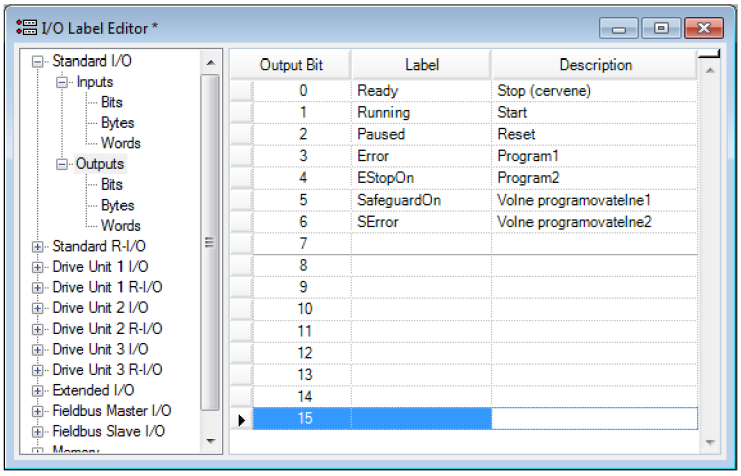
b) Zkontrolovat připojení PC k řídící jednotce pomocí USB.

c) Ve vývojovém prostředí vybrat typ připojení USB.

d) Ve vývojovém prostředí **[Tools]-[I/O Label Editor]-[Inputs]** je možné pojmenování či popisu zapojených vstupních zařízení podle čísla vstupního pinu na konektoru. (obr. 21)

e) Ve vývojovém prostředí **[Tools]-[I/O Label Editor]-[Outputs]** je možné pojmenování či popisu zapojených výstupních zařízení podle čísla vstupního pinu na konektoru. (obr.21)

f) Ve vývojovém prostředí **[Tools]-[I/O Monitor]-[Standard View]** je možné monitorovat nakonfigurované I/O zařízení a u vstupních zařízení, dokonce jejich přímé řízení. (obr. 6)



Obrázek 21: Editace I/O zařízení

K úplnému ověření funkčnosti I/O zařízení byl použit program jazyku SPEL+, ve kterém se LED diody rozsvítily na podnět tlačítek.

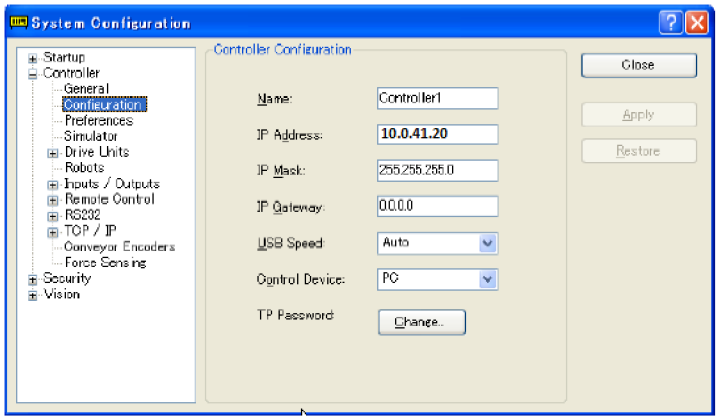
3.5 Ethernet rozhraní

Pro vytvoření komunikace se C# knihovnou je potřeba, aby PC s řídící jednotkou komunikoval přes ethernetové rozhraní. IP adresa řídící jednotky byla zvolena pro testovací účely **10.0.41.20**.

3.6 Konfigurace řídící jednotky

a) Ověřit propojení PC s řídící jednotkou přes USB.

b) Ve vývojovém prostředí **[Setup]-[System Configuration]-[Controller]-[Configuration]** zadat IP adresu **10.0.41.20** a masku sítě **255.255.255.0**.

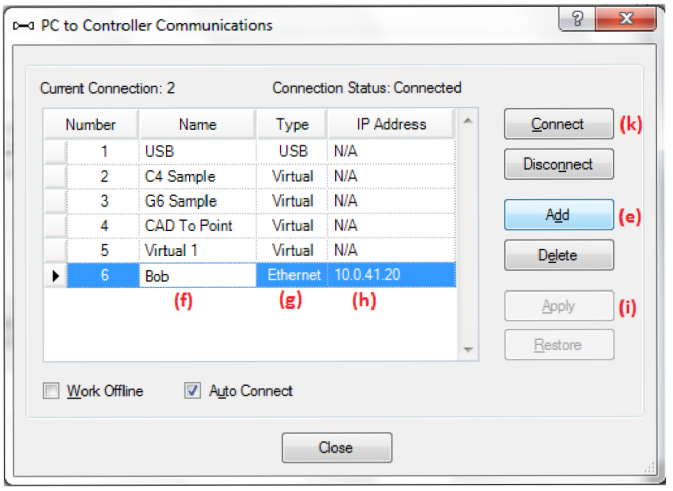


Obrázek 22: Nástroj systémové konfigurace

c) Kliknout na tlačítko **„Apply“.**

d) Zavřít okno nástroje **System Configuration** (řídící jednotka se automaticky restartuje).

e) Po restartu řídící jednotky otevřít nástroj **[Setup]-[PC to Controller Comunications],** kde kliknout na tlačítko **„Add“** a přidat novou možnost připojení řídící jednotky.



Obrázek 23: Připojení řídící jednotky

f) Pojmenovat nové připojení.

g) Zvolit typ připojení „Ethernet“.

h) Zadat IP adresu 10.0.41.20.

i) Potvrdit tlačítkem **„Apply“.**

j) Připojit PC k řídící jednotce pomocí rozhraní Ethernet.

k) Stisknout tlačítko **„Connect“**

l) Je možné odebrání propojení pomocí rozhraní USB.

Po dokončení tohoto návodu je řídící jednotka schopna komunikovat přes ethernetové rozhraní pod IP adresou **10.0.41.20**.

3.7 Vytvoření komunikace TCP/IP na portu 2000

a) Otevřít vývojového prostředí EPSON RC+ 7.0.

b) V **[Setup]-[System Configuration]-[Controller]-[TCP/IP]-[Port 201]** nastavit komunikační port, přes který komunikace poběží:

- IP adresa: 10.0.41.20
- TCP/IP Port: 2000
- Protokol: TCP

- Terminátor: CR

c) Otevřít již vytvořený projekt **parserVUT**
   - V **[Project]-[Open Project]** vyhledat složku projektu a otevřít ji tlačítkem „Open“

d) V **[Project Explorer]-[Program Files]** otevřít zdrojový kód programu Main.prg.

e) Upravit dříve definované názvy I/O zařízení u příkazů, které s nimi pracují např. On/Off, Sw atp.

3.8 Kompilace a sestavení projektu

Pro sestavení, kompilaci a spuštění naimplementovaného programu SPEL+ potažmo celého projektu **parserVUT** je potřeba přejít do **[Run]-[Run Window]**, během spuštění nástroje se projekt sestaví a dolní stavové okno zobrazí výsledek kompilace a sestavení projektu.

Při sestavení projektu **parserVUT**, který byl vytvořen ve starší verzi vývojového prostředí EPSON RC+ 5.0 obvykle nastane chyba souboru **pts**, který pracoval s odlišným rozvržením souboru pro ukládání pozic. Tento soubor v projektu **parserVUT** není kompatibilní s novější verzí a je nutné tento soubor přepsat do nového souboru vytvořeného novější verzí nebo robota naučit pozicím novým.

Po ošetření nebo neprojevení této chyby je možné pokračovat se řádně zkompilovaným a sestaveným projektem.

3.9 Spuštění a ověření komunikace se C# knihovnou

Pro komunikaci jednotky RC180 byla vytvořena knihovna v jazyce C# jako diplomová práce pana Ing. Martina Fireše, která v implementaci kódu nepočítala s použitím jiných řídících jednotek. Avšak řídící jednotka RC700-A je nástupcem jednotky RC180 a knihovna je při použití této jednotky plně funkční. Změny, které byly v jednotce vytvořeny nemají žádný vliv na základní funkce, které jsou v knihovně řešeny, dokonce i komunikace pomocí standardní třídy **TcpClient** ze jmenného prostoru **System.Net.Sockets** založená na soketovém přístupu funguje bez problému.

K ověření plné komunikace s knihovnou je potřeba využít projekt **parserVUT** a **EpsonC3Com**, které jsou součástí přílohy. Tyto programy spolu komunikují pomocí protokolu

TCP/IP server (řídící jednotka robotu), klient (PC). Postup vytvoření komunikace mezi těmito programy je následující:

a) V nastavení síťových karet na PC nastavit IP adresu ve stejné podsíti jako řídící jednotka manipulátoru a standartní masku **255.255.255.0**.
b) V EPSON RC+ 7.0 spustit řádně zkompilovaný projekt **parserVUT** stlačením tlačítka **[Run]-[Run Windows] „Start“.**
c) Na PC spustit komunikační program ***EpsonC3Com*.**
d) Stlačení tlačítka **„Prog1“** na ovládacím panelu pro zahájení programu komunikace se C# knihovnou.
e) Program **parserVUT** vytvoří TCP/IP spojení jako server a čeká na odezvu klienta.
f) V programu **EpsonC3Com** nastavit IP adresu robotu na **10.0.41.20** a komunikační port 2000.
g) Ověřit komunikaci stlačením tlačítka **„Ověření TCP“**. Tím se odešle zpráva TCP serveru o zdařené komunikaci.
h) Program **parserVUT** zprávu přijímá a posílá klientovi ověřovací zprávu. (dostává se do přijímací smyčky a naslouchá dalším zprávám.
i) Před samotným ovládáním z programu **EpsonC3Com** zmáčkneme tlačítko „Inicializace“ k nastavení počátečních hodnot (napájecí mód, rychlost, atd…).

V tomto stavu je robot ovládán prostřednictvím programu EpsonC3Com a je možné využívat naimplementované metody C# knihovny. Pro seznam možných kolizních stavů a detailní popis komunikace slouží Diplomová práce pana Ing. Martina Fireše [7].

4. SW NÁSTROJ TCPC3COM V2.0

Pro zefektivnění práce a využití všech rozšiřujících funkcí knihovny **EpsonC3Com** byl vytvořen softwarový nástroj **TcpC3Com v2.0**. Ten byl vytvořen tak, aby manipulace a případné plánování trajektorie dokázal i běžný uživatel a přitom využil všech funkcí knihovny. Nástroj si zanechal svojí úvodní lištu v podobě připojení a inicializace mezi robotem a knihovnou, která je popsána v kapitole 3.8.

4.1 Návrh formulářové vrstvy

Layout softwarového nástroje je uspořádán do 7 karet v podobě menu. Obsahem je úvodní obrazovka s ovládacím panelem, různé způsoby pohybu a plánování trajektorie robotického ramene a také následné ukládání dat potřebných k plánování trajektorií.

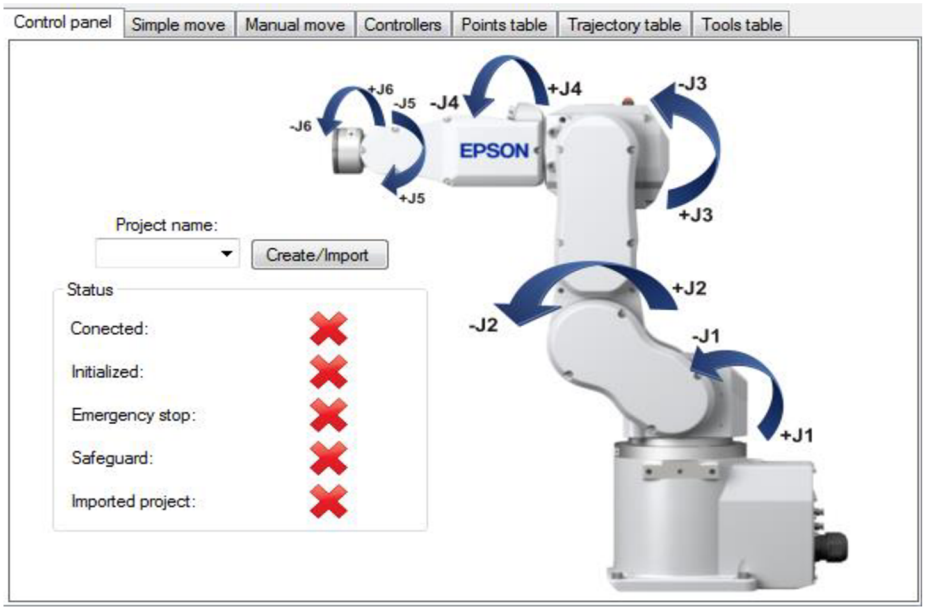
4.1.1 Ovládací panel (Control panel)

Úvodní obrazovka nazvaná **Control panel** na první pohled obsahuje obrázek ovládaného robota s přehledným popisem a směrem pohybu, který je nezbytný pro orientaci směru při manuálním pohybu.

Hlavní součástí této sekce je vytvoření či import projektu, který obsahuje všechny ukládaná data (body, trajektorie, nástroje), se kterými se v nástroji pracuje. Tento tzv. projekt je spravován třídou **RobData** (kap. 4.2), která obsahuje příslušné metody a atributy k práci s jednotlivými daty.

Dále se tu nachází logický přehled základních proměnných, které jsou nezbytné pro bezpečný a správný chod aplikace:

a) **Connected** – Indikace je propojená s navázáním spojením přes protokol TCP/IP (zelená = připojeno, červená = nepřipojeno)
b) **Initialized** – Indikace je propojená s inicializační metodou třídy TcpC3Com (zelená = inicializováno, červená = neinicializováno)
c) **Emergency Stop** – Indikuje stav nouzového tlačítka na vnějším ovládacím panelu (zelená = neaktivní, červená = aktivní)
d) **Safeguad** – Indikuje stav bezpečnostního okruhu (zelená = připraven, červená = nepřipraven)
e) **Imported project** = Indikuje aktivitu importovaného projektu (zelená = aktivní, červená = neaktivní)

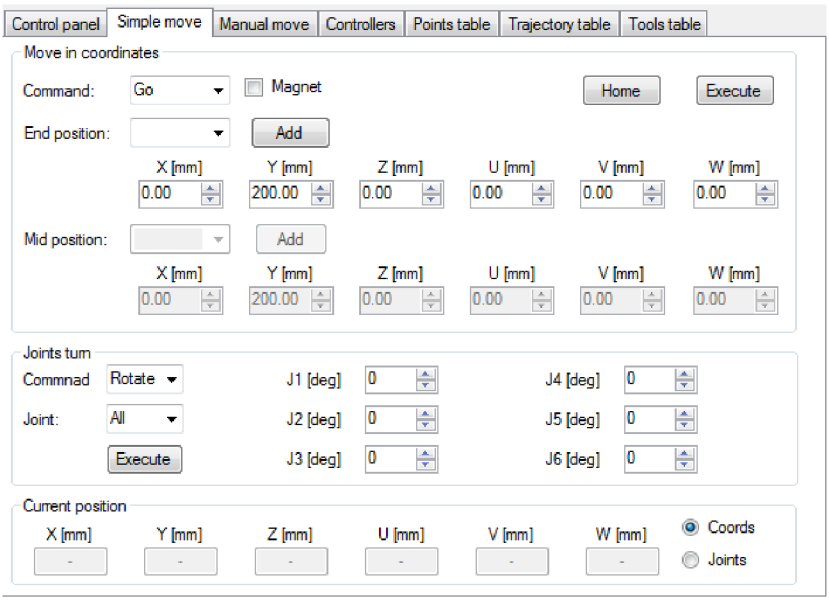


Obrázek 24: EpsonC3Com v2.0 - Control panel

4.1.2 Jednoduchý pohyb (Simple move)

Ve druhé kartě byly ponechány pozůstatky testovacího programu **TcpC3Com**, které simulují základní pohyby robotu. V sekci **„Move in coordinates“** je uživatel schopen souřadnicového přesunu robotického ramene. Použité souřadnice je možné uložit jako bod pomocí tlačítka **„Add“**. Sekce **„Joints turn“** slouží k jednoduchému otočení kloubu či více kloubů ramene najednou.

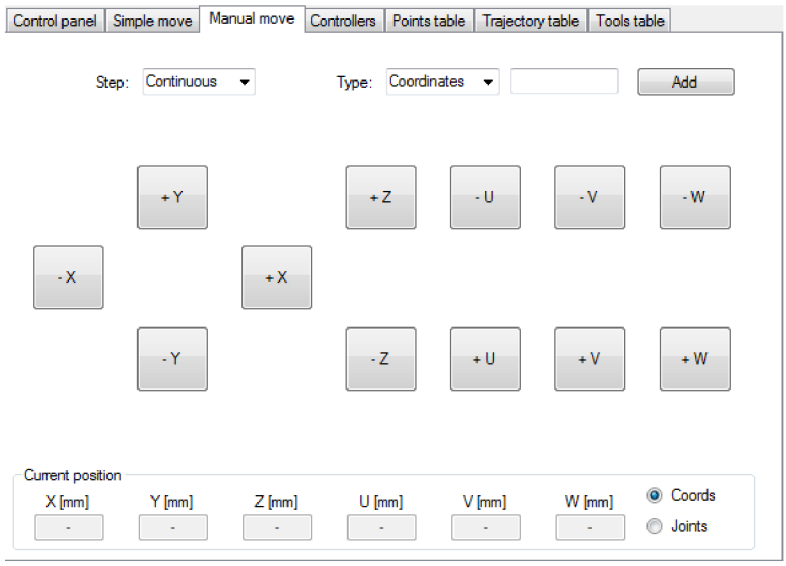
Ve spodní části karty se nachází sekce **„Current position“**, která zobrazuje aktuální pozici ovládaného robotického ramene. Aktuální pozice se aktualizuje po každém pohybu ramene a je možné vybrat si mezi dvěma zobrazovacími formáty (souřadnice, natočení kloubů).



Obrázek 25: EpsonC3Com v2.0 - Jednoduchý pohyb

4.1.3 Manuální pohyb (Manual move)

Karta manuálního pohybu obsahuje řadu tlačítek k pohybu s robotem. Každé tlačítko reprezentuje inkrementaci/dekrementaci souřadnic nebo úhlu natočení kloubů. Inkrementace/dekrementace je možná ve třech velikostech kroku (malý, střední, velký), kde malý reprezentuje hodnotu kroku 0.1mm (0.1˚), střední 1mm (1˚) a velký 10mm (10˚). Z nabídky je možnost výběru kontinuálního pohybu (continuous) s robotem, kdy při stlačeném příslušném tlačítku se robot pohybuje v daném směru, dokud není tlačítko uvolněno. Ve spodní části je opět zobrazovací sekce pro aktuální pozici ramene, která má stejné vlastnosti jako v kapitole 4.1.2.

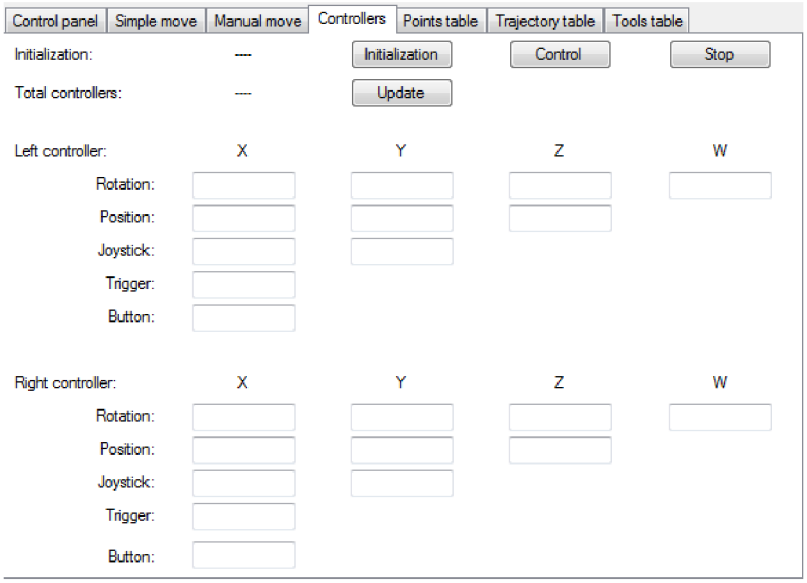


Obrázek 26: EpsonC3Com v2.0 - Manuální pohyb

4.1.4 Ovladače (Controllers)

Další možností pohybu a plánování trajektorie je pomocí ovladačů **Razor Hydra**, které jsou připojeny k počítači přes USB port. Tenhle způsob pohybu je detailně rozebrán v kapitole 5. Pro zahájení řízení je nejdříve potřeba inicializovat komunikační knihovnu tlačítkem **Initialization** (při správné inicializaci vrátí hodnotu 0) a následně ověřit připojení obou ovladačů tlačítkem **Update** (vrátí počet připojených ovladačů). Řídící vlákno se spouští tlačítkem **Control** a ukončuje tlačítkem **Stop.**

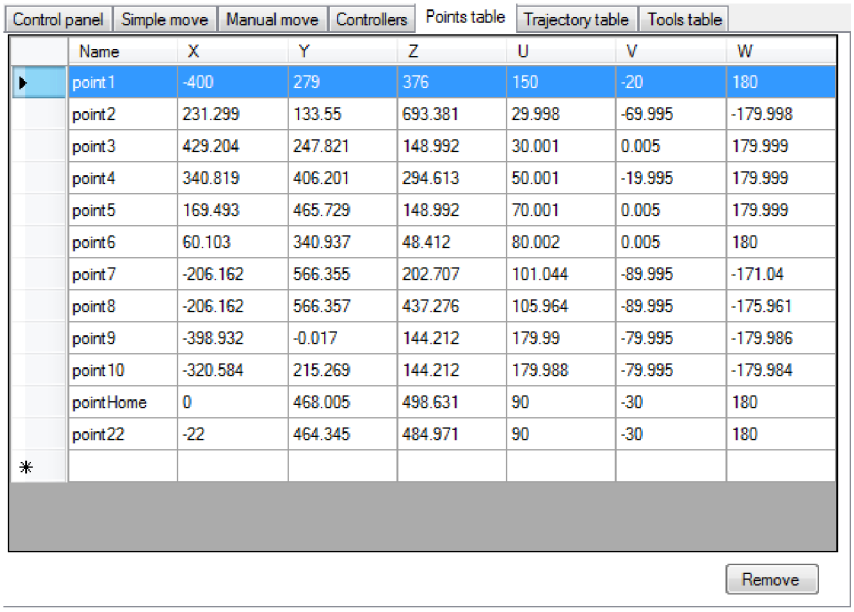
Pro inicializaci počátku souřadnic ovladačů je potřeba zmačknout tlačítko **Start** na daném ovladači, jestliže se začnou v textovém poli pozice reálně načítat data, tak je poloha počátku úspěšně inicializovaná. Pak už jen stačí podržet tlačítko **Bumper** vždy, když je potřeba odesílat robotu pohybové příkazy.



Obrázek 27: EpsonC3Com v2.0 - Ovladače

4.1.5 Tabulka bodů (Points table)

Tabulka bodů zobrazuje informace o uložených bodech projektu. Informace obsahuje jméno a souřadnice daného bodu. Body se přidávají v kartách jednoduchého a manuálního pohybu, kdy se přímo s robotem a potencionálními souřadnicemi pracuje. Ve spodní části karty se nachází tlačítko k odstranění označeného bodu z tabulky potažmo z celé paměti.



Obrázek 28: EpsonC3Com v2.0 - Tabulka bodů

4.1.6 Tabulka trajektorie (Trajectory table)

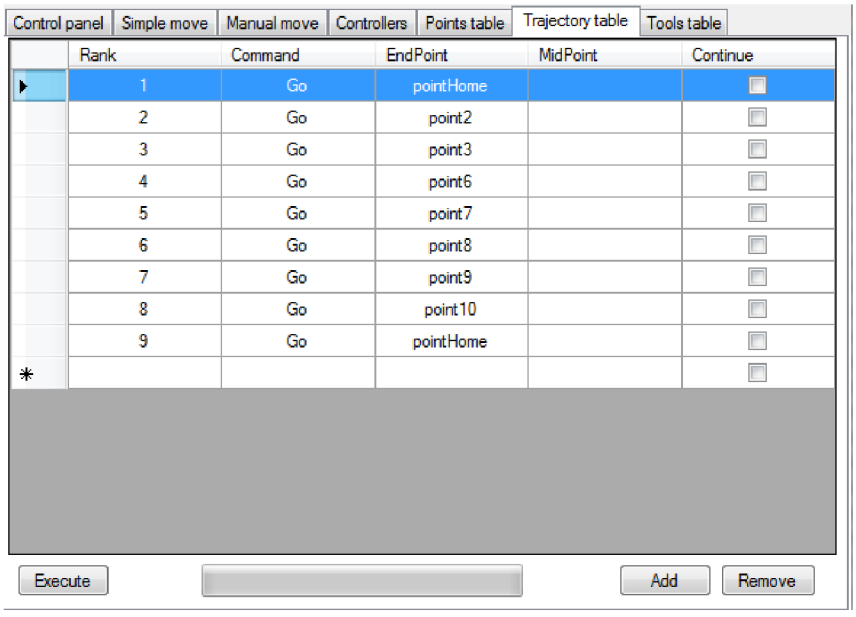
Tato tabulka je vytvořena v podobné stylu jako tabulka bodů, kde v horní části karty se nachází patřičné náležitosti jednotlivých bodů plánované trajektorie a ve spodní části funkční tlačítka.

Z levé strany se nachází tlačítko **Execute** pro vykonání trajektorie. Při následném stlačení se začínají sekvenčně vykonávat body trajektorie (po tuto dobu je možné sledovat její časový průběh pomocí ukazatele ve střední části, ale není možné s aplikací nijak pracovat) od prvního řádku tabulky po její konec.

Dalším funkčním tlačítkem je **Add** pro přidání bodu trajektorie do tabulky, které využívá rozšiřujícího formuláře pro zadání jednotlivých složek bodu (příkaz, koncový bod, průchozí bod, pokračovat).

a) **Příkaz (Command)** - zadává pohyb na základě příkazu
b) **Koncový bod (End point)** – konečný bod jednotlivého pohybu (nikoliv celé trajektorie)
c) **Průchozí bod (Middle point)** – průchozí bod, který je zapotřebí při použití např. příkazu **Arc**.
d) **Pokračovat (Continue)** – logická proměnná, která specifikuje setrvání nebo jen projetí konečného bodu.

Při stlačení posledního tlačítka **Remove** se odstraní označený bod trajektorie z tabulky resp. z celé paměti.



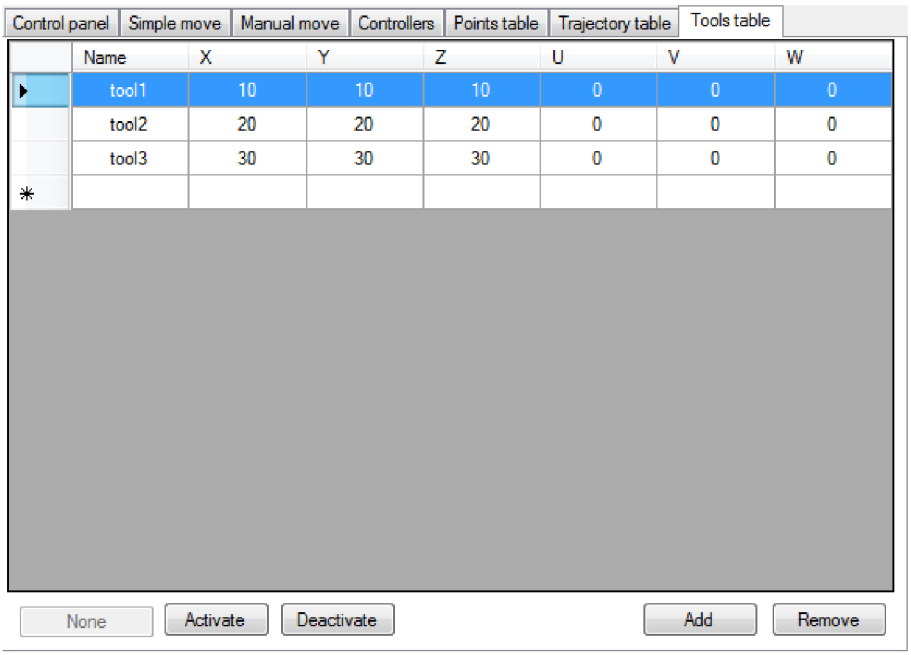
Obrázek 29: EpsonC3Com v2.0 - Tabulka naplánované trajektorie

4.1.7 Tabulka nástrojů (Tools table)

Jedná se o tabulku offsetových hodnot počátku souřadnic manipulátoru. Každý nástroj obsahuje informaci o přiřazeném jménu a jednotlivých offsetových kartézských souřadnicích. Ve spodní části jsou opět umístěny funkční tlačítka. Tlačítko **Activate**, aktivuje označený nástroj v tabulce a k počátku souřadnic přičte offsetové hodnoty. Při aktivním nástroji je jeho jméno pro přehlednost indikováno v textovém poli pod tabulkou.

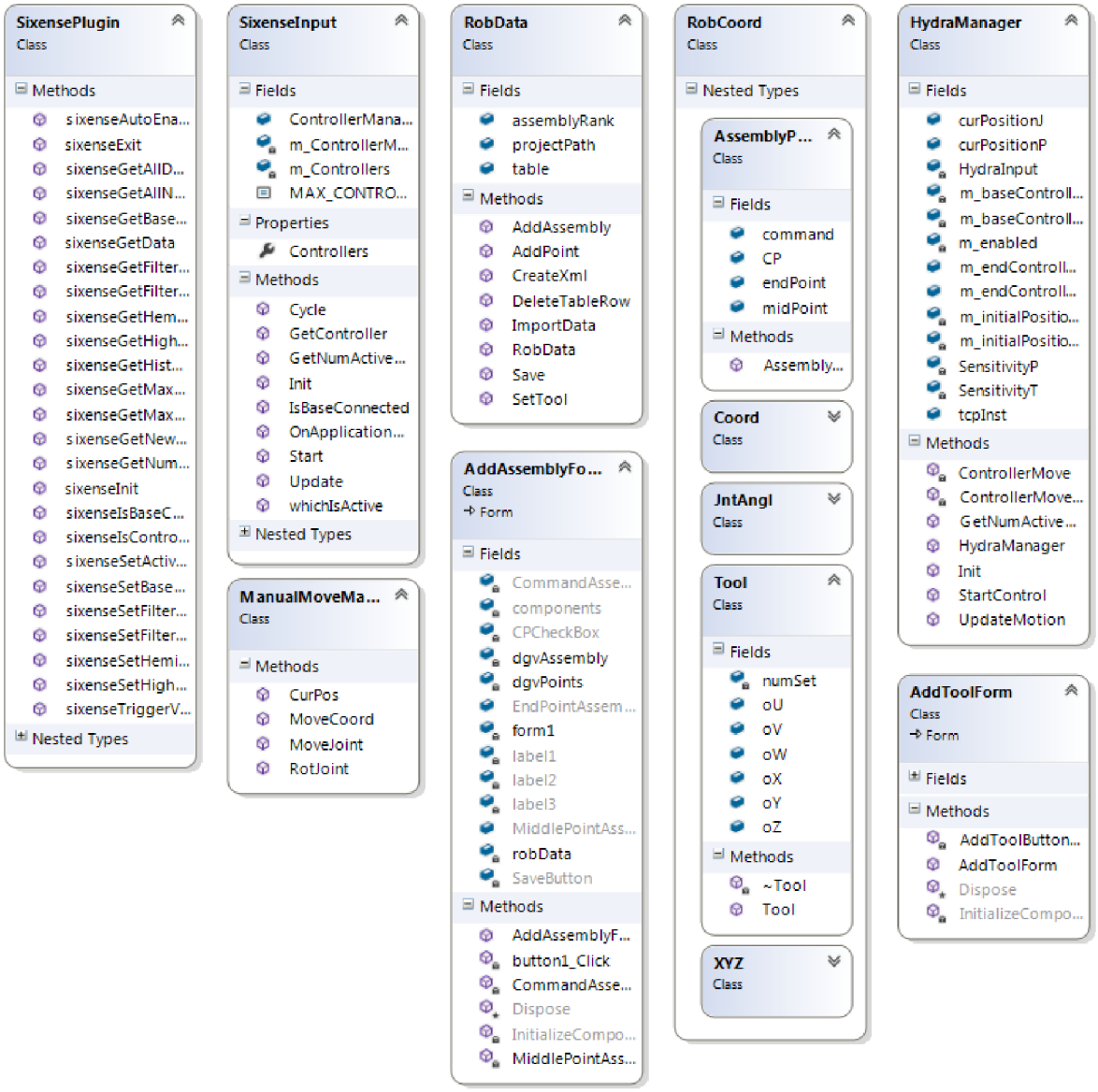
V případě reaktivace stačí postupovat stejně jako při aktivaci nástroje, ale při deaktivaci je potřeba stlačit tlačítko **Deactivate** a offsetové hodnoty se nastaví na nulu.

Offsetové hodnoty je možné vytvářet a mazat pomocí tlačítek přidat (**Add**) a odebrat (**Remove**) avšak jejich maximální počet vychází z nastavení samotného manipulátoru a je omezen na 15 položek.



Obrázek 30: EpsonC3Com v2.0 - Tabulka nástrojů

4.2 Návrh logické vrstvy



Obrázek 31: Diagram vytvořených a převzatých tříd

4.2.1 Třída RobData

Je to třída, která spravuje a synchronizuje uložená data mezi XML soubory a tabulkami ve formulářové vrstvě. Při vytvoření či importování tzv. projektu se vytvoří instance třídy **RobData** pro jednotlivé části (body, trajektorii a nástroje).

Při volání konstruktoru třídy se vytvoří tabulka v podobě instance třídy **DataTable** závislá na argumentu, který specifikuje, o jaký typ tabulky se jedná. Dále si vytvoří cestu k projektu v podobě proměnné **string**.

4.2.1.1 Metoda CreateXml()

Metoda bez vstupních parametrů, která vytvoří XML soubor v adresáři projektu předaným konstruktorem třídy. Je volána zpravidla při vytvoření nového projektu.

4.2.1.2 Metoda ImportData()

Vstupní parametry metody obsahují formulářovou tabulku **DataGridView**, jméno projektu a již vytvořené projektové adresáře. Metoda se stará o importaci všech částí projektu z XML souborů do formulářové tabulky.

4.2.1.3 Metoda AddPoint()

Metoda pro vytvoření nového bodu nebo nástrojového offsetu a následná synchronizace mezi XML a formulářovou tabulkou **DataGridView**. Vstupní parametry obsahují instanci třídy **RobCoord.XYZ** (souřadnice bodu) a formulářovou tabulku.

4.2.1.4 Metoda AddAssembly()

Obdoba metody **AddPoint()**, ale pracuje se s instancí třídy **RobCoord.AssemblyPoint**, která zapouzdřuje informace o jednotlivých bodech trajektorie.

4.2.1.5 Metoda SetTool()

Metoda nastavující offsetové hodnoty počátku souřadnic robota. Synchronizuje data XML, tabulky a odesílá nastavení řídící jednotce robota pomocí komunikační třídy **TcpC3Com**.

4.2.1.6 Metoda Save()

Metoda ukládající vytvořená data do příslušného XML souboru.

4.2.1.7 Metoda DeleteTableRow()

Metoda, která vymaže aktivní řádek tabulky a následně synchronizuje data pro ostatní vrstvy paměti.

4.2.2 Třída ManualMoveManager

Vytvořená třída spravuje manuální pohyb robotického ramene. O tyto pohyby se starají 3 níže zmíněné metody, které rozlišují krokové/kontinuální pohyby zadané buď v kartézských souřadnicích, nebo v natočení jednotlivých kloubů.

Pro tyto typy pohybů byl také modifikován program **parserVUT** v jazyce SPEL+. Modifikace spočívala ve vyhlazení kontinuálních pohybů robota a tudíž přidání případných parametrů k volajícím funkcím. V tuhle chvíli je již možné předávat parametry typu **CP** nebo **ROT** pro příkazy **Go** nebo **Move** a to jak v kartézských souřadnicích, tak v natočeních kloubů. I přesto, že byli tyto parametry do kódu naimplementovány, tak robot stále nevykonává 100% plynulý pohyb. Je to dáno přesností dojetí do bodu, kterým má trajektorie procházet.

4.2.2.1 Metoda MoveCoord()

Vstupní parametry metody obsahují instanci třídy **TcpC3Com** a třídy **RobCoord.XYZ**, informaci o směru a velikosti kroku pohybu ramene. Metoda inkrementuje krok v daném směru do následně odesílaných souřadnic pohybu. Nakonec vrací hodnoty nových souřadnic.

4.2.2.2 Metoda RotJoint()

Vstupní parametry metody obsahují instanci třídy **TcpC3Com**, číslo rotujícího kloubu, směr a krok rotace ramene. Na základě těchto parametrů se pomocí komunikační třídy **TcpC3Com** odesílají data o rotaci kloubů robota.

4.2.2.3 Metoda MoveJoint()

Obdoba metody **MoveCoord().** Inkrementuje hodnoty kroku pro kontinuální pohyb při rotaci kloubů. Pro tuto metodu byl upraven program **parserVUT** a připsán příkaz **„Go AglToPls [CP]“** pro kontinuální rotaci kloubu.

5. PLÁNOVÁNÍ TRAJEKTORIE

Z mnoha možností různých zařízení pro názorné plánování trajektorie (skriptování, 3D myš, kinect atp.) byly vybrány ovladače **Razor Hydra**, které výborně splňují všechny předpoklady k rychlému, jednoduchému a hlavně bezpečnému plánování trajektorie robotického ramene Epson C4.

5.1 Možnosti plánování trajektorie

Při výběru plánovacího zařízení byl kladen důraz především na široký rozsah citlivostí, což je hlavní faktor použitelnosti ovládání. To hlavně nesplňovala možnost aplikování 3D myši, která je velmi citlivá na změnu pozičních hodnot (jakýkoliv třes rukou byl robotem výrazně zaznamenán).

5.1.1 3D myš

3D myš je vstupní zařízení, které je modifikací standardních ovládacích myší k počítači. Na rozdíl od standardní myši je 3D myš navržena pro usnadnění práce v grafických editorech na bázi CAD. Tzv. hlavička myši je schopna pracovat v šesti stupních volnosti a mnohdy je její součásti víc než uspokojivý počet volně programovatelných tlačítek.

Tato možnost použití výborně splnila většinu předpokladů, které jsou potřeba k jednoduchému a názornému plánování trajektorie, avšak hlavní nevýhodou je její citlivost a rozsah posuvu (otočení) hlavičky myši. K práci v takhle malém pracovním rozsahu je potřeba tzv. modelářských rukou uživatele k dokonalému pohybu.

5.1.2 Pohybový senzor Kinect

Dalším nápadem k plánování trajektorie pro tuto práci bylo použití pohybového senzoru Kinect, který by byl použit ke snímání buď přímo lidských rukou nebo nějakého specifického snímaného nástroje či ovladače.

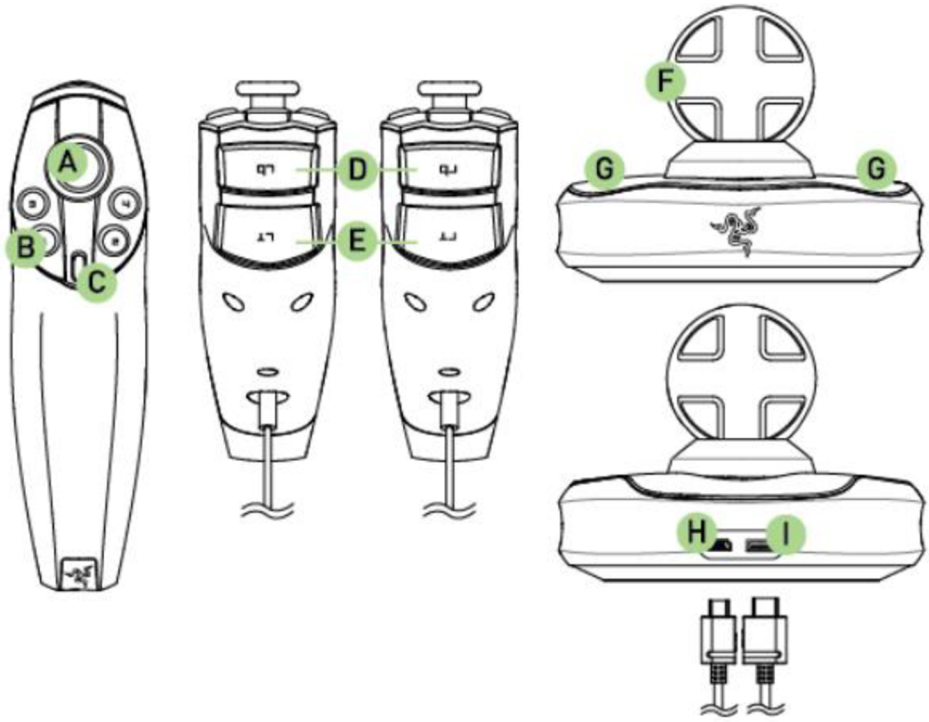
Od této metody bylo upuštěno z důvodu složitosti realizace nad rámec této práce, kdy by bylo potřeba pracovat s počítačovým viděním a detailním rozebráním principu pohybového senzoru. Dalším ovlivňujícím faktorem byla dostupnost senzoru, kde značně převyšovaly ovladače **Razor Hydra**, které byly dostupné ihned v laboratoři.

5.2 Razor Hydra ovladače

Tyto ovladače byli navrženy především pro hráče počítačových her, avšak jejich zdařilá konstrukce a softwarová podpora předčila jejich počáteční očekávání a aplikace se rozšířila do nejen hráčského odvětví. Ovladače **Razor Hydra** byli vybrány především díky jejich dostupnosti a lehce nastavitelnou citlivostí a rozsahu pohybu, kde nejmenší přesnost citlivosti se udává od 1mm (popř. 1˚).

5.2.1 Hardwarový popis

Princip zařízení spočívá v magnetickém pohybovém senzoru (F) v podobě dokovací stanice, která snímá polohu připojených jednotlivých ovladačů. Na dokovací stanici najdeme vyhloubená místa (G) k odložení ovladačů a samozřejmostí jsou porty pro komunikaci s PC (H) a ovladači (I).



Obrázek 32: Ovladače Razor Hydra

Jednotlivé ovladače, jak už je zmíněno výše obsahují řadu volně programovatelných tlačítek. Najdeme zde klasická tlačítka označena číslicemi 1-4 (B) a tlačítko **„Start“** (C). Ve střední části ovladače je umístěn joystick (A), který snímá polohu ve dvou osách **X, Y**. V přední části ovladače se nachází velké pohotovostní, jednoduše stlačitelné tlačítko tzv. **„Bumper“** (D)

a tlačítko **„Trigger“** (E) detekující sílu stlačení, které v našem případě má velký potenciál např. v ovládání pneumatických nástrojů, které mohou být nainstalovány na rameni.

5.2.2 Softwarová podpora

Společnost **Sixense Entertainment** vytvořila mnoho zdařilých softwarových nástrojů na podporu vývoje těchto ovladačů. Nejaktuálnější verzí je SW nástroj MotionCreator, který slouží především pro hráčské nadšence s širokou možností naprogramování ovladačů v jednoduchém uživatelském prostředí. Nad možností použití tohoto SW nástroje bylo uvažováno, ale nakonec odstoupeno z důvodu nepraktičnosti v případě dalších modifikací pohybu s robotem.

V této práci byla komunikace vytvořena pomocí SDK (**Software development kit**) pro **Unity engine** od stejnojmenné společnosti. Komunikační knihovna **Sixense.dll** a třídy **SixensePlugin, SixenseInput** byly převzaty z tohoto vývojářského balíčku a k nim naimplementována třída **HydraManager** k řízení robota na základě načtených dat z ovladačů.

5.3 Knihovna Sixense.dll

Knihovna převzatá ze **Sixense SDK**, která byla naimplementována v jazyce C++ a stará se o komunikaci mezi PC a ovladači **Razor Hydra**. Knihovna načítá data z USB portu, které po inicializaci následně přeposílá.[9]

5.3.1 Funkce sixenseInit()

Inicializační funkce **Sixense** knihovny. Ostatní funkce začnou odpovídat až po této inicializační funkci, dokud není zavolána funkce **sixenseExit().**[9]

5.3.2 Funkce sixenseExit()

Zavírá spojení se **Sixense** knihovnou a ostatní funkce nejsou aktivní, dokud není opět zavolána funkce **sixenseInit**().[9]

5.3.3 Funkce sixenseBaseConnected()

Funkce vrací hodnotu 0, pokud je dokovací stanice řádně připojena k PC. V opačném případě vrací hodnotu -1. [9]

5.3.4 Funkce sixenseGetNumActiveControllers()

Vrací hodnotu rovnu počtu aktivních ovladačů. [9]

5.3.5 Funkce sixenseIsControllerEnabled()

Volání funkce vrací stav připojení referenčního ovladače. Hodnota 1 značí, že ovladač je připojen a 0 naopak nepřipojen. [9]

5.3.6 Funkce sixenseGetNewestData()

Prvním argumentem funkce je struktura **sixenseControllerData**, která obsahuje aktuální data z ovladače (kap. 5.4). Druhým argumentem je specifické ID ovladače, se kterým se pracuje. Funkce končí úspěšně, jestliže byly získány všechny potřebné data struktury a tyto data vrací návratovou hodnotou v podobě struktury sixenseControllerData. [9]

5.4 Třída SixensePlugin

Tato třída slouží k importaci metod výše zmíněné knihovny **Sixense.dll**. Také vytváří datovou strukturu **sixenseControllerData**, která obsahuje přijímaná data z ovladače. Popis jednotlivých prvků struktury: [9]

pos	X, Y a Z pozice ovladače
joystick_x	Horizontální pozice joysticku (-1,0 je plně vlevo, 0 je ve středu a 1,0 je plně vpravo).
joystick_y	Vertikální pozice joysticku (-1,0 je plně dole, 0 je ve středu a 1,0 je plně nahoře).
buttons	Bitový vektor popisující stav tlačítek ovladače.
rot_quat	Aktuální úhly natočení ovladače.
enabled	Jestliže je ovladač připojen, vrací hodnotu 1, naopak vrací 0.

5.5 Třída SixenseInput

Tato třída byla převzata ze **Sixense SDK**. Třída mimo jiné zapouzdřuje třídu **Controller**, do jejíž atributů se načítají prvky z datové struktury **sixenseControllerData**. [9]

5.5.1 Metoda Start()

Při volání této metody se postupně inicializují všechny aktivní ovladače připojené k dokovací stanici a vytvářejí se instance třídy **Controller**. [9]

5.5.2 Metoda Update()

Je hlavní metodou třídy **SixenseInput**, která se stará o načítání hodnot ze struktury **sixenseControllerData** do jednotlivých vytvořených instancí třídy **Controller**. Používá k tomu importované funkce knihovny **Sixense.dll**. [9]

5.6 Třída HydraManager

Tato třída zajišťuje synchronizaci dat mezi robotem a ovladači. Obsahuje aktuální hodnoty pozice robota v podobě třídy **RobCoord.XYZ** a zároveň i ovladačů, které mají data uložená ve třídě Vector3D. Hodnoty pozice robota jsou brány jako referenční a poziční hodnoty ovladačů se k nim přepočítávají na základě dané citlivosti. V metodách **UpdateMotion()**, **ControllerMove()** a **ControllerMoveRight()** je naimplementovaný individuální testovací algoritmus pro řízení robotického ramene. Vzhledem ke struktuře programu je velice jednoduché tento algoritmus obměnit a přizpůsobit konkrétnímu řešenému problému. Testovací algoritmus je řízen pomocí kartézských souřadnic **X, Y, Z** a **U, V, W** robota, ale toto řízení je možné jednoduše změnit např. pro řízení na základě rotace jednotlivých kloubů.

5.6.1 Metoda Init()

Metoda volající inicializační metodu **SixenseInput.Start()**. Navíc vytváří instance tříd **Vector3D** a **RobCoord.XYZ** k ukládání aktuálních pozičních hodnot.

5.6.2 Metoda StartControl()

Je volána z vlákna pro řízení robotického ramene pomocí ovladačů. V první části načítá data ovladačů pomocí metody **SixenseInput.Update()**. Dále vytváří objekty jednotlivých ovladačů, do kterých ukládá načtená data z předchozího kroku. Na tyto vytvořené a naplněné objekty v podobě **SixenseInput.Controller** jsou následně aplikovány algoritmy řízení.

5.6.3 Metoda UpdateMotion()

V této metodě dochází k synchronizaci inicializačních hodnot ovladačů a aktuálních pozičních hodnot robota. V testovacím algoritmu se inicializace těchto hodnot provede při stlačení tlačítka **Start** na daném ovladači. Dojde k vytvoření počátku souřadnic v místě, kde se ovladač právě nachází. Jednotlivé ovladače se inicializují nezávisle na sobě.

Jestliže dojde k inicializaci počátku, metoda začne volat příslušné metody k výpočtu diference pohybu opět na základě daného ovladače.

5.6.4 Metoda ControllerMove()

Metoda, která je součástí testovacího algoritmu a spravuje řízení levého ovladače. Vstupním parametrem metody je třída **SixenseInput.Controller**, která obsahuje aktuální načtená data z ovladače. Poziční hodnoty z této třídy jsou diferenčně přepočteny vůči již inicializovanému počátku z metody **UpdateMotion()** a následně přičteny k aktuálním pozičním hodnotám robota (levý ovladač řídí translační pohyb v osách **X, Y** a **Z**) třídy **RobCoord.XYZ**, se kterými je volána komunikační třída **TcpC3Com** k zaslání požadavku na pohyb robota do vypočtených souřadnic. Je potřeba dodat, že metoda zašle požadavek o pohybu robota jen v případě zmáčknutí **Bumper** tlačítka na daném ovladači.

5.6.5 Metoda ControllerMoveRight()

Obdoba metody **CotrollerMove()** s tím rozdílem, že se vypočítávají hodnoty rotačního pohybu v osách manipulátoru **U, V** a **W** a pohyb je snímán z pravého ovladače.

6. ZÁVĚR

Bakalářská práce obsahovala mimo jiné rozsáhlé teoretické i praktické seznámení s celým konceptem robotického manipulátoru EPSON C4 a řídící jednotky EPSON RC700-A.

Pro bezpečné oživení manipulátoru byl navržen bezpečnostní okruh. Ten však eliminoval některé bezpečnosti prvky, které bylo potřeba vzhledem k laboratorním a realizačním podmínkám opomenout. Vzrostly tak nároky na vyklizení pracovního prostoru manipulátoru a jeho bezprostředního okolí.

K vnějšímu řízení byl navržen ovládací panel s volně programovatelnými tlačítky a sadou LED diod, které jsou zapojeny podle předepsaných pravidel v dokumentaci řídící jednotky. Díky ovládacímu panelu je uživatel schopen řídit základní funkce řídící jednotky bez použití PC a vývojového prostředí EPSON RC+ (v případě sestaveného projektu v řídící jednotce).

Jednotlivé příkazy odesílané komunikační knihovnou EpsonC3Com byly rozšířeny o volitelné parametry CP a ROT k dosažení kontinuálního pohybu ramene manipulátoru. Tento pohyb se však nedokázalo vyladit do 100% plynulého pohybu, z důvodu množství faktorů závisejících na přesnosti projetí daných bodů trajektorie.

Pro plánování trajektorie robota byl vytvořen SW nástroj, který je schopen přehledného vytváření a ukládání trajektorií. Uživatel je schopen vytvořit celou řadu procházejících bodů trajektorie. Tato trajektorie se ukládá do tzv. projektu, ve kterém jsou ukládány mimo zmíněné trajektorie také informace o použitých bodech (v kartézských souřadnicích) a nástrojových offsetech.

Jako druhá možnost plánování trajektorie byly vybrány ovladače Razor Hydra se softwarovou podporou společnosti Sixense Entertainment. Za pomocí ovladačů je uživatel schopen pohybovat s manipulátorem v reálném čase. Tato metoda plánování není určená pro aplikace s extrémní přesností pohybu. Naopak velmi ulehčuje práci při plánování ne tolik přesných trajektorii.

Mezi náměty k dalšímu vývoji práce patří především vytvoření vizualizačního či simulačního nástroje pro přehledné zrekapitulování či virtuální vytvoření plánované trajektorie. Také zavrhnutý nápad řízení pohybu pomocí pohybového senzoru kinect by mohl skrývat velmi zajímavou realizaci na plánování trajektorií robotického manipulátoru.

Seznam literatury:

[1] RC700 Controller manual [cit. 2017-05-29]. Dostupné na URL: http://robots.epson.com/admin/uploads/product_catalog/files/EPSON_RC700_RC700A_Controller_Manual(R9).pdf#zoom=90

[2] RC700 Safety and Installation Controller Manual [cit. 2017-05-29]. Dostupné na URL: http://robots.epson.com/admin/uploads/product_catalog/files/EPSON_RC700_Safety_and_Installation_Manual(R8).pdf#zoom=90

[3] RC700 Drive Unit Controller Manual [cit. 2017-05-29]. Dostupné na URL: http://robots.epson.com/admin/uploads/product_catalog/files/EPSON_RC700DU_RC700DU-A_Controller_Manual(R3).pdf#zoom=90

[4] Epson RC+ 7 0 User's Guide (for RC700 & RC90 Controllers) [cit. 2017-05-29]. Dostupné na URL: http://robots.epson.com/admin/uploads/product_catalog/files/EPSON_RC+70_Users_Guide-RC700_RC90(V71R3).pdf#zoom=90

[5] Epson SPEL+ 7 0 Language Reference Manual (for RC700 & RC90 Controllers) [cit. 2017-05-29]. Dostupné na URL: http://robots.epson.com/admin/uploads/product_catalog/files/EPSON_SPEL+_70_Language_Reference-RC700_RC90(V71R3).pdf#zoom=90

[6] Epson PG Motion Systém Manual (for RC700 & RC90 Controllers) [cit. 2017-05-29]. Dostupné na URL: http://robots.epson.com/admin/uploads/product_catalog/files/EPSON_PG_Motion_System_Manual-RC700_RC90(R2).pdf#zoom=90

[7] FIREŠ, M. Demonstrační úloha pro robotický manipulátor EPSON. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2012. 73 s. Vedoucí diplomové práce doc. Ing. Luděk Žalud, Ph.D.. [cit. 2017-05-29]

[8] Epson C4 Robot Manual[cit. 2017-05-29]. Dostupné na URL: http://robots.epson.com/admin/uploads/product_catalog/files/EPSON_C4_Robot_Manual(R4).pdf#zoom=90

[9] Sixense SDK [cit. 2017-05-29]. Dostupné na URL: http://sixense.com/sixensesdk

Seznam obrázků:

Seznam elektronických příloh:

Elektronická příloha (CD) obsahuje:

- Kompletní projekt SW nástroje EpsonC3Com v2.0 ve vývojovém prostředí Microsoft Visual Studio 2013.
- Kompletní projekt s programem do řídící jednotky ve vývojovém prostředí EPSON RC+
- Video nahrávka s ukázkou funkčnosti
- Zdrojový soubor s textem bakalářské práce v převedeném PDF dokumentu