VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ

FACULTY OF MECHANICAL ENGINEERING

ÚSTAV AUTOMATIZACE A INFORMATIKY

INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

ŘÍZENÍ KOLABORATIVNÍHO ROBOTA UNIVERSAL ROBOTS UR10E PROSTŘEDNICTVÍM PLC OD SPOLEČNOSTI SIEMENS

CONTROL OF THE COLLABORATIVE ROBOT UNIVERSAL ROBOTS UR10E THROUGH PLC FROM SIEMENS COMPANY

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — Bc. Nina Štěrbová

AUTHOR

VEDOUCÍ PRÁCE — Ing. Jakub Kůdela, Ph.D.

SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání diplomové práce

Ústav:	Ústav automatizace a informatiky
Studentka:	**Bc. Nina Štěrbová**
Studijní program:	Aplikovaná informatika a řízení
Studijní obor:	bez specializace
Vedoucí práce:	**Ing. Jakub Kůdela, Ph.D.**
Akademický rok:	2023/24

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma diplomové práce:

Řízení kolaborativního robota Universal Robots UR10e prostřednictvím PLC od společnosti Siemens

Stručná charakteristika problematiky úkolu:

Práce bude zahrnovat rešerši v oblasti systémové integrace, tj. průmyslových komunikačních protokolů (Profinet, Powerlink, EtherNet/IP, apod.), a kolaborativní robotiky s důrazem na koncept Průmyslu 4.0. Teoretická část práce bude zahrnovat také rešerši v oblasti optimalizace průmyslových robotů s ohledem na minimalizaci dráhy, energie, apod. Předmětem práce bude rešerše současného stavu robotické buňky EDUset ONE, výběr komunikačního protokolu pro systémovou integraci kolaborativního robota UR10e prostřednictvím programovatelného logického automatu (PLC) od společnosti Siemens, a v neposlední řadě návrh a implementace modulární knihovny pro řízení a sběr dat. Dílčí částí práce bude statická optimalizace robota s ohledem na minimalizaci dráhy (příp. energie) využívající navrženou knihovnu pro řízení. Závěr práce bude věnován ověření funkčnosti navrženého řešení prostřednictvím simulace a také v interakci s reálnou robotickou buňkou.

Práce bude realizována ve spolupráci s firmou Intemac Solutions s.r.o., která poskytne řídicí prostředky a část know–how v pojednávané oblasti.

Cíle diplomové práce:

– Proveďte rešerši v oblasti systémové integrace a kolaborativní robotiky s důrazem na koncept Průmyslu 4.0.
– Analyzujte dostupné metody pro problém optimalizace průmyslových robotů s ohledem na minimalizaci dráhy, energie, apod.
– Proveďte rešerši současného stavu robotické buňky EDUset ONE.
– Zvolte komunikační protokol pro systémovou integraci kolaborativního robota UR10e prostřednictvím programovatelného logického automatu (PLC) od společnosti Siemens.
– Navrhněte a implementujte modulární knihovnu pro řízení a sběr dat.
– Navrhněte a implementujte program pro statickou optimalizaci robota s ohledem na minimalizaci dráhy (příp. energie) využívající navrženou knihovnu pro řízení.
– Ověřte funkčnost vytvořeného řešení prostřednictvím simulace a také v interakci s reálnou robotickou buňkou.

Seznam doporučené literatury:

KOLÍBAL, Zdeněk, Roboty a robotizované výrobní technologie. Brno: Vysoké učení technické v Brně - nakladatelství VUTIUM, 2016. ISBN 978-80-214-4828-5.

SICILIANO, Bruno a KHATIB, Oussama, ed. Springer handbook of robotics. 2nd edition. Berlin: Springer, 2016. ISBN 978-3-319-32550-7.

Kevin M. Lynch and Frank C. Park, Modern Robotics: Mechanics, Planning, and Control. Cambridge University Press, 2017. ISBN 9781107156302.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2023/24

V Brně, dne

L. S.

prof. Ing. Radomil Matoušek, Ph.D. ředitel ústavu	doc. Ing. Jiří Hlinka, Ph.D. děkan fakulty

ABSTRAKT

Tato práce se zabývá systémovou integrací a kolaborativní robotikou v kontextu Průmyslu 4.0 se zaměřením na optimalizaci kolaborativních robotů. Práce obsahuje analýzu současného stavu robotické buňky EDUset ONE a výběr komunikačního protokolu pro systémovou integraci kolaborativního robota UR10e prostřednictvím programovatelného logického automatu (PLC) od společnosti Siemens. Hlavním cílem je návrh a implementace modulární knihovny pro řízení a sběr dat a dále návrh programu pro statickou optimalizaci trajektorie robota s ohledem na minimalizaci dráhy a energie. Funkčnost navrženého řešení je ověřena na reálné robotické buňce.

ABSTRACT

This work deals with system integration and collaborative robotics in the context of Industry 4.0, focusing on the optimization of collaborative robots. The work includes an analysis of the current state of the EDUset ONE robotic cell and the selection of a communication protocol for the system integration of the UR10e collaborative robot through a programmable logic controller (PLC) from Siemens. The main goal is to design and implement a modular library for control and data collection, and further to design a program for static optimization of the robot's trajectory with regard to minimizing the path and energy. The functionality of the proposed solution is verified on a real robotic cell.

KLÍČOVÁ SLOVA

Systémová integrace, Kolaborativní robotika, Průmysl 4.0, Optimalizace průmyslových robotů, UR10e, Siemens PLC, Nevergrad.

KEYWORDS

System Integration, Collaborative Robotics, Industry 4.0, Optimization of Industrial Robots, UR10e, Siemens PLC, Nevergrad.

ÚSTAV AUTOMATIZACE A INFORMATIKY

2024

BIBLIOGRAFICKÁ CITACE

ŠTĚRBOVÁ, Nina. *Řízení kolaborativního robota Universal Robots UR10e prostřednictvím PLC od společnosti SIEMENS*. Brno, 2024. Dostupné také z: https://www.vut.cz/studenti/zav-prace/detail/156952. Diplomová práce. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav automatizace a informatiky, Vedoucí práce: Ing. Jakub Kůdela, Ph.D.

ČESTNÉ PROHLÁŠENÍ

Prohlašuji, že tato diplomová práce je mým původním dílem, vypracovala jsem ji samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury.

Jako autorka uvedené práce dále prohlašuji, že v souvislosti s vytvořením této práce jsem neporušila autorská práva třetích osob, zejména jsem nezasáhla nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědoma následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků.

V Brně dne 21. 5. 2024 .....

Nina Štěrbová

PODĚKOVÁNÍ

Děkuji Ing. Jakubovi Kůdelovi, Ph.D. za vedení diplomové práce a odborné konzultace v oblasti optimalizace. Dále bych chtěla poděkovat Ing. Romanovi Parákovi a Ing. Peteru Janusovi ze společnosti Intemac Solutions s.r.o. za odborné konzultace v oblasti robotiky.

OBSAH

1 ÚVOD

Průmyslová automatizace se v posledních letech dynamicky rozvíjí. Vzrůstá tak potřeba efektivní integrace různých technologií do jednotného systému, aby se dosáhlo maximální efektivity. Jednou z velkých výzev v této oblasti je integrace průmyslových zařízení, jako jsou roboti, PLC (programovatelné logické automaty) a další.

Výše popsané adresuje koncept Průmyslu 4.0, který se zaměřuje na propojení všech zařízení v továrně do sítě, čímž umožňuje jejich vzájemnou komunikaci a sdílení dat. To vede k optimalizaci procesů, zvýšení produktivity a celkové efektivity výroby.

V tomto kontextu je klíčové zajistit možnost řídit robota jiným způsobem, než programovat každou úlohu zvlášť, protože přístup je neefektivní a časově náročný, obzvlášť v dynamicky se měnícím prostředí. Řešením by bylo řídit robota prostřednictvím PLC, které je napojené na jiná zařízení ve výrobě i na vyšší vrstvy podniku. Díky novému řešení bude možné dynamicky měnit zadání robotické úlohy dle aktuálních potřeb a sbírat aktuální data z robota.

V současné době se vyvíjí rozhraní mezi konkrétními výrobci robotů a PLC, avšak takovéto řešení mezi dvěma světovými výrobci Universal Robots a Siemens stále chybí. Proto cílem této diplomové práce je navrhnout a implementovat modulární knihovnu pro komunikaci mezi kolaborativním robotem Universal Robots UR10e a PLC od společnosti Siemens, která zajistí řízení robota prostřednictvím PLC a sběr dat z robota. Nedílnou součástí tohoto úkolu je výběr vhodného komunikačního protokolu, který zajistí výměnu dat v reálném čase.

Snahou průmyslové automatizace je mimo jiné zlepšovat současná řešení, zefektivňovat práci a snižovat provozní náklady, proto dalším cílem této práce je vytvořit program, který navrhne trajektorii robota, která minimalizuje ujetou dráhu nebo spotřebu energie. V případě optimalizace je stěžejní vybrat vhodnou metodu, a protože neexistuje jedna universální, vybere se několik algoritmů, s nimiž se následně provedou experimenty.

V úvodní kapitole 2 je přehled současného stavu Průmyslu 4.0 a komunikačních protokolů potřebných pro systémovou integraci. Kapitola zároveň obsahuje výběr komunikačního protokolu a popis současného stavu robotického buňky EDUset ONE společnosti Intemac Solutions s.r.o., se kterou byla v rámci této závěrečné práce navázána spolupráce.

Kapitola 3 je věnována tvorbě a implementaci modulární knihovnu pro řízení kolaborativního robota prostřednictvím PLC včetně jejího testování.

V kapitole 4 jsou představeny použité optimalizační metody a kapitola 5 obsahuje návrh a implementaci programu pro statickou optimalizaci trajektorie robota včetně výsledků experimentů.

2 PŘEHLED SOUČASNÉHO STAVU V OBLASTI PRŮMYSLU 4.0

V rámci této kapitoly je nastíněn koncept Průmyslu 4.0, který bude dále rozšířen o kapitoly zaměřující se na vertikální systémovou integraci. Konkrétně podkapitoly 2.2 a 2.3 přibližují vybrané průmyslové komunikační protokoly, resp. kolaborativní robotiku. Závěrečná podkapitola 2.4 je věnována popisu robotické buňky EDUset ONE, která je navržena dle zmiňovaného konceptu.

2.1 Průmysl 4.0

Průmysl 4.0 byl představený německou vládou v roce 2011 [1] v reakci na rozvíjející se informační a komunikační technologie a reprezentuje transformaci tradičního centralizovaného průmyslu do inteligentního decentralizovaného systému řízeného daty. Základem jsou technologie jako umělá inteligence, kyber-fyzikální systémy (CPS), internet věcí (IoT), *cloud computing*, *big data*, robotika, architektury orientované na služby (SOA) a další. Průmysl 4.0 se zaměřuje na digitalizaci a automatizaci výrobních procesů, což vede k vytvoření sítě inteligentních zařízení. Tato zařízení jsou pak schopna mezi sebou komunikovat, vyměňovat si informace napříč celým produkčním řetězcem a to v reálném čase bez nutnosti intervence člověka [2].

Cílem této strategie je pomocí digitalizace propojit fyzický a virtuální svět pro efektivnější shromažďování a analýzu dat. Dalším je interoperabilita umožňující plynulou a neustálou výměnu dat mezi všemi zařízeními a technologiemi výrobního procesu se schopností reagovat na změny v reálném čase. A v neposlední řadě je to modularita řešení, která umožňuje adaptaci a rozšiřování systémů [2].

Realizace konceptu Průmyslu 4.0 má řadu pozitivních důsledků. Obecně dochází k zvýšení výkonu, přesnosti a efektivity, kdy je díky optimalizaci procesů maximalizována produkce s ohledem na minimalizaci nevyužitých zdrojů a produktů. Dalším důsledkem je zvýšení transparentnosti, která vede k větší kontrole nad výrobními procesy, což je umožněno monitorováním a následnou analýzou všech dílčích procesů [2].

Obecně je průmyslová automatizace strukturována do hierarchie o čtyřech úrovních. Shora je to:

1) **informační úroveň**, kde dochází k vytváření produkčních plánů za základě požadavků zákazníků nebo analýzy trhu;
2) **úroveň kontroly a řízení procesu (SCADA systémy)**, kde se kontroluje správné nastavení parametrů, údržba a kvalita; dále do této úrovně spadá monitoring, shromažďování a archivace dat;

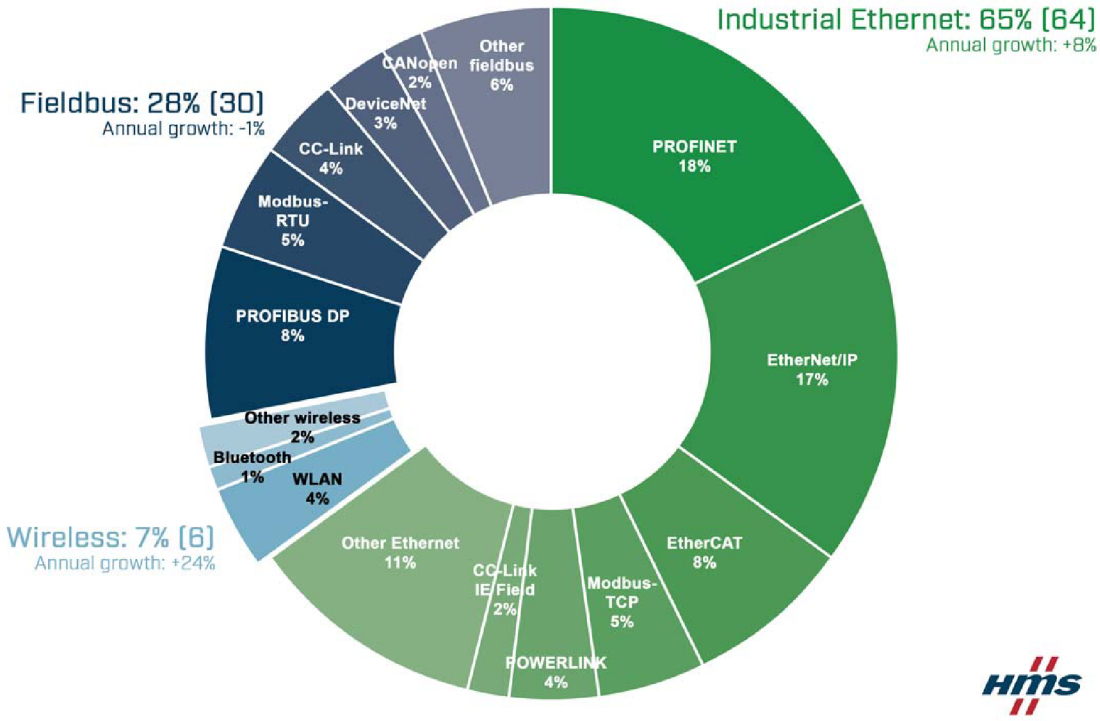
3) **úroveň řízení**, kde hlavní řídicí jednotkou je PLC a pro interakci operátora se strojem, resp. procesem slouží HMI (*Human Machine Interface*);
4) **provozní úroveň**, kam spadají senzory, akční členy a další podobná zařízení pro sběr dat a ovládání procesů [3].

S ohledem na zmíněnou strukturu je při integraci klíčové zajistit plynulý tok dat jak v horizontálním, tak vertikálním směru [3]. Komunikace PLC se zařízeními na řídicí a provozní úrovni probíhá prostřednictvím I/O zařízeních, provozních sběrnic a sítí průmyslového Ethernetu. Interakce je zajištěna v podobě přenosu malých datových paketů vysokou přenosovou rychlostí v reálném čase. Důležitými požadavky jsou pak interoperabilita mnoha systémů a zabezpečený síťový komunikační protokol [4].

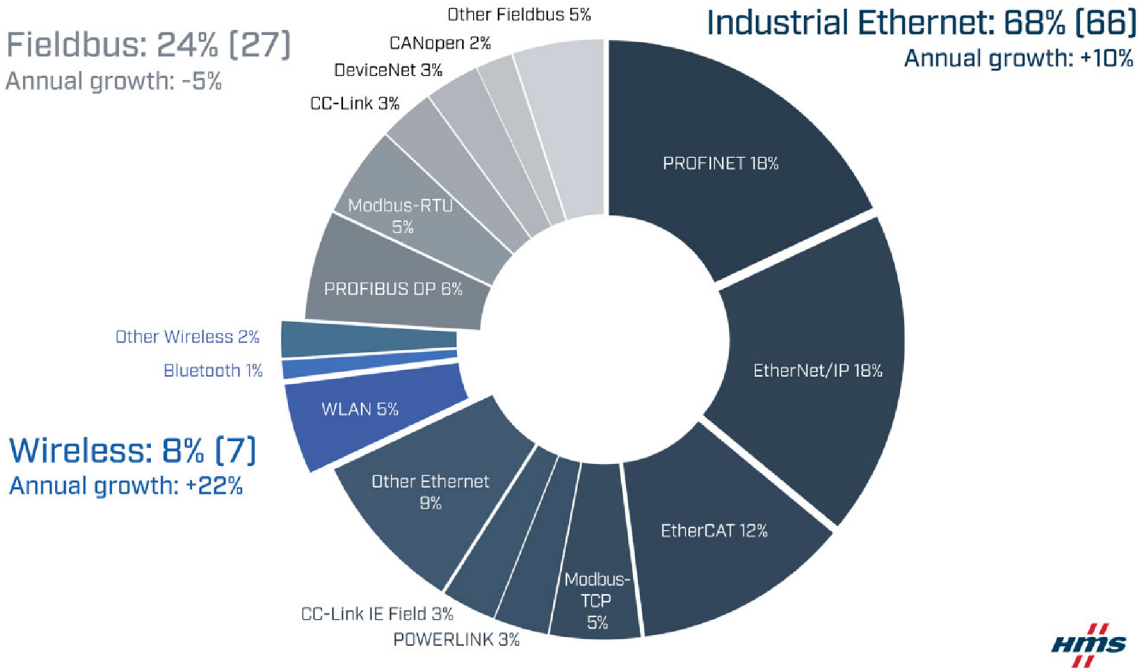
Jako podnikové komunikační sítě se označují sítě, které propojují řídicí úroveň s vyššími. Požadavky na tuto síť bývají v porovnání s předchozí zmiňovanou odlišné. Například zde není přísný nárok na rychlost a čas přenosu [4].

2.2 Průmyslové komunikační protokoly

V posledních letech se dominantním komunikačním standardem v průmyslové automatizaci čím dál tím víc stává průmyslový Ethernet [4]. O jeho vzrůstající popularitě vypovídá každoroční analýza průmyslových sítí prováděná společností HMS Networks. Zatímco ještě k roku 2017 byl podle [5] poměr zastoupení sběrnic (48 %) a průmyslového Ethernetu (46 %) poměrně vyrovnaný, viz Obr. 1, k roku 2023 se tento poměr výrazně změnil [6]. Využití sběrnic kleslo na 24 % a průmyslový Ethernet vzrostl na 68 %. Zbylá procenta přísluší bezdrátových sítím, viz Obr. 2. Výběr konkrétních protokolů závisí na aplikaci a pozici v automatizačním systému. Několik vybraných ethernetových protokolů bude dále popsáno v podkapitolách 2.2.1-2.2.4.



Obr. 1: Analýza průmyslových sítí 2017 [5]



Obr. 2: Analýza průmyslových sítí 2023 [6]

2.2.1 PROFINET

PROFINET (PROcess FIeld NETwork) je průmyslový komunikační protokol spravovaný asociací Profibus International (PI). Vychází z norem IEC 61158, IEC 61784 a IEC 62061/ISO 13849-1, což zajistí interoperabilitu a konformitu standardu s jinými automatizačními zařízeními. PROFINET podporuje vysokorychlostní deterministickou komunikaci v reálném čase, která umožní precisní a deterministické řízení v aplikacích průmyslové automatizace, a zároveň garantuje spolehlivou výměnu dat s nízkým zpožděním. Nabízí škálovatelnou architekturu, která zjednodušuje rozšiřování nebo modifikace systému. PROFINET podporuje nejrůznější síťové topologie jako je sběrnicová, kruhová nebo hvězdicová [7, 8, 9].

Komunikace PROFINETu je založena na standardním Ethernetu, díky čemuž je s touto technologií schopen koexistovat v rámci jedné kabelové sítě. Výhodou Ethernetu je možnost přístupu do systému prakticky odkudkoli, což s sebou ale přináší bezpečnostní rizika. PROFINET tyto problémy adresuje ve své bezpečnostní specifikaci, která se mimo jiné věnuje:

1) **ochraně proti chybám a nesprávnému provozu** - mechanismy pro detekci a prevenci chyb nebo nesprávných operací, které by mohly ohrozit integritu sítě nebo narušit její funkci;
2) **prevenci neoprávněného přístupu** - implementace přístupových kontrol, autentizačních protokolů a šifrovacích metod, které zabraňují neoprávněnému přístupu k síti;
3) **využití ověřených a certifikovaných bezpečnostních standardů** - využití certifikovaných bezpečnostních standardů, jako jsou firewally a virtuální privátní sítě (VPN) [8].

PROFINET získává informace o I/O zařízeních prostřednictvím kontroléru z GSD (*General Station Description*) souborů. Prvně se projekt v rámci SW nakonfiguruje a následně nahraje do kontroléru, což nastaví komunikaci mezi zařízeními. Výhodou je možnost měnit konfiguraci zařízení i za provozu. Datová výměna probíhá cyklicky podle intervalu předem daného kontrolérem. Vedle cyklických dat dochází k výměně i acyklických dat, která slouží k diagnostice [8].

Komunikační standard nabízí širokou škálu diagnostických funkcí, které umožňují jak lokální, tak vzdálené připojení. Diagnostické přehledy zobrazují informace o zařízení, modulech, kanálu a přerušení s různými úrovněmi detailů. K přehledům se dá přistupovat přes standardizované displeje nebo webové rozhraní [8].

PROFINET je definován v několika komunikačních kanálech, které mohou být použity zároveň. Jsou to:

1) **Standardní TCP/IP** - tento kanál je vhodný pro nedeterministické funkce jako parametrizace, video/audio přenos a přenos dat do vyšších úrovní systému;
2) **PROFINET RT (*Real Time*)** - tento kanál zajistí deterministickou komunikaci pro automatizační aplikace v rozsahu 1 - 10 ms;
3) **PROFINET IRT (*Isochronous Real Time*)** - prostřednictvím prioritizace signálů a rozvrženému plánování zajistí tento kanál přesnou synchronizaci, proto se typicky využívá pro řízení pohybu;
4) **TSN (*Time Sensitive Networking*)** - dalším rozvíjejícím se kanálem je PROFINET s rozšířením o TSN protokoly, které ještě více snižují zpoždění a ztrátovost dat při přenosu [8, 10].

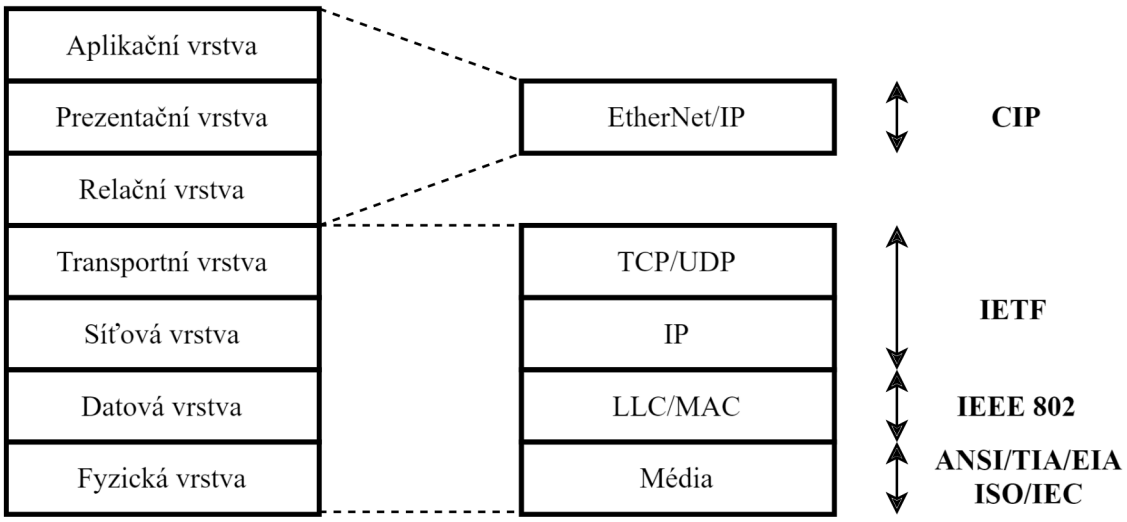
Standard je postaven na komunikačním modelu zprostředkovatel/odběratel (*Provider/Consumer*). Model vychází z jednoho centrálního zařízení figurujícího jako zprostředkovatel a decentralizovaných zařízení coby odběratelů. Zprostředkovatel disponuje procesními daty, která zprostředkovává odběratelům. Komunikace mezi zprostředkovatelem a odběratelem je prvně sestavena v rámci tzv. aplikačního vztahu (AR - *Application Relationship*) a následně tzv. komunikačních vztahů (CR - *Communication Relationships*). Jakmile je navázáno spojení, datový provoz a správa alarmů je nakonfigurována a zařízení si mohou začít vyměňovat data v modu plného duplexu. Ke komunikaci se dají použít různá přenosová média jako měď, optická vlákna, bezdrátové přenosy a další [7, 9].

2.2.2 EtherNet/IP

EtherNet/IP byl v roce 2011 představen organizacemi ODVA (Open DeviceNet Vendors Association) a ControlNet International jako nový průmyslový komunikační protokol patřící do skupiny standardů vycházející z CIP (*Common Industrial Protocol*). EtherNet/IP kombinuje CIP spolu se standardním Ethernetem, což umožňuje propojení napříč podnikem. Protokol je standardizován jako IEC 62413 [11, 12].

Protokol je postaven na modelu producent/konzument, který umožňuje plynulou výměnu, konfiguraci a sběr dat napříč jednou sítí nebo propojenými CIP sítěmi [11]. EtherNet/IP vychází z OSI (*Open Systems Interconnection*) modelu, který definuje implementaci síťových protokolů na sedmi úrovních [12], jak je uvedeno na Obr. 3.

CIP tvoří aplikační vrstvu. Každé zařízení v síti je popsané skupinou objektů, přičemž každý objekt musí obsahovat atributy (data), služby (příkazy) a specifikaci funkcí (reakce na události). Existují tři kategorie objektů: povinné, aplikační a objekty definované výrobcem. Mezi povinné patří:



Obr. 3: OSI model protokolu EtherNet/IP, podle [13]

1) objekt k identifikaci zařízení (*identification object*);
2) objekt ke specifikaci předávání zpráv (*message router object*);
3) objekt pro správu spojení (*connection object*);
4) jeden a více objektů s parametry konfigurace komunikační sítě (*network link object*) [11].

Aplikační objekty jsou vázané na konkrétní zařízení a jejich funkce, a tvoří tak profil těchto zařízení. Každé zařízení má svůj elektronický popis zařízení (EDS - *Electronic Device Sheets*) potřebný ke konfiguraci EtherNet/IP sítě a k propojení zařízení. Jedná se o textový soubor obsahující typ a verzi objektů, identifikační údaje a konfigurovatelné síťové parametry [11].

EtherNet/IP podporuje dva základní typy komunikace:

1) **explicitní** - spojení jsou přímá mezi dvěma zařízeními a realizována prostřednictvím TCP/IP;
2) **implicitní** - spojení jsou realizována prostřednictvím UDP/IP a slouží k cyklickému přenosu vstupně-výstupních dat, přičemž se může využít jako unicast nebo mutlicast mod. [11, 14]

Díky základu v Ethernetu je EtherNet/IP z pohledu síťové architektury flexibilní, tzn. není dán maximální počet zařízení v síti, která je navíc modifikovatelná a rozšiřitelná. Dále je kompatibilní s různými druhy přenosových médií (měď, optické vlákno, bezdrátové přenosy) a síťovými topologiemi (hvězda, linka, a kruh) [12].

2.2.3 EtherCAT

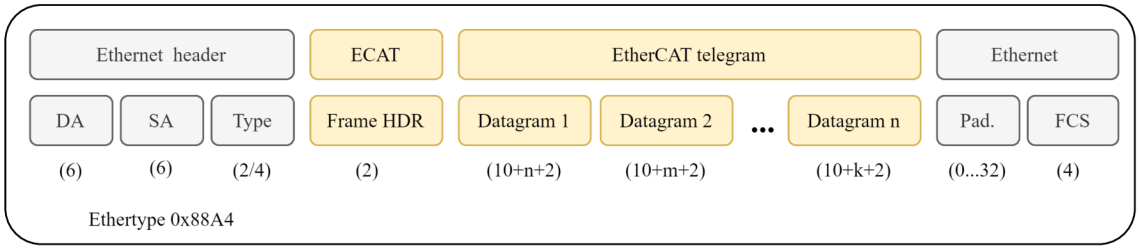
[15] EtherCAT (Ethernet for Control Automation Technology) je průmyslová komunikační technologie představená firmou Beckhoff Automation v roce 2003. Dnes je pod správou EtherCAT Technology Group (ETG), kterou tvoří výrobci i koncoví uživatelé. Protokol je standardizován v rámci IEC 61158 a je vhodný mimo jiné pro real-time automatizační aplikace. Vývoj EtherCATu se zaměřuje na krátké cykly ($\leq$ 100 µs), nízké kolísání odezvy (*jitter*) a nízké ceny potřebného hardwaru.

Protokol je postaven na komunikačním modelu Master/Slave. Master zařízení používá standard Ethernet MAC (*Media Access Controller*) bez nutnosti dalšího komunikačního procesoru. Díky tomu může být masterem jakékoli hardwarové zařízení, které má Ethernet port, bez ohledu na použitý operační systém nebo aplikační software. Slave zařízení potřebují EtherCAT Slave Controller (ESC), díky kterému jsou schopna zpracovat datové rámce přímo na hardwaru a nezatěžovat síť.

EtherCAT master řídí komunikaci v síti a posílá datagramy slave zařízením. Slave zařízení tato data přijímají a na základě cílové adresy si vyberou pouze ty informace, které jsou pro ně relevantní. Poté do datového rámce vloží vlastní data a odešlou ho zpět masterovi. Pouze master má právo vytvářet nové datové rámce, narozdíl od slave zařízení, která je mohou jen přeposílat dál. Tento koncept zamezí nedeterministickým zpožděním a zaručí real-time vlastnosti.

EtherCAT využívá standardních Ethernetových rámců, viz Obr. 4. K rozpoznání rámce slouží Iidentifikátor (0x88A4). Dále EtherCAT rámec obsahuje jeden a více datagramů. Hlavička datagramu určuje, jaký typ přístupu master vyžaduje:

1) čtení/zápis/čtení i zápis;
2) přístup k určitému slave zařízení prostřednictvím přímého adresování nebo přístup k více slave zařízením s implicitním adresováním.



Obr. 4: EtherCAT rámec, podle [15]

Implicitní adresování je určeno pro cyklickou výměnu procesních dat. Datagram adresuje konkrétní oblast procesu pokrytého EtherCAT sítí. Během inicializace sítě je každému slave zařízení přiřazena jedna nebo více adres. Pokud je více zařízení ve stejné oblasti, mohou být adresovány jedním datagramem. Datagramy

obsahují veškeré přístupové informace a pro každý datagram je vymezeno 4 GB adresového prostoru. V rámci implicitního adresování je možné, aby master adresoval slave zařízení prostřednictvím pozice v síti, čehož se využívá při spouštění sítě k identifikaci topologie. Po kontrole nastavení sítě přiřadí master každému uzlu pevnou adresu, přes kterou se k němu bude dostávat. Toto umožňuje masteru se k zařízení dostat i po případné změně topologie. EtherCAT podporuje většinu topologií: linka, strom, hvězda a řetězec.

Výhodou otevřeného rozhraní je možnost integrovat jakýkoli IT protokol, např. OPC UA, MQTT nebo AMQT, a to v rámci mater zařízení nebo přímo do slave zařízení. Díky této vlastnosti EtherCAT umožňuje navázat systém od spodních vrstev se senzory až na cloud bez nutnosti hardwarových nebo softwarových změn. Vývoj protokolu probíhá i se zaměřením na TSN, což napomáhá k zlepšení komunikace v reálném čase. A v neposlední řadě, EtherCAT Technology Group byla jedna z prvních organizací vyvíjející průmyslové komunikační protokoly, která byla členem OPC Foundation, takže EtherCAT je taktéž doplněn o OPC UA protokol. Zmíněné vlastnosti vedou k vysokému výkonu a flexibilitě protokolu, umožňují horizontální i vertikální komunikaci, což z něj dělá konkurence schopný protokol v kontextu Průmyslu 4.0, a je zároveň jedním z nejrozšířenějších průmyslových komunikačních protokolů [15].

2.2.4 POWERLINK

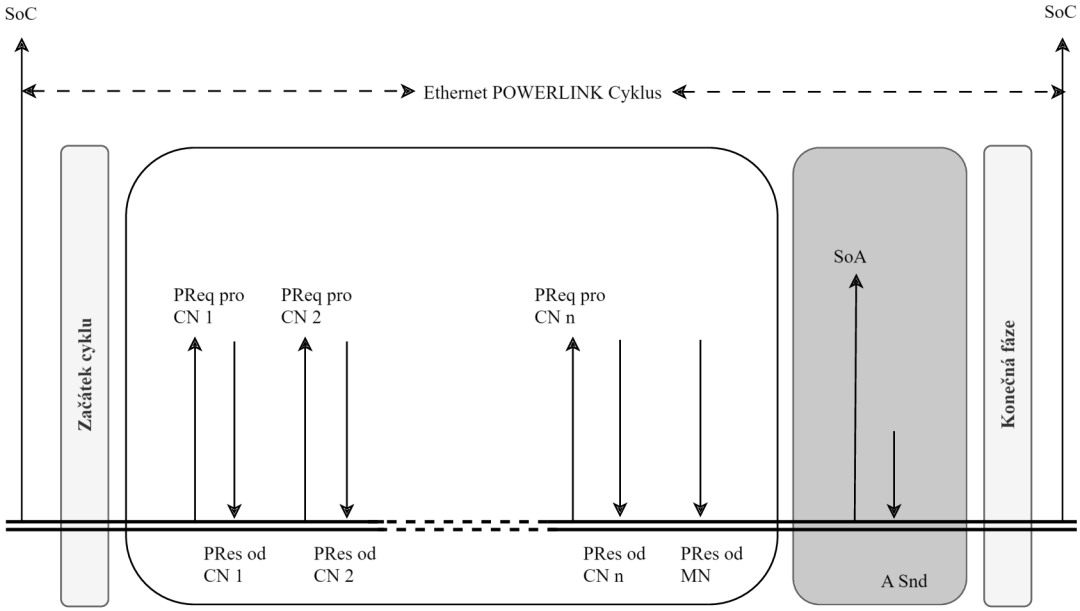
Ethernet POWERLINK (EPL) je komunikační protokol vyvinutý pro řízení průmyslových aplikací v reálném čase. POWERLINK je spravován Ethernet Powerlink Standardization Group (EPSG) a normován podle ISO a IEC 61784. Protokol staví na Ethernetu, což zajistí deterministickou datovou výměnu v časových cyklech kratších než 10 ms s velmi nízkým kolísáním odezvy [16, 17].

Protokol využívá upravený komunikační model Master/Slave, kde změna tkví v možnosti slave zařízení sdílet informace nejen s masterem, ale i dalšími slave zařízeními [18]. POWERLINK definuje dva typy zařízení: Managing Node (MN) a Controlled Node (CN). Master je typu MN, je v síti jediný a řídí datový tok tak, že nechá TCP/IP rámce procházet sítí pouze v určitých časových okamžicích. Tím zajišťuje přenos v reálném čase. Spojení master-slave je založeno na stále se opakujících cyklech, které jsou rozděleny do čtyř částí, viz Obr. 5 [16, 17]:

1) **Začátek cyklu (SoC)** - MN pomocí broadcastu rozešle všem CNs informaci o začátku cyklu, abych došlo k synchronizaci všech zařízení;
2) **Isochronní úsek** - MN se postupně dotazuje každého CN na výstupní data pomocí tzv. rámce žádosti o přijetí změn (Poll Request frame - PReq). Jakmile CN přijme tento rámec, vyšle broadcast odpověď (Poll Response - PRes) v po-

době vstupních dat. Na konci isochronního úseku MN opět broadcastově vyšle zprávu o začátku asynchronního úseku;

3) **Asynchronní úsek (SoA)** - řetí část cyklu je acyklická. MN udělí konkrétnímu CN povolení k odeslání acyklické zprávy (A Snd). Když NC dokončí odesílání zprávy, začíná závěrečná fáze;
4) **Klidová fáze** - po zbývající čas MN čeká, než bude moct začít nový cyklus [16].



Obr. 5: Ethernet Powerlink cyklus, podle [16]

Díky základům na Ethernetu disponuje POWERLINK SCNM segmentem (*Slot Communication Network Management*), který zajišťuje bezkoliznost, čímž je možné zaručit deterministický datový tok v reálném čase [16].

Také POWERLINK je průmyslový protokol vhodný do konceptu Průmysl 4.0, protože je stavěný pro decentralizované systémy, kde zaručí flexibilitu v rámci integrace, modifikace a rozšíření systému. POWERLINK je volně dostupný v plné podobě jako openPOWERLINK a je snadno implementovatelný. Podle studie [18] má POWERLINK dobře řešené zabezpečení. V porovnání s EtherNet/IP má větší odolnost vůči kyber-útokům a rychleji se obnoví do původního stavu [18].

2.3 Kolaborativní robotika

Kolaborativní roboty (tzv. coboty) se staly významnou součástí Průmyslu 4.0, přestože jejich myšlenka byla představena ještě před čtvrtou průmyslovou revolucí. [19]. V porovnání se standardními průmyslovými roboty nabízí větší flexibilitu (jeden cobot může vykonávat velmi různorodé činnosti), produktivitu a mobilitu (díky

menším rozměrům a váze mohou být snadno přemisťovány po výrobě dle potřeby). Kolaborativní roboty jsou navrhnuty tak, aby byly schopny pracovat ve stejném prostředí spolu s lidmi [20]. Zpravidla se to týká aktivit, které jsou pro člověka rutinní nebo fyzicky náročné a které robot zvládne vykonat efektivněji [21]. Tabulka 1 nabízí přehled silných a slabých stránek při vykonávání práce člověkem v porovnání s robotem [20].

Tab. 1: Porovnání práce člověk vs. robot, podle [20]

Člověk		**Robot**	
Výhody	**Nevýhody**	**Výhody**	**Nevýhody**
Zručnost	Slabost	Síla	Neznalost procesu
Flexibilita	Únavnost	Výdrž	Absence zkušeností
Kreativita	Nepreciznost	Preciznost	Absence kreativity
Schopnost rozhodování	Nízká produktivita	Vysoká produktivita	Neschopnost rozhodování

Kolaborativní roboty zpravidla sdílí s člověkem stejný pracovní prostor, proto je nutné aplikovat speciální bezpečnostní opatření [22]. V této souvislosti se autoři [23] domnívají, že je nutné, aby Průmysl 4.0 zohledňoval implementaci tří základních typů bezpečnostních systémů v kontextu kolaborativní robotiky, a sice systém, který:

1) je schopen kvantifikovat míru úrazu po kolizi člověk-robot;
2) minimalizuje úrazy při spolupráci člověk-robot;
3) předchází kolizím člověk-robot.

K dosažení výše zmíněného kolaborativní roboty často disponují funkcemi jako například:

1) **safety stop** - jakmile robot detekuje člověka (překážku) ve svém pracovním prostoru, zastavuje;
2) **hand guiding** - člověk může bezpečně robota naučit novou trajektorii. Výhodou je bezpečná spolupráce člověk-robot a to bez nutnosti znalosti programování;
3) **speed & separation monitoring** - v případě detekce člověka robot upraví rychlost a případně přizpůsobí pohyby;
4) **power & force limitation** - v případě neočekávaného kontaktu s člověkem, robot automaticky sníží svou sílu, aby nedošlo k újmě na zdraví. K dosažení toho cíle se používají různé metody, například monitorování proudu, snímače síly a momentu nebo speciální jiskrově bezpečná konstrukce [20].

Práce ve výrobě mnohdy vyžaduje pohyby, kdy dochází u člověka k přetěžování určitých partií těla, což z dlouhodobého hlediska může být nebezpečné. V souvislosti s ergonomií a bezpečností práce se v průmyslu vyskytují kolaborativní roboty, které asistují při fyzických prací. Konkrétním příkladem jsou exoskeletony [21]. Jedná se o podpůrné zařízení, které si člověk nasadí. Exoskeleton, který je navržen tak, aby se přizpůsobil pohybu lidského těla, poskytuje podporu při fyzických úkonech, funguje jako ochranný prvek a/nebo rozšiřuje lidské schopnosti, jako je zvýšení síly a citlivosti. Může pokrývat konkrétní části těla nebo být celotělní. Článek [24] uvádí využití exoskeletonů v automotiv průmyslu. Jedná se o exoskeleton pro horní část těla, který zprostředkovává podporu při provádění operací nad úrovní hlavy a odlehčuje namáhaným svalům a kloubům. Dalším konkrétním příkladem z automobilového průmyslu je exoskeleton pro pracovníky u pásu, kteří musí celý den stát. Tento robot se v případě potřeby přizpůsobí a slouží jako židle. Člověk si tím může odpočinout, aniž by ztratil potřebnou mobilitu. Mimo průmysl mají exoskeletony velký význam v oblasti rehabilitace a zdravotnictví [24].

Důsledkem změn v návaznosti na Průmysl 4.0 a implementací robotů potřeba člověka úplně nemizí. Mění se ale charakter práce, který se začíná více zaměřovat na znalosti, a na pracovníky jsou tak kladeny větší nároky. Roboty najdou uplatnění i v souvislosti s podporou kognitivní práce [20].

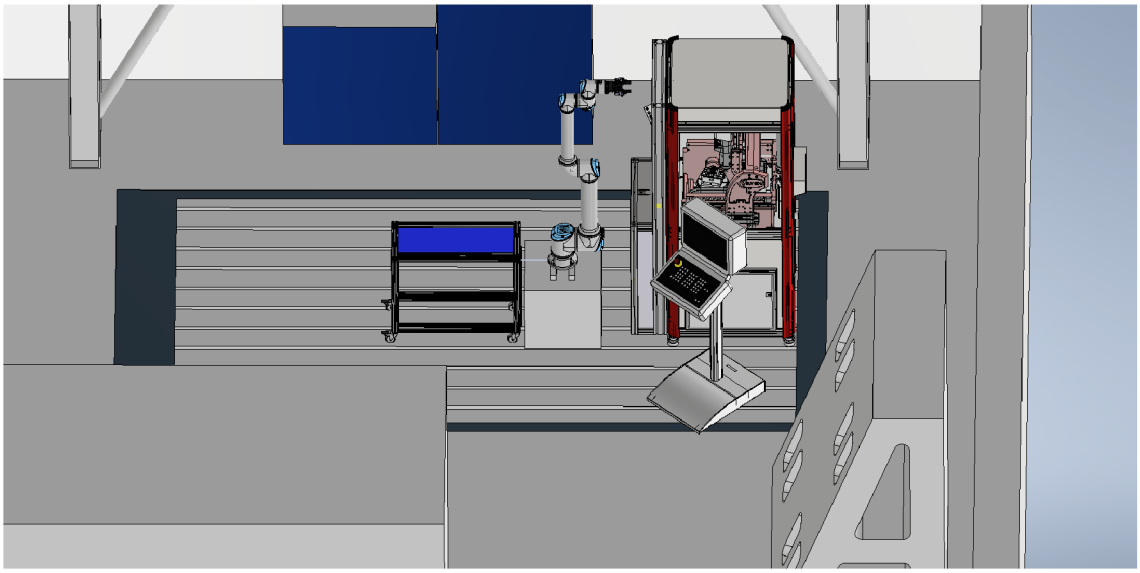
Význam kolaborativní robotiky exponenciálně roste. Výzkum *Global Collaborative Robot (Cobot) Market: Focus on Payload, Application and Industry - Analysis and Forecast, 2019-2024* předpovídá, že roční míra růstu (CAGR) je 60,85 %. Vzhledem k návratnosti investice jsou pořizovací ceny cobotů nízké. Díky tomu, snadné integraci a vysoké flexibilitě mají kolaborativní roboty rychlejší návratnost v porovnání s průmyslovými roboty [20].

2.4 Robotická buňka EDUset ONE

V souladu s kapitolou 2.3 se robotická buňka stává klíčovou v kontextu bezpečnosti práce, kde kolaborativní robot a člověk sdílí stejný pracovní prostor. Zpravidla se jedná o vyznačené místo dedikované konkrétní úloze, které figuruje robot. Toto uspořádání přispívá k zvýšené bezpečnosti, jelikož buňky mohou být vybaveny bezpečnostními mechanismy, které se aktivují při detekci člověka. Navíc, vymezený prostor intuitivně vede člověka ke zvýšené obezřetnosti.

Robotická buňka s pracovním názvem EDUset ONE, kterou spravuje firma Intemac Solutions, s.r.o., slouží v rámci této práce jako fyzické vývojové, resp. testovací zázemí. Obr. 6 zachycuje rozvržení robotické buňky. Hlavní komponenty jsou:

1) **kolaborativní robot** UR10e od Universal Robots. Jedná se o šestiosého kolaborativního robota s dosahem 1300 mm a nosností 12.5 kg [25], který je vhodný pro nejrůznější typy aplikací, např. paletizace, svařování, montáž, inspekce kvality, *pick&place* úlohy a další.
2) **frézka** SLV EDU od Solid Vision s pěti osami je vhodná k obrábění tvarově náročnějších dílů [26]. Zařízení disponuje servomotory a CNC řídicím systémem Sinumerik ONE od společnosti Siemens.
3) **PLC** SIMATIC S7-1500 od společnosti Siemens se spolu s hlavním vypínačem buňky a dalšími I/O moduly nachází v technické skříni.



Obr. 6: Rozvržení robotické buňky EDUset ONE

Všechna zařízení jsou propojena přes switch, který se také nachází v technické skříni. Dále je součástí pracoviště stůl, který slouží jako odběrové místo pro úlohu pick&place, na níž bude návrh modulární knihovny pro řízení a sběr dat testován.

Obráběcí frézka byla vyvinuta primárně pro technické školy jako výukový prvek, který umožňuje studentům získat praxi v podmínkách co nejvíce se blížících reálné praxi [26]. V kombinaci s univerzálním kolaborativním robotem a PLC od významných světových výrobců robotů, resp. elektroniky to dělá z EDUsetu ONE vhodný edukační nástroj. Robotická buňka je taktéž vhodná pro výzkumné účely, protože umožňuje simulaci reálných průmyslových procesů v bezpečném a kontrolovaném prostředí. Řídicí jednotka PLC S7-1500 má vysoký výkon a je vhodná pro řízení komplexních a výpočtově náročných procesů. V případě EDUset ONE není maximálně využit její potenciál a stačilo by PLC nižší řady, tj. S7-1200.

2.4.1 Výběr komunikačního protokolu

Při výběru komunikačního protokolu pro řízení UR10e prostřednictvím PLC od Siemens hrála klíčovou roli možnost výměny dat v reálném časem a podpora protokolu oběma zařízeními. S odkazem na kapitolu 2.2 se množina průmyslových protokolů vhodných pro danou aplikaci zúžila pouze na protokoly založené na Ethernetu. Zvolen byl PROFINET z následujících důvodů:

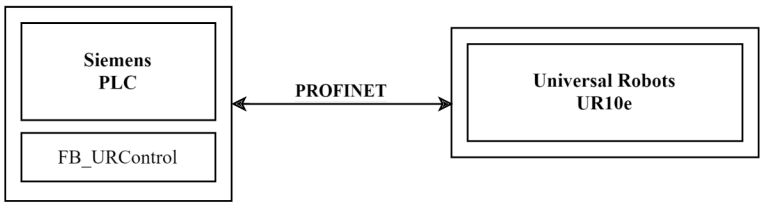
1) PLC SIMENS S7-1200/1500 umožňuje komunikaci přes PROFINET;
2) robot UR10e od Universal Robots umožňuje komunikaci přes PROFINET;
3) PROFINET je vhodný pro výměnu dat v reálném čase;
4) PROFINET je snadno konfigurovatelný, spolehlivý a zabezpečený;

5) PROFINET je v současné době nejrozšířenějším průmyslovým komunikačním protokolem založeným na Ethernetu.

Z popsaných protokoů v kapitole 2.2 nespňoval požadavky na komunikaci v reálném čase EtherNet/IP. Přestože protokoly EtherCAT a POWERLINK splňovaly požadovaná kritéria, PLC Siemens S7-1200/1500 nepodporuje jejich rozhraní [27].

3 KNIHOVNA PRO ŘÍZENÍ ROBOTA POMOCÍ PLC

Tato kapitola se věnuje první částí hlavní náplni práce, a sice tvorbě modulární knihovy pro řízení kolaborativního robota Universal Robots UR10e prostřednictvím PLC od společnosti Siemens, který spolu komunikují přes PROFINET, jak znázorňuje schéma na Obr. 7. Jednotlivé podkapitoly postupně popisují návrh modulární knihovny (3.1) včetně její implementace a vytvoření potřebného programu pro robota. Závěrečná podkapitola 3.2 se věnuje vytvoření konkrétní úlohy pro účely otestování řešení.



Obr. 7: Obecné schéma řízení robota pomocí PLC

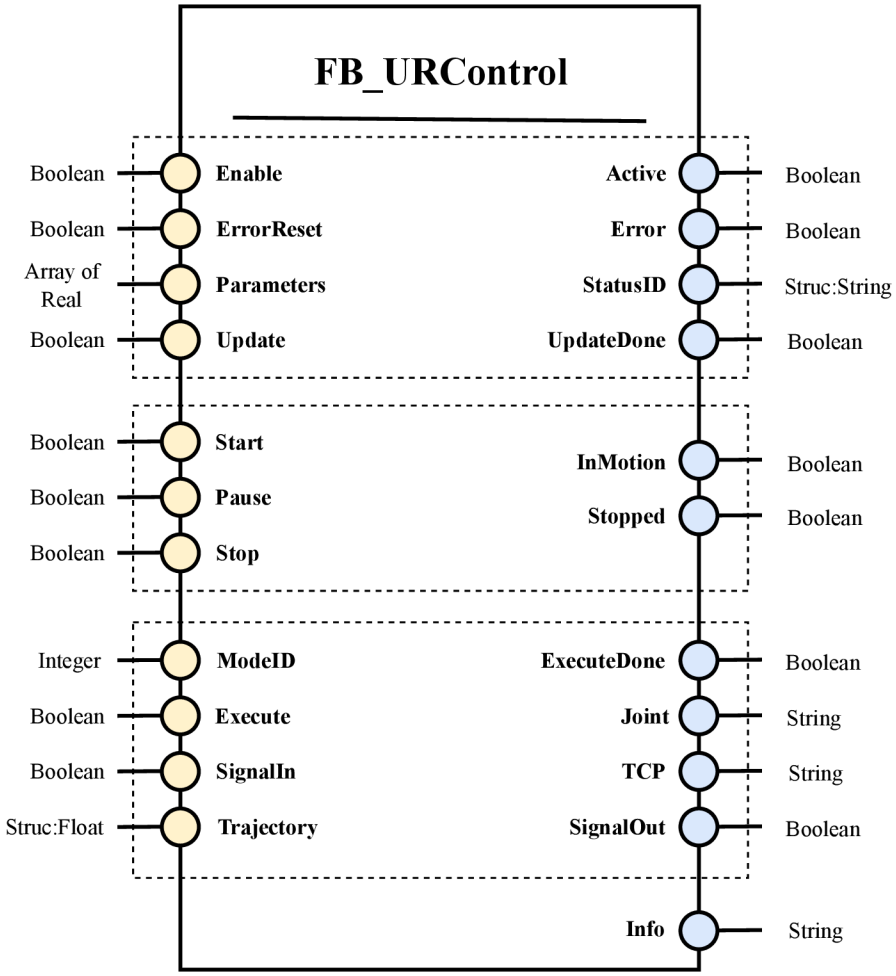
3.1 Návrh modulární knihovny

Cílem bylo vytvořit takovou knihovnu na řízení a sběr dat, která bude splňovat následující požadavky:

1) bude modulární, tzn. přenositelná mezi různými projekty v TIA Portalu;
2) zajistí řízení UR kolaborativního robota UR10e přostřednictvím Siemens PLC, konkrétně:
   - řízení trajektorie v kloubovém modu,
   - řízení trajektorie v TCP modu,
   - ovládání koncového zařízení na robotu (např. gripper, ejektor, atd.),
   - aktualizace parametrů jako je rychlost nebo zrychlení,
   - zapínání/vypínání/pozastavování programu běžícím na robotu;
3) zajistí sběr dat z UR kolaborativního robota UR10e.

Vzhledem k charakteristice vývoje softwaru pro Siemens PLC je zvolenou podobou knihovny funkční blok (FB). Výhodou této implementace je jednoduchá přenositelnost mezi projekty. Další významnou výhodou je skutečnost, že programátor při využívání tohoto FB nemusí znát jeho vnitřní logiku. Na Obr. 8 je znázorněn návrh FB. Na levé straně jsou vstupy do FB, které jsou koncipovány jako tzv. *settery*,

tedy funkce pro řízení robota. Na pravé straně FB jsou výstupy, které slouží ke sběru dat, tzv. *gettery*. Vlastnosti jednotlivých vstupů a výstupů, které jsou na Obr. 8 pro přehlednost organizovány do logických celků, jsou popsány v tabulce 2.



Obr. 8: Schéma funkčního bloku

Tab. 2: Vstupy a výstupy FB

Název	**Datový typ**	**Popis**
Enable	boolean	Vstup `True` aktivuje FB (používání knihovny), `False` deaktivuje FB.
ErrorReset	boolean	Vstup `True` je požadavek pro reset chybového stavu.
Parameters	array of reals	Nové hodnoty parametrů ([0] - rychlost, [1] - zrychlení).
Update	boolean	Příkaz na aktualizaci parametrů. Vstup `Paramaters` nesmí být prázný.
Start	boolean	Příkaz pro spuštění programu na robotu.
Pause	boolean	Příkaz pro pozastavení programu na robotu.
Stop	boolean	Příkaz pro zastavení programu na robotu.
ModeID	int	Definice typu pohybu, který má být vykonán. [0] - nedefinován, [1] - kloubový pohyb, [2] - TCP pohyb, [3] - signál pro koncové zařízení.
Execute	boolean	Příkaz pro vykonání příslušného pohybu dle `ModeID`.
SignalIn	boolean	Signál pro koncové zařízení.
Trajectory	array of reals	Body trajektorie ve formátu (x, y, z, rx, ry, rz, blend) nebo (j1, j2, j3, j4, j5, j6, blend).
Activate	boolean	Výstup indikující, zdali je (`True`) nebo není (`False`) FB aktivovaný.
Error	boolean	Informace o případné chybě (`True` - chyba je, `False` - chyba není).
StatusID	string	Charakteristika chybového stavu.
UpdateDone	boolean	Informace, zdali aktualizace parametrů proběla (`True`), či nikoli (`False`).
InMotion	boolean	Informace, zdali se robot hýbe, tj. vykonává příslušnou trajektorii (`True` - v pohybu).
Stopped	boolean	Informace, zdali je program na robotu pozastaven (`True`) nebo spuštěn (`False`).
Joint	string	Pozice kloubů.
TCP	string	Pozice TCP.
SignalOut	boolean	Signál z koncového zařízení.
ExecuteDone	boolean	Informace, zdali byla vykonána trajektorie, ev. akce koncového zařízení.
Info	string	Informace pro případnou diagnostiku.

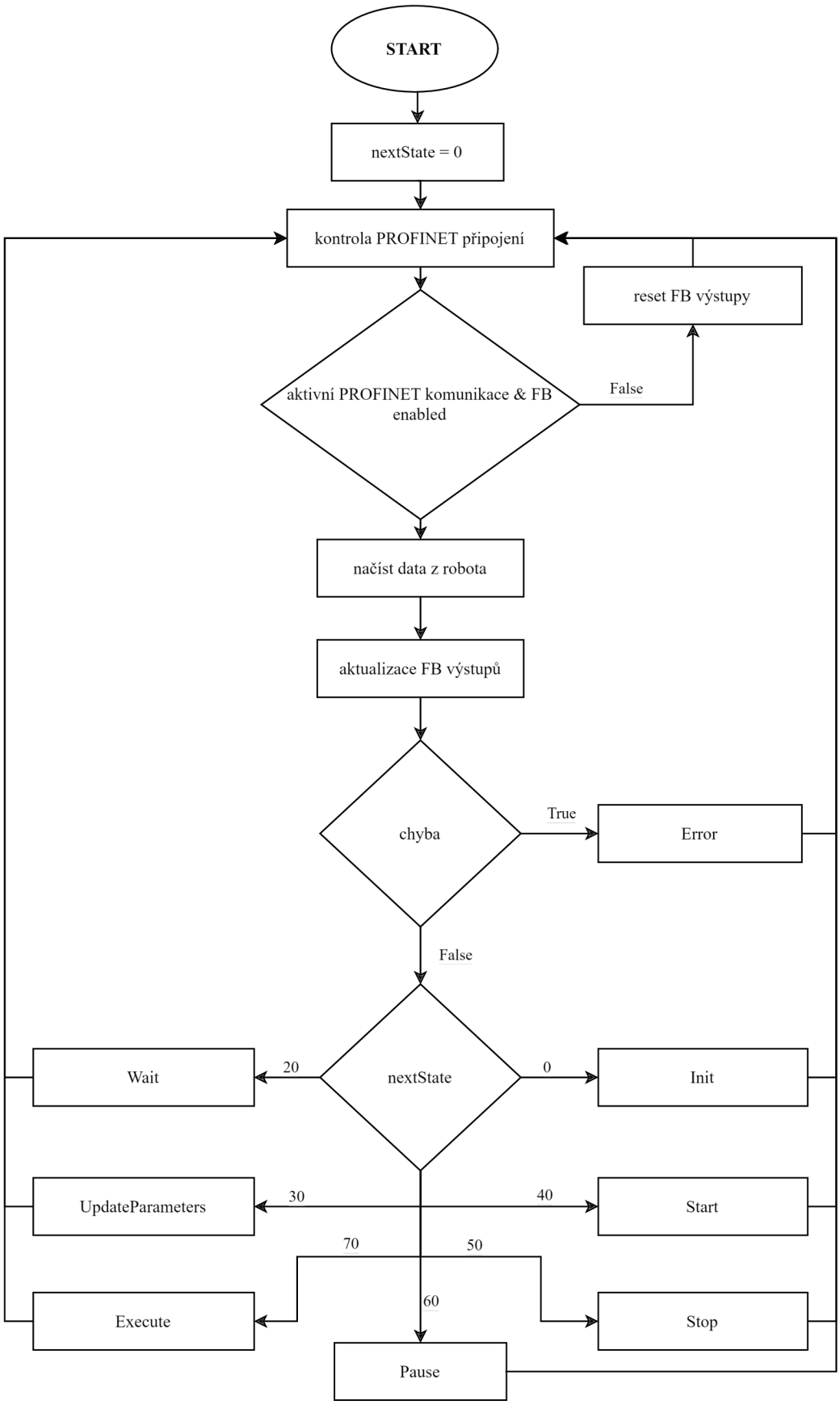
Vnitřní logika FB je popsána diagramem na Obr. 9. Architektura je koncipována jako stavový automat, který je řízen proměnnou `nextState`. Ta je při spuštění inicializována na nulu. Dále proběhne kontrola připojení přes PROFINET. Pokud je připojení aktivní a FB je aktivovaný, načtou se data z robota (veškerý seznam výstupně/vstupních dat z robota do PLC lze najít v příloze A) a zaktualizují výstupy funkčního bloku. Před určením nového stavu je potřeba zkontrolovat, zda se nikde nevyskytla žádná chyba. V případě, že byla chyba detekována, stavový automat se ustálý v chybovém stavu (`Error`). V tomto stavu je zablokované vykonání jakýchkoli jiných funkcí. Automat v tomto stavu zůstává, dokud se neodstraní chyba a nepřijde příkaz k resetu chyby (`ErrorReset`). Jestliže žádná chyba nebyla detekována, dojde k vyhodnocení, do jakého stavu se má automat dostat. Možností je sedm: `Init`, `Wait`, `Start`, `Stop`, `Pause`, `UpdateParameters` a `Execute`.

Stav `Init` je inicializační. Znamená to tedy, že stavový automat se do tohoto stavu dostane za normálních okolností jednou, a to při každém spuštění (ev. restartu) PLC. V daném okamžiku dojde k nastavení potřebých vnitřních proměnných a výstupů FB.

Stav `Wait` je stav, ve kterém PLC nevykonává v souvislosti s řízením robota žádné aktivity. V případě změn na vstupech FB dojde k vyhodnocení dalšího stavu.

Stavy `Start`, `Pause` nebo `Stop` dají příkaz robotu zapnout, pozastavit nebo vypnout program. Stav `UpdateParameters` řídí aktualizaci parametrů robota. V současnosti se jedná o parametry rychlost a zrychlení, nicméně kód je navržen tak, aby byl snadno rozšiřitelný pro přídání dalších parametrů.

A v neposlední řadě stav `Execute`. Tento stav řídí veškeré pohyby robota. Na základě hodnoty `modeID` na vstupu FB řídí pohyb robota podle předem definované trajektorie, a to v kloubovém, resp. TCP modu, nebo pošle signál (`True`/`False`), který ovládá koncového zařízení.



Obr. 9: Diagram funkčního bloku

3.1.1 Implementace knihovny

Funkční blok byl vyvíjen v TIA Portal V18, což je licencované vývojové přostředí pro Siemens PLC. Výsledkem je složka `UR_Control` v projektu `dp_nsterbova_1500.ap18`, který lze nalézt v příloze B. Složka představuje zdroj knihovny pro nový projekt. Na Obr. 10 je zestručněná struktura projektu, ve které jsou ponechány pouze relevantní uzly v souvistlosti s použitím zmiňované knihovny.

`URControl (FB)` je samotná implementace funknčího bloku z Obr. 8. Všechny funkce, které funkční blok používá, jsou uloženy v jedné složce s názvem `Functions`. Globální proměnné jsou definované v datovém bloku (DB) `Global_var`. Pro komunikaci s robotem byly v rámci tohoto konkrétního projektu v souboru `UR_tags` zabrány některé vstupní a výstupní adresy PLC, ty jsou sepsány v tabulkách 3 a 4. V případě využití knihovny v novém projektu je nutné dle individuálních potřeb adresy přepsat. Z robota je možné získat více informací. Kompletní seznam je uveden v příloze A. Pro práci s těmito daty jsou připravené DB ve složce `UR_variables`.

Popsaný projekt je možné využít jako šablonu pro tvorbu nového projektu či jako zdroj FB a souvisejících souborů pro přenesení do jiného projektu v rámci TIA Portal. Ve druhém zmíněném případě stačí přenést do nového projektu celou složku `UR_Control` a naadresovat plc tagy dle `UR_tags`.

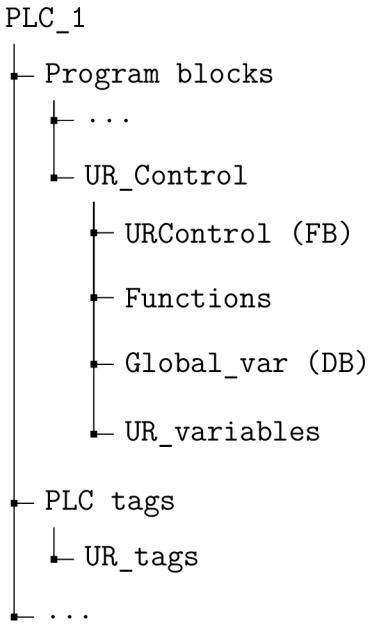
Při implemetaci stavu `Execute` se vycházelo z myšlenky poslat robotu všechny body trajektorie najednou. Toto řešení by zajistilo větší plynulost pohybu. Vyžadovalo by to ovšem, aby UR Scripting Language, v němž je psán program pro UR robota, podporoval datovou strukturu, do které je možné uložit více vektorů. Taková datová struktura v jazyce chybí, proto se přistoupilo k řešení posílat body zvlášť a vykonávat pohyb jednotlivě.

Tab. 3: Vstupy do PLC/Výstupy z UR

Název	**Datový typ**	**Adresa**
updateDone	boolean	%I34.1
trajectoryDone	boolean	%I34.2
signalOut	boolean	%I34.3
inMotion	boolean	%I34.5
urErrorType	DInt	%ID316

Tab. 4: Výstupy z PLC/Vstupy do UR

Název	**Datový typ**	**Adresa**
signalIN	boolean	%Q26.0
execute	boolean	%Q26.1
update	boolean	%Q26.2
urErrorReset	boolean	%Q26.4
x/j1	real	%QD78
y/j2	real	%QD82
z/j3	real	%QD86
rx/j4	real	%QD90
ry/j5	real	%QD94
rz/j6	real	%QD98
blend	real	%QD102
speed	real	%QD106
acceleration	real	%QD110
modeID	DInt	%QD30



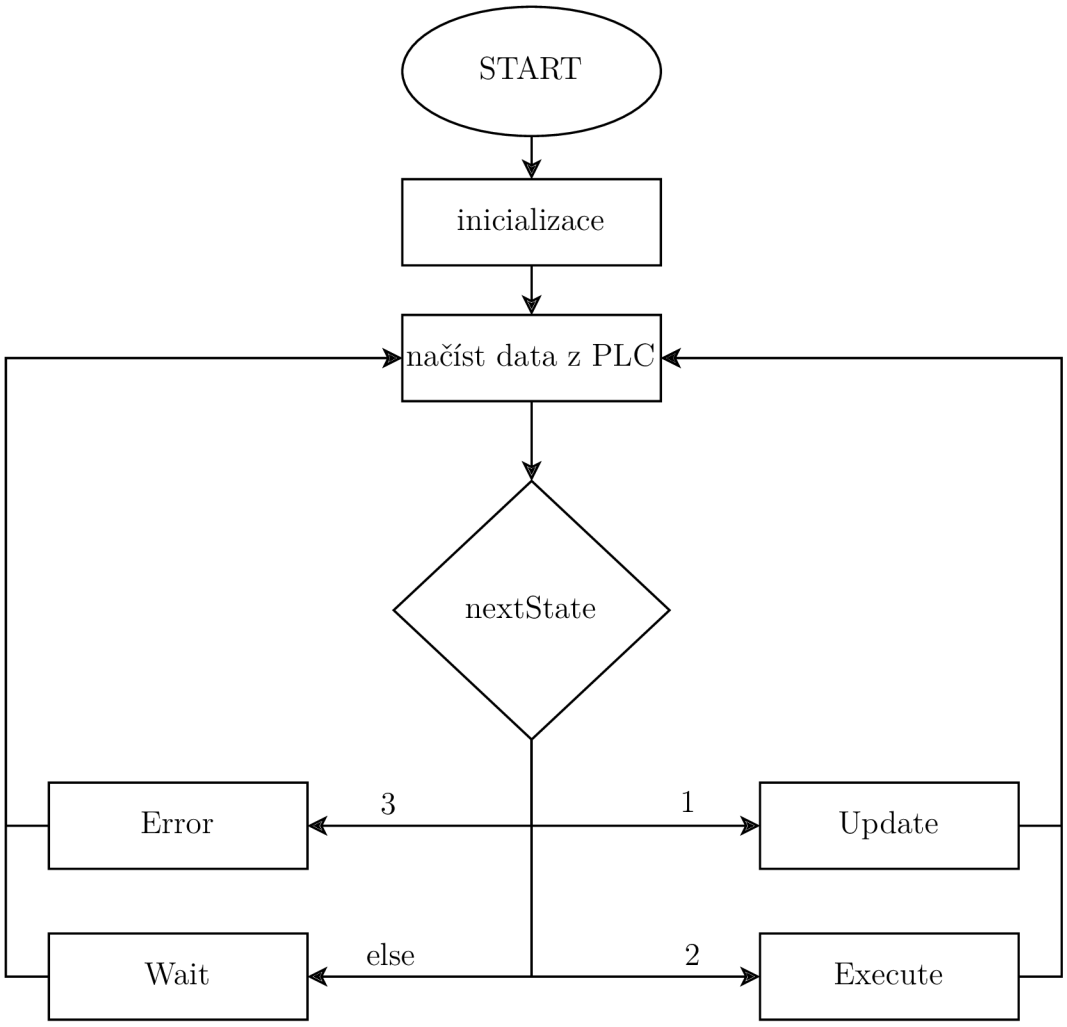
Obr. 10: Struktura projektu `dp_nsterbova_1500.ap18`

3.1.2 UR script

V druhém kroku bylo zapotřebí vytvořit univerzální program, díky kterému bude robot schopen reagovat na přicházející signály z PLC. Za tímto účelem byly vytvořeny

dva skripty v UR Scripting Language sloučené do programu `dp_nsterbova.urp`. Všechny tři zmíněné soubory jsou v příloze C.

Logika programu je nasníněna v diagramu na Obr. 11. První krok pokrývá skript `ur_init.script`, ve kterém je část kódu, který se spustí pouze při prvním spušetění programu na robotu. Jedná se o sérii pokynů, kdy dojde k deklaraci a inicializaci několika globálních proměnných. Zbývající kroky zajišťuje druhý skript `ur_main.script`, který začne načtením potřebných vstupů do robota (tj. výstupů z PLC). Stejně jako vnitřek FB je i hlavní část programu robota napsána formou stavového automatu. Robot má čtyři stavy. V případě, že se FB dostane do chybového stavu a čeká na reset, robot se též dostane to stavu `Error`, ve kterém čeká na odstranění chyby. Zbylé stavy `Update`, `Execute` a `Wait` reagují na své protějšky ve FB. Tzn. ve stavu `Update` dojde k aktualizaci rychlosti a zrychlení. Stav `Execute` vykoná příslušné pohyby, ať už se jedná o pohyb v kloubovém/TCP modu nebo o práci s koncovým zařízením. Ve stavu `Wait` čeká na příkazy z PLC.



Obr. 11: Diagram UR programu `dp_nsterbova.urp`

3.2 Testování

Po domluvě s firmou Intemac Solutions, s.r.o. byla za účelem otestování modulární knihovny pro řízení kolaborativního robota prostřednictvím PLC vytvořena *pick&place* úloha využívající zmiňovaný FB blok z kapitoly 3.1, která je taktéž součástí projektu `dp_nsterbova_1500.ap18`.

Úkolem robota je zvednout neopracovanou kostku ze stolu (ten může simulovat například běžicí pás ve výrobě), vložit ji do frézky a následně se vrátit do výchozí pozice nad stůl, kdy je robot připraven pro zopakování téhož úkonu.

Trajektorie robota se opírá o několik referenčních bodů popsaných v tabulce 5, přičemž jeden cyklus úlohy, tj. sebrání kostky, založení do frézky a vrácení se zpět do výchozí pozice, je rozdělen do několika částí:

1) **1. Fáze:** robot jede s otevřeným gripperem z výchozího bodu *cube_home* přes *cube_offset* na souřadnice *cube*;
2) **2. Fáze:** gripper se zavírá, čímž sevře kostku, s níž se dále pohybuje;
3) **3. Fáze:** robot najede do pozice *cube_offset*;
4) **4. Fáze:** robot se pohybuje z pozice *cube_offset* do pozice *cnc_offset*;
5) **5. Fáze:** robot najede do pozice *cnc*;
6) **6. Fáze:** gripper se otevírá, čímž pouší kostku;
7) **7. Fáze:** robot najede zpátky do pozice *cnc_offset*;
8) **8. Fáze:** v závěrečné fázi se vrací robot z pozice *cnc_offset* do výchozí pozice pro nový cyklus *cube_home*;

Tab. 5: Průjezdové body úlohy *pick&place*

Název bodu	**Souřadnice**
cube	x = -0.00091, y = 0.75915, z = 0.03997
cube_offset	x = -0.00091, y = 0.75915, z = 0.06853
cube_home	x = -0.00091, y = 0.75915, z = 0.16211
mid_point	x = 0.4498, y = 0.18753, z = 0.22953
cnc_home	x = 0.00247, y = -0.66235, z = 0.45232
cnc_offset	x = 0.002067, y = -1.14277, z = 0.45232
cnc	x = 0.02066, y = -1.14277, z = 0.41982

Při definování trajektorie a sesbírání průjezdových bodů se využilo možnosti vést robota ručně. Zásahem člověka se tam však zanesly nepřesnosti, čemuž se následně věnuje kapitola 5.

Testování vytvořeného řešení modulární knihovny proběhlo ve dvou formách. Zaprvé, v interakci s reálnou buňkou EDUset ONE, kde bylo použito PLC S7-1500.

Implementovaná knihovna se testovala také v prostředí simulace, čemuž se věnuje podkapitola 3.2.2.

3.2.1 Testování s reálnými zařízeními

K vývoji i testování řešení byl k dispozici kolaborativní robot Universal Robots UR10e a PLC Siemens. V první fázi vývoje se použilo PLC S7-1200 model CPU 1214 DC/DC/DC, které bylo později nahrazeno PLC S7-1500 model CPU 1515F-2-PN. Druhé zmíněné PLC bylo použito i v případě testování *pick&place* úlohy.

Na PLC byl nahrán projekt `dp_nsterbova_1500.ap18` a robot spouštěl program `dp_nsterbova.urp` obsahující skripty `ur_init.script` a `ur_main.script`.

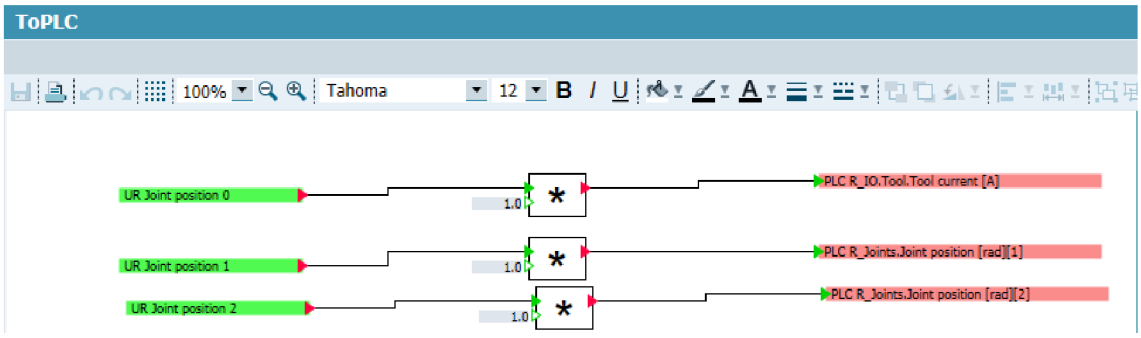
Robot vykonal úlohu, která je popsána v této práci výše, dle očekávání a bez zjevných komplikací. Video zachycující jeden cyklus je v příloze D.

3.2.2 Testování v simulaci

K realizaci testování v simulaci bylo zapotřebí následujících programů:

1) **TIA Portal** - licencované vývojové prostředí pro zařízení od firmy Siemens;
2) **URSim** - simulace UR robota;
3) **PLCSim Advanced** - licencovaná simulace Siemens PLC;
4) **SIMIT** - licencovaná simulační platforma od společnosti Siemens.

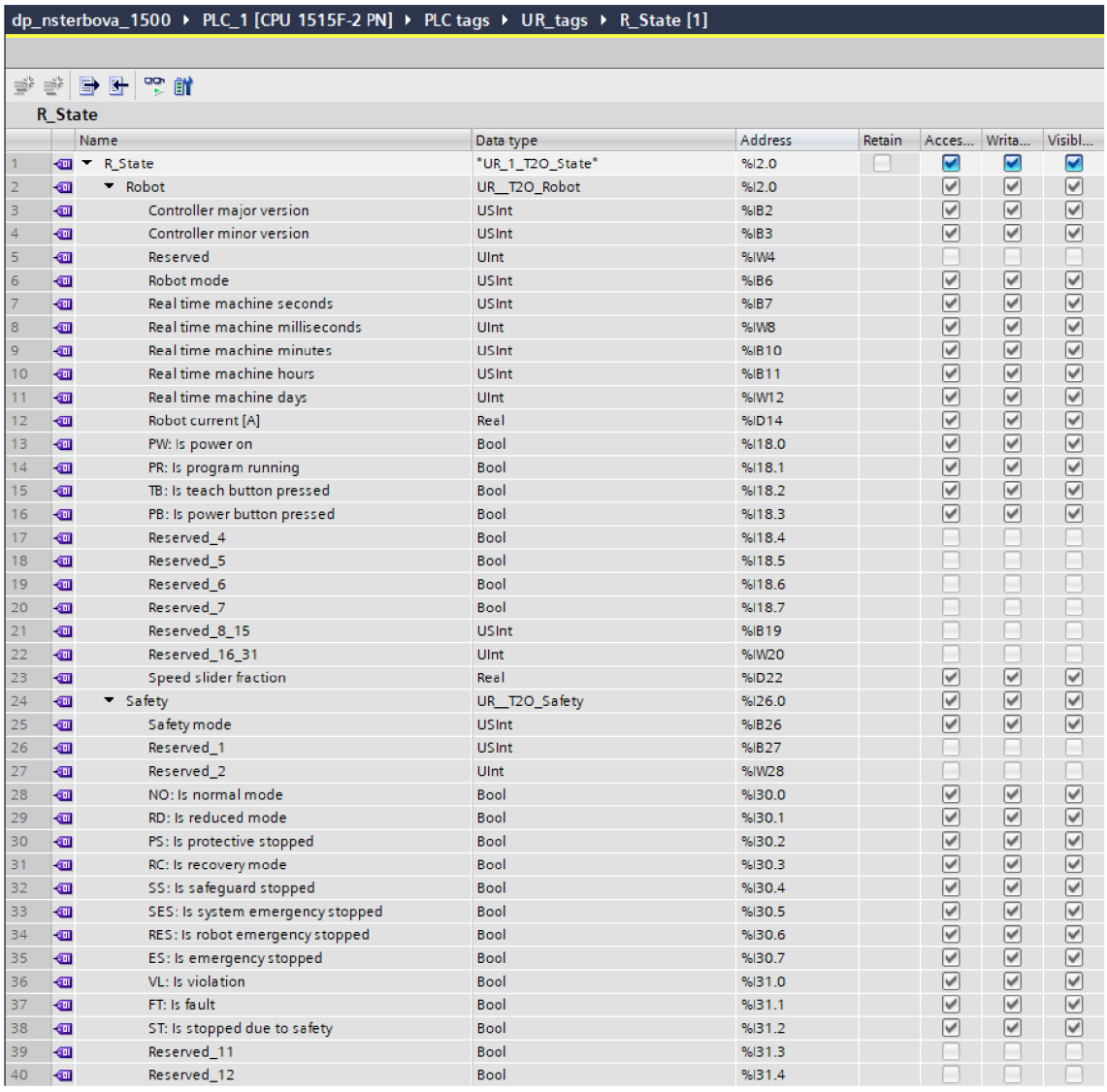
SIMIT je prostředí, které dokáže simulovat přenos signálů přes PROFINET při komunikaci mezi simulací robota a simulací PLC. Klíčovým krokem je manuální napojení výstupů z robota na vstupy do PLC a obráceně, jak je tomu například na Obr. 12.



Obr. 12: Propojení signálů v SIMIT prostředí

K tomu je zapotřebí mít v SIMITu přidané moduly jednotlivých zařízení, které obsahují popsané vstupní a výstupní signály. Modul pro PLC je v SIMITu implicitně, jelikož se jedná o zařízení přímo do Simensu. V případě UR robota je nutné si o modul zažádat v rámci licence. K dispozici byly pouze licence TIA Portalu a SIMITu, které nebyly kompatibilní. Konkrétním příkladem je práce se signály,

které nesou informace o stavu robota. Na Obr. 13 jsou naznačeny všechny očekávané signály z pohledu PLC, zatímco Obr. 14 zobrazuje signály z robota ve formě, v jaké je dokáže prostředí SIMIT interpretovat a předat dál. Z obrázků je očividné, že si moduly neodpovídají. Toto chování je pravděpodobně způsobeno zastaralejší verzí modulu UR robota v porovonání s novější verzí vývojového prostředí TIA Portal. Z tohoto důvodu nebylo možné testování řešení v simulaci realizovat.

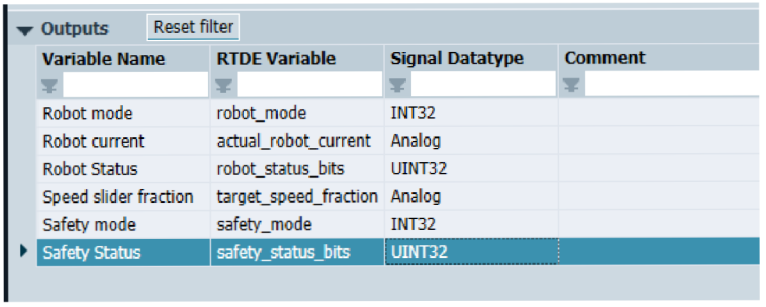


dp_nsterbova_1500 ▸ PLC_1 [CPU 1515F-2 PN] ▸ PLC tags ▸ UR_tags ▸ R_State [1]

R_State

	Name	Data type	Address	Retain	Acces...	Writa...	Visibl...
1	R_State	"UR_1_T2O_State"	%I2.0	☐	☑	☑	☑
2	Robot	UR__T2O_Robot	%I2.0		☑	☑	☑
3	Controller major version	USInt	%IB2		☑	☑	☑
4	Controller minor version	USInt	%IB3		☑	☑	☑
5	Reserved	UInt	%IW4		☐	☐	☐
6	Robot mode	USInt	%IB6		☑	☑	☑
7	Real time machine seconds	USInt	%IB7		☑	☑	☑
8	Real time machine milliseconds	UInt	%IW8		☑	☑	☑
9	Real time machine minutes	USInt	%IB10		☑	☑	☑
10	Real time machine hours	USInt	%IB11		☑	☑	☑
11	Real time machine days	UInt	%IW12		☑	☑	☑
12	Robot current [A]	Real	%ID14		☑	☑	☑
13	PW: Is power on	Bool	%I18.0		☑	☑	☑
14	PR: Is program running	Bool	%I18.1		☑	☑	☑
15	TB: Is teach button pressed	Bool	%I18.2		☑	☑	☑
16	PB: Is power button pressed	Bool	%I18.3		☑	☑	☑
17	Reserved_4	Bool	%I18.4		☐	☐	☐
18	Reserved_5	Bool	%I18.5		☐	☐	☐
19	Reserved_6	Bool	%I18.6		☐	☐	☐
20	Reserved_7	Bool	%I18.7		☐	☐	☐
21	Reserved_8_15	USInt	%IB19		☐	☐	☐
22	Reserved_16_31	UInt	%IW20		☐	☐	☐
23	Speed slider fraction	Real	%ID22		☑	☑	☑
24	Safety	UR__T2O_Safety	%I26.0		☑	☑	☑
25	Safety mode	USInt	%IB26		☑	☑	☑
26	Reserved_1	USInt	%IB27		☐	☐	☐
27	Reserved_2	UInt	%IW28		☐	☐	☐
28	NO: Is normal mode	Bool	%I30.0		☑	☑	☑
29	RD: Is reduced mode	Bool	%I30.1		☑	☑	☑
30	PS: Is protective stopped	Bool	%I30.2		☑	☑	☑
31	RC: Is recovery mode	Bool	%I30.3		☑	☑	☑
32	SS: Is safeguard stopped	Bool	%I30.4		☑	☑	☑
33	SES: Is system emergency stopped	Bool	%I30.5		☑	☑	☑
34	RES: Is robot emergency stopped	Bool	%I30.6		☑	☑	☑
35	ES: Is emergency stopped	Bool	%I30.7		☑	☑	☑
36	VL: Is violation	Bool	%I31.0		☑	☑	☑
37	FT: Is fault	Bool	%I31.1		☑	☑	☑
38	ST: Is stopped due to safety	Bool	%I31.2		☑	☑	☑
39	Reserved_11	Bool	%I31.3		☐	☐	☐
40	Reserved_12	Bool	%I31.4		☐	☐	☐

Obr. 13: UR tagy v TIA Portal



Outputs | Reset filter

Variable Name	RTDE Variable	Signal Datatype	Comment
Robot mode	robot_mode	INT32	
Robot current	actual_robot_current	Analog	
Robot Status	robot_status_bits	UINT32	
Speed slider fraction	target_speed_fraction	Analog	
Safety mode	safety_mode	INT32	
Safety Status	safety_status_bits	UINT32	

Obr. 14: UR modul v SIMIT prostředí

4 METODY PRO OPTIMALIZACI KOLABORATIVNÍCH ROBOTŮ

Roboty jsou v současné době klíčovým prvkem automatizovaných procesů v průmyslu, což přikládá význam optimalizaci, která by zvýšila efektivitu nebo by snížila s tím spojené náklady [28].

Optimalizace trajektorie je přístup používaný k minimalizaci (ev. maximalizaci) nejrůznějších (multi)kriteriálních funkcí jako například spotřeba energie, délka dráhy, čas a jiné [28].

Minimalizace spotřeby energie je velmi komplexní problém, ke kterému lze přistupovat z různých stran. V současnosti jsou optimalizační metody rozdělené do tří základních kategorií - návrh vhodné trajektorie, návrh provozních parametrů nebo návrh vhodného rozvrhu a plánování provozu [29]. Tato práce se omezí pouze na první zmíněnou kategorii.

Následující podkapitoly popisují optimalizační metody bez gradientů, které byly dále použity v práci (viz kapitola 5).

4.1 NGOpt16

Úlohy, které vyžadují k ohodnocení simulaci nebo živý experiment, bývají zpravidla časově náročné. Pro jejich řešení jsou vhodné black-box optimalizační algoritmy. Jedná se o automatizovaný nástroj, který dle základních informací o úloze a výpočetních zdrojích vybere vhodný algoritmus, resp. algoritmy [30].

Článek [30] představuje black-box optimalizátor NGOpt, který je integrovaný v rámci Python knihovny *Nevergrad* spravovanou společností Meta [31]. V následujícím výčtu je přehled algoritmů, se kterými NGOpt operuje. Rozdělené jsou podle základních charakteristik úloh:

1) **Pouze diskrétní rozhodovací proměnné s nulovým šumem:**
   - genetický algoritmus s bandit algoritmem;
   - 1Plus1 evoluční algoritmus s lineárně se zmenšujícím krokem;
   - adaptivní 1Plus1 algoritmus;
   - CMandAS2;
   - FastGA.
2) **Pouze numerické rozhodovací proměnné s nenulovým šumem:**
   - progresivní optimalizace;
   - Test-based populatio adoption (TBPSA);
   - sekvenční kvadratické programování.

3) **Pouze numerické rozhodovací proměnné s nulovým šumem a vysokým stupně paralelizace:**
   - MetaTuneRecentering;
   - DiagonalCMA-ES;
   - NativeTBPSA.
4) **Pouze numerické rozhodovací proměnné s nulovým šumem a sekvenčním ohodnocením:**
   - CMA-ES;
   - Powell;
   - 1Plus1 evoluční strategie s pravidlem 1/5;
   - Cobyla [30].

Z [32] vyplývá, že NGOpt je vhodným nástrojem pro optimalizaci časově náročných úloh velkých dimenzí. Proto bude uvažován pro minimalizaci spotřeby energie v kapitole 5, přičemž z podrobnějšího rozboru v [30] se dá očekávat, že použitým algoritmem bude Cobyla.

4.2 CMA-ES

Algoritmus adaptace kovarianční matice (Covariance Matrix Adaptation Evolution Strategy) je stochastická metoda vhodná pro optimalizaci reálných parametrů nelineárních, nekonvexních úloh [33].

Průběh algoritmu se dá rozdělit do tří fází:

1) vzorkování pro vytvoření nového řešení;
2) výpočet hodnoty kriteriální funkce;
3) aktualizace distribucí parametrů m, σ a C.

V každé iteraci jsou řešení generována následovně

$$\mathbf{x}_{t+1} = \mathbf{m}_t + \sigma_t \mathbf{y}_i, \quad \mathbf{y}_i \sim N(\mathbf{0}, \mathbf{C}_t), \tag{1}$$

kde σ_t je *scalling* faktor, y_i je směr prohledávání, $\mathbf{m}$ je průměr a $\mathbf{C}$ je kovarianční matice.

Pro všechna nově vygenerovaná řešení se spočítá hodnota kriteriální funkce $f(x_i)$ a seřadí se podle velikosti vzestupně. Pro selekci jsou použita řešení z první poloviny řady. μ vybraných řešení následně slouží k aktualizaci parametrů m, σ a C.

Průměr distribuce je vážený odhad maximální věrohodnosti (MLE) vybraných řešení μ.

CMA-ES standardně používá kumulativní úpravu délky kroku. Jeho principem je, že směr následného hledání by měl být konjugovaný. V případě pozitivní

korelace mezi následujícími směry hledání, je velikost kroku příliš malá a měla by se zvětšit. V opačném případě je velikost kroku příliš velká a měla by se snížit.

Aktualizace kovarianční matice je provedena ve dvou částech, tzv. *rank-1* aktualizace a *rank-σ* aktualizace. *Rank-1* aktualizace zohledňuje přímou cestu evoluce a je považována za úspěšný směr hledání. Původní CMA-ES obsahuje pouze tuto aktualizaci a vykazuje dobré výsledky při malých populacích. *Rank-σ* aktualizace je vážený odhad maximální věrohodnosti (MLE) pomocí vybraných σ řešení s odstraněním parametru délky kroku. Tato aktualizace hraje důležitější roli, pokud je velikost populace větší [34].

4.3 Diferenciální evoluce

Diferenciální evoluce (DE) je považována za dobrou obecnou metodu pro optimalizaci náročných úloh. Jedná se o jeden z evolučních algoritmů, přičemž v porovnání s jinými algoritmy tohoto druhu je DE poměrně efektivní a jednoduchý na implementaci [35]. Algoritmus DE se skládá ze čtyř fází a je popsán pseudokódem v Algoritmu 1 [36].

Algoritmus 1 Diferenciální evoluce

1: vygenerování počáteční populace velikosti N
2: **while** ukončovací podmínka není splněna **do**
3: **for** každé n v P **do**
4: vygenerování náhodných čísel $r_1 \neq r_2 \neq r3 \neq n$
5: **for** každé n v P **do**
6:

$$X'_{i,n} = \begin{cases} X^G_{i,r1} + F \cdot (X^G_{i,r2} - X^G_{i,r3}) & \text{pokud } rand_n \leq Cr \\ X_{i,n} & \text{jinak} \end{cases}$$

7: **end for**
8: **if** X'_n je lepší než X_n **then**
9: náhrada X'_n za X_n
10: **end if**
11: **end for**
12: **end while**

První fáze je **inicializace**, kdy dojde náhodně k vytvoření populace P^G. Jedná o n-tici D-rozměrných vektorů pro generaci G

$$X^G_n = x^G_{1,n}, x^G_{2,n}, ..., x^G_{D,n}. \tag{2}$$

Vektor X_n^G je generován pomocí *rand(0,1)*

$$X_n^G = X_{lb} + (X_{ub} - X_{lb}) \cdot rand(0,1), \tag{3}$$

kde X_{lb}, X_{ub} jsou dolní a horní meze prohledávaného prostoru.

Ve fázi **mutace** se pro každý vektor X_n^G vygeneruje mutační vektor

$$V_n^G = X_{r1}^G + F \cdot (X_{r2}^G - X_{r3}^G), \tag{4}$$

kde F je *scaling* faktor, jehož hodnota je v rozmezí 0 až 1. Vektory $X_{r1}^G, X_{r2}^G, X_{r3}^G$ jsou navzájem rozdílné a náhodně vybrané.

Třetí fází je **křížení**, kdy vzniká tzv. *trial* vektor U_n^G. Křížení probíhá mezi původním vektorem X_n^G a mutačním vektorem V_n^G s použitím pravděpodobnosti křížení *Cr*, která nabývá hodnot mezi 0 a 1. Vektor křížení vzniká jako

$$u_{i,n}^G = \begin{cases} v_{i,n}^G & \text{pokud } rand_n \leq Cr \\ x_{i,n}^G & \text{jinak} \end{cases} \tag{5}$$

Ve čtvrté fázi, kterou je **selekce**, se porovnává původní vektor s vektorem křížení. Ten s lepší *fitness* hodnotou přežije do další generace. Fáze mutace, křížení a selekce se opakují do té doby, dokud není splněna ukončovací podmínka [36].

4.4 cGA

[37] Kompaktní genetický algoritmus (cGA) vychází z klasického genetického algoritmu (GA), který mírně modifikuje. Na rozdíl od GA zpracovává každý gen nezávisle, čímž snižuje výpočetní náročnost. Díky diskrétní reprezentaci populace pomocí pravděpodobnostního rozdělení dosahuje cGA nižších paměťových nároků ve srovnání s klasickým GA. Zároveň je navržen tak, aby napodoboval chování klasického GA s lineární závislostí. Algoritmus 2 popisuje pseudokód cGA.

Během selekce jsou upřednostňováni jedinci s vyšší fitness, avšak tato selekce probíhá na úrovni celých jedinců, nikoliv na úrovni jednotlivých genů.

Na rozdíl od klasického GA, který vyžaduje ukládání n bitů pro každou pozici genu, krok aktualizace cGA má konstantní velikost *1/n*. cGA uchovává pouze podíl jedniček a nul jakožto konečnou sadu *n+1* čísel *(0, 1/n, 2/n, ..., n/n)*. Toto rozdělení lze uložit s pomocí $log_2(n+1)$ bitů.

Křížení v cGA slouží ke kombinování prvků z fitness řešení. Opakované použití křížení vede k dekorelaci genů v populaci, čímž se populace stává kompaktnější a umožňuje efektivnější reprezentaci pomocí pravděpodobnostního vektoru [37].

Algoritmus 2 Kompaktní genetický algoritmus

```
    // inicializace pravděpodobnostního vektoru
 1: for i in range (1,l) do
 2:     p[i] = 0.5
 3: end for
    // vygenerování dvou jedinců z vektoru p
 4: a = generate(p)
 5: b = generate(p)
    soutěžení
 6: winner, loser = compete(a,b)
    // aktualizace pravděpodobnostního vektoru
 7: for i in range (1,l) do
 8:     if winner[i] ≠ loser[i] then
 9:         if winner[i] == 1 then
10:             p[i] = p[i] + 1/n
11:         else
12:             p[i] = p[i] - 1/n
13:         end if
14:     end if
15: end for
    // kontrola, jestli vektor konvergoval
16: for i in range (1,l) do
17:     if p[i]>0 and p[i] <1 then
18:         return to step 4
19:     end if
20: end for
21: return p
```

4.5 Cobyla

[38] COBYLA (Constrained Optimization BY Linear Approximation) je deterministický optimalizační algoritmus bez derivací, který k optimalizaci používá lineární interpolační modely cílové funkce a omezujících funkcí. Tato metoda může být vysoce efektivní, zejména pro problémy, kde je cílová funkce a její omezení obtížné řešit přímo.

Aproximace je dosažena procesem interpolace na *n+1* vrcholech simplexu, kde n představuje počet proměnných optimalizační úlohy.

Algoritmus iterativně hledá optimální řešení nad určitou oblastí (*region of trust*). Když najde lepší řešení, než některé ze stávajících, to staré je nahrazeno no-

vým. Pokud žádné lepší řešení není nalezeno, prohledávací prostor se zmenší. Tento proces pokračuje, dokud není dosaženo bodu, kde nelze provést další zlepšení [38].

4.6 SPSA

Simultánní stochastická aproximace s perturbací (Simultaneous Perturbation Stochastic Approximation, SPSA) je algoritmus se schopností efektivně odhadnou gradient funkce pomocí dvou funkčních volání. Proto je vhodný pro optimalizační úlohy velkého rozsahu, u nichž není možné počítat gradient. Navíc je poměrně robustní vůči šumu.

Algoritmus SPSA iterativně hledá řešení rovnice

$$\mathbf{x}_{k+1} = \mathbf{x}_k - a_k \hat{g}_k(\mathbf{x}_k), \tag{6}$$

kde $\hat{g}_k(\mathbf{x}_k)$ je stochastická aproximace $\nabla f(\mathbf{x})$ a a_k je klesající váhový koeficient. Pokud odhad gradientu nahradíme skutečným gradientem, odpovídá vzorec 6 metodě nejstrmějšího poklesu.

Běžnou aproximací gradientu je centrální diference, která však není vhodná pro úlohy vyšších dimenzí. Proto SPSA to řeší implementací stochastické gradientní aproximace definované následovně:

$$\hat{g}_k(\mathbf{x}_k) = \frac{y(\mathbf{x}_k + c_k\Delta_k) - y(\mathbf{x}_k - c_k\Delta_k)}{2c_k}, \tag{7}$$

kde Δ_k je náhodný perturbační vektor $\Delta_k = [\Delta_{k1}, \Delta_{k2}, ..., \Delta_{kp}]^T$, c_k je nějaké malé pozitivní číslo [39].

Algoritmus 3 popisuje pseudokód SPSA algoritmu [40].

Algoritmus 3 SPSA

1: inicializace a výběr koeficientů
2: vygenerování simultánního vektoru perturbace
3: ohodnocení *fitness* funkce
4: gradientní aproximace
5: aktualizace odhadu Θ
6: krok 1 **OR** ukončit

5 STATICKÁ OPTIMALIZACE ROBOTA

Tato kapitola se věnuje druhé části závěrečné práce a sice statické optimalizaci robota UR10e. Vrámci kapitoly je popsán návrh programu s ohledem jak na minimalizaci spotřeby energie, tak na minimalizaci dráhy. A dále se kapitola věnuje samotné implementaci a zpracování výsledků optimalizace.

5.1 Návrh programu

Cílem bylo napsat takový program, který optimalizuje dráhu robota vzhledem a) ke spotřebě energie, b) k délce dráhy. Jako programovací jazyk byl zvolen Python, který disponuje širokou škálou knihoven vhodných pro optimalizaci. Vzhledem k charakteristice úlohy byla vybrána knihovna *Nevergrad*, která se zaměřuje na algoritmy bez gradientů. Hlavní strukturu programu popisuje Algoritmus 4.

Algoritmus 4 Program pro statickou optimalizaci

```
1: procedure MAIN
2:     import knihoven
3:     deklarace globálních proměnných
4:     parametrizace & výběr optimalizátoru
5:     minimalizace kriteriální funkce
6:     uložení výsledků
7:     kontrola proveditelnosti výsledného řešení
8:     vykreslení grafu konvergence kriteriální funkce & grafu výsledné trajektorie
9: end procedure
```

Pro hledání optimální trajektorie robota existují dva přístupy. První možností bylo nadefinovat souřadnice počátečního a koncového bodu dráhy a bez jakýchkoli omezení nechat vybraný algoritmus najít vhodnou trajektorii. Z Obr. 6 je patrné, že frézka a stůl pro odebrání kostky se nacházejí v jedné linii, na které je mezi nimi umístěn samotný robot. V případě minimalizace dráhy by se tak dalo očekávat, že nejkratší cestou bude přímka od stolu k frézce a zpět. Takovéto řešení ovšem není proveditelné z důvodu kolize robota.

Proto byla zvolena druhá možnost, která spočívala v ručním definování bezpečné trajektorie robota (viz kapitola 3.2) a následném hledání optimálních odchylek jednotlivých bodů na trase. Toto řešení je bezpečnější, jelikož kolizím se dá předcházet vhodnou úpravou maximálních rozměrů odchylek.

V rámci definování parametrů optimalizace se vytvořil vektor o velikosti počtu bodů trajektorie krát šest (jeden bod je reprezentován šesti čísly typu double, což odpovídá formátu x, y, z, rx, ry, rz). Uvažovalo se také pole o velikosti *počet*

bodů X 6, nicméně v tomto formátu zvolený optimalizátor nefungoval korektně. Kriteriální funkce je tomu uzpůsobena. Jejím vstupním parametrem je vektor, který se hned zpočátku převádí na pole o velikosti *počet bodů X 6*, s nímž dále pracuje.

Dalšími dvěma parametry jsou spodní a horní hranice, které definují interval, do něhož musí spadat nalezené odchylky. Optimalizační algoritmus v základu pracuje s intervalem od -10 do 10.Vzhledem k tomu, že pro každou souřadnici mohou platit jiná omezení, je původní interval dle vztahu 8 transformován do požadovaného rozsahu.

$$transformed\_value = lb\_new + \left((value - lb) \cdot \frac{(ub\_new - lb\_new)}{(ub - lb)} \right), \qquad (8)$$

kde *lb* a *ub* je původní dolní, resp. horní mez, *lb_new* a *ub_new* jsou nové meze, *value* je vstupní hodnota a *transformed_value* je výstupní hodnota transformace.

Výstupem programu jsou čtyři soubory. Prvním z nich je textový soubor `results.txt`, který obsahuje nalezené optimální odchylky. Dále soubor `optim.csv`, kde je uložena výsledná trajektorie. A v neposlední řadě dva grafy - jeden zobrazující konvergenci kriteriální funkce v průběhu celého běhu a druhý zobrazující nalezenou optimální trajektorii.

5.2 Implementace programu

Pro práci s daty se využívaly Python knihovny `numpy`, `pandas`, `matplotlib` a `csv`. Pro řízení robota v reálném čase se využily knihovny `rtde_control` a `rtde_receive`. Dále bylo nutné importovat optimalizační knihovnu `nevergrad` a pro další práci se systémem se využily knihovny `sys` a `threading`.

Tabulka 5 obsahuje body, jimiž je trajektorie robota definovaná. Vzhledem k poměrně malému počtu bodů bylo nutné shromáždit více informací souvisejících s pohybem robota.

Za tímto účelem byl vytvořen skript `opc_ua.py` (viz příloha E). Jedná se o OPC UA klienta, který umožňuje připojení k PLC fungující jako OPC UA server, a vyčítat z něj dostupná data. Zároveň se využívá implementovaná modulární knihovna pro řízení robota, která z robota sbírá různá data (kompletní seznam je uveden v příloze A. V rámci skriptu OPC UA klienta se definovalo několik konkrétních uzlů, které potenciálně mohly být relevantní další práci. Jedná se o aktuální:

1) proud ve všech kloubech;
2) pozici kloubů;
3) rychlost kloubů;
4) TCP pozici;
5) TCP rychlost.

Popis struktury skritpu `opc_ua.py` popisuje Algoritmus 5. Výstupem je csv soubor

Algoritmus 5 OPC UA klient

```
1: procedure MAIN
2:     sestavení spojení s OPC UA serverem
3:     vytvoření csv souboru pro průběžné ukládání dat
4:     while !konec do
5:         přečtení hodnot konkrétních uzlů na OPC UA serveru
6:         uložení načtených hodnot uzlů
7:     end while
8:     odpojení od OPC UA serveru
9: end procedure
```

`opcua_data.csv`, který obsahuje sesbíraná data z robota přes PLC, který vykonával jeden cyklus *pick&place* úlohy. Z těchto dat se následně vyextrahovalo 20 bodů na trajektorii robota začínající v bodě *cube_offset* a končící v bodě *cnc_offset*. Tyto souřadnice jsou uloženy v souboru `coordinates.csv` a slouží jako referenční trajektorie pro optimalizaci. Oba soubory lze najít v příloze E.

5.2.1 Minimalizace spotřeby energie

Cílem bylo najít optimální trajektorii, která minimalizuje spotřebu energie. Při implementaci se vycházelo z Algoritmu 4, v němž se specifikovala konkrétní kriteriální funkce, jíž je v tomto případě výpočet energie. Ten vychází ze vztahu:

$$E = \sum_{j=1}^{N} \sum_{i=1}^{M} I_{ji} \cdot V \cdot \Delta t_{ji}, \tag{9}$$

kde *E [J]* je celková spotřeba energie, N je počet kloubů a M je počet časových kroků pro každý kloub, *I [A]* je proud v kloubu j a časovém kroku i, *V [V]* je napětí, Δt_{ji} je rozdíl mezi časovými kroky i a *i-1* pro kloub j.

Algoritmus 6 popisuje pseudokód kriteriální funkce zaměřující se na výpočet spotřeby energie.

Algoritmus 6 Kriteriální funkce

```
1: procedure Spotřeba energie(odchylky)
2:     vytvoření trajektorie se započítáním odchylek
3:     řízení robota paralelně se čtením dat ze simulace
4:     sync
5:     kalkulace celkové spotřeby energie při dané trajektorii
6: end procedure
```

Pro vyhodnocení kriteriální funkce v každé iteraci je nutné ověřit danou trajektorii a její vliv na spotřebu energie. Z bezpečnostních i časových důvodů nebylo možné používat reálného robota, proto se skript `optim.py` pokaždé napojil na simulované prostředí robota a sbíral data z něj.

Při práci se simulací se narazilo na dvě spolusouvisející překážky. 1) Při jedné a té samé úloze generuje simulace poměrně rozdílné hodnoty. Není konzistentní, což znemožňuje ji začlenit do jakýchkoli výpočtů. 2) Při porovnání dat ze simulace s daty sesbíranými z reálného robota (při stejné úloze za stejných podmínek) se dojde k závěru, že ta simulovaná neodpovídají reálnému světu (viz tabulka 6). Vyplývá z toho, že simulace robota v současné době negeneruje relevantní hodnoty proudů, což znemožnilo dále pracovat a úloha minimalizace spotřeby energie musela být v tomto bodě ukončena.

Tab. 6: Porovnání spotřeby energie reálného vs. simulovaného robota

Kloub	**Robot - E[J]**	**Simulace - E[J]**
Kloub 1	-107.80	0.88
Kloub 2	-3275.31	-3128.91
Kloub 3	-3695.78	-4839.55
Kloub 4	-398.74	-425.36
Kloub 5	21.03	0.10
Kloub 6	48.07	-0.95
Celkem	-7408.52	-8393.77

5.2.2 Minimalizace délky trajektorie

Cílem bylo najít trajektorii s minimální délkou. Při implementaci se vycházelo z Algoritmu 4, v němž se specifikovala konkrétní kriteriální funkce, jíž je v tomto případě výpočet dráhy. Ten vychází ze vztahu pro Euklidovskou vzdálenost:

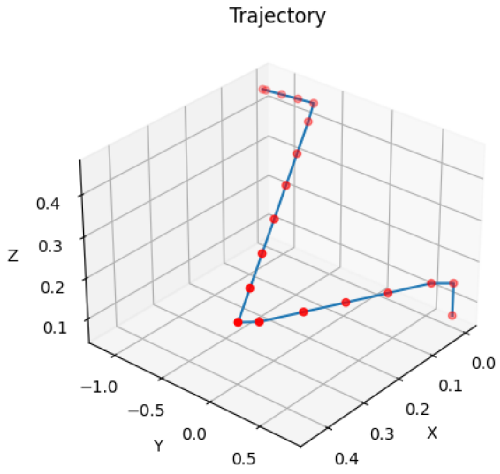
$$dráha = \sum_{i=0}^{n-2} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 + (z_{i+1} - z_i)^2}, \tag{10}$$

kde *x, y, z* jsou souřadnice bodu. Kriteriální funkci popisuje Algoritmus 7.

Původní trajektorie z bodu *cube* do bodu *cnc* byla optimalizována pouze v úseku mezi body *cube_offset* a *cnc_offset*. Krajní body musely zůstat fixní z důvodu bezpečnostních požadavků na dojezd do cílových bodů. Zmíněný úsek znázorněný na Obr. 15 má původní délkou 2,27038 m.

Algoritmus 7 Kriteriální funkce

```
1: procedure Délka trajektorie(odchylky)
2:     vytvoření trajektorie se započítáním odchylek
3:     řízení robota paralelně se čtením dat ze simulace
4:     sync
5:     výpočet délky dané trajektorie
6: end procedure
```



Obr. 15: Původní úsek trajektorie

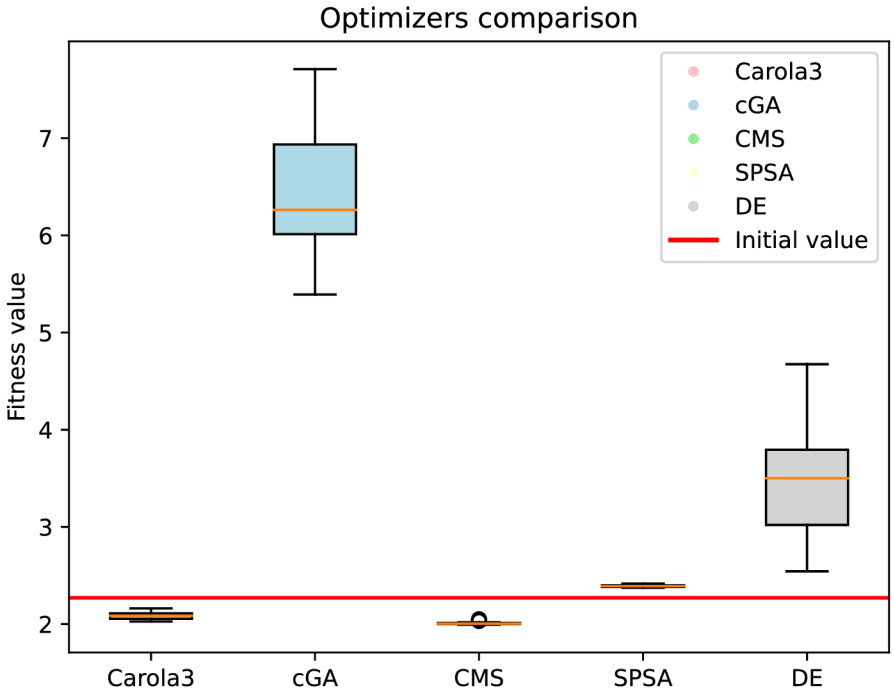
Pro optimalizaci dráhy bylo vybráno sedm algoritmů: NGOpt16, CMA, Carola3, cGA, SPSA, DE a Cobyla. Jejich bližší popis je v kapitole 4. Každý algoritmus byl spuštěn 20krát a výsledky jsou shrnuty v tabulce 8 a grafech na Obr. 16 a 17. Tabulka 7 ukazuje, jak dlouho jednotlivým metodám optimalizace trvala. Z výsledků je patrné, že nejlepších výsledků dosáhl algoritmus CMA, který trajektorii zkrátil až o 12,19 %. Naopak algoritmy cGA, DE a SPSA trajektorii prodloužily.

Tab. 7: Časy optimalizace

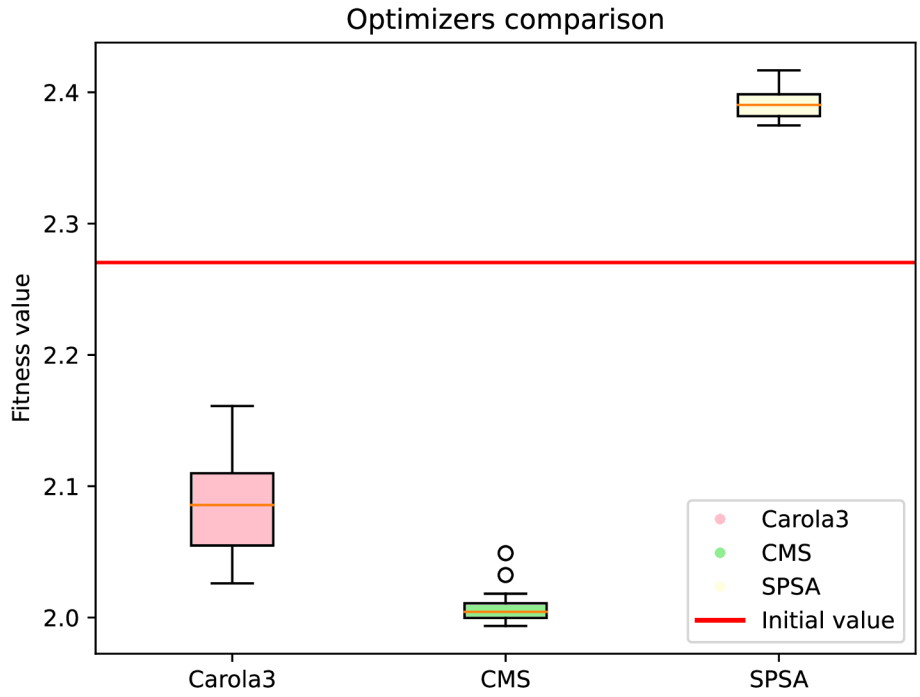
Algoritmus	**Carola3**	**cGA**	**CMA**	**SPSA**
Průměrný čas [s]	70,42823	169,2616	62,394346	53,79372
Algoritmus	**DE**	**NGOpt16**	**Cobyla**	
Průměrný čas [s]	48,82361	59,36872	51,74378	

Tab. 8: Výsledné délky trajektorie pro jednotlivé metody

Délka [m]	**Carola3**	**cGA**	**CMA**	**SPSA**	**DE**	**NGOpt16**	**Cobyla**
Průměr	2,0859	6,3920	2,0076	2,3924	3,5348	2,1930	2,1930
Medián	2,0857	6,2620	2,0043	2,3904	3,5011	2,1930	2,1930
Minimum	2,0260	5,3914	1,9936	2,3748	2,5430	2,1930	2,1930
Maximum	2,1611	7,7109	2,0490	2,4168	4,6740	2,1930	2,1930
Zlepšení [%]	**Carola3**	**cGA**	**CMA**	**SPSA**	**DE**	**NGOpt16**	**Cobyla**
Průměr	8,13	-	11,57	-	-	3,41	3,41
Minimum	4,81	-	1,99	-	-	3,41	3,41
Maximum	10,76	-	12,19	-	-	3,41	3,41

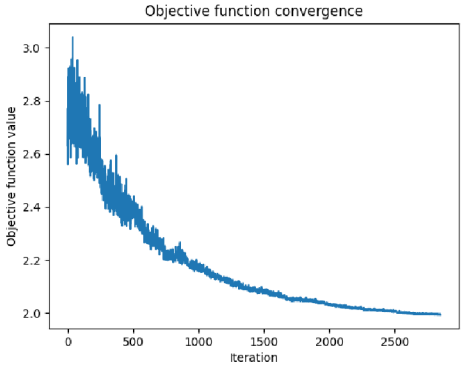


Obr. 16: Výsledky 20ti běhů všech stochastických algoritmů



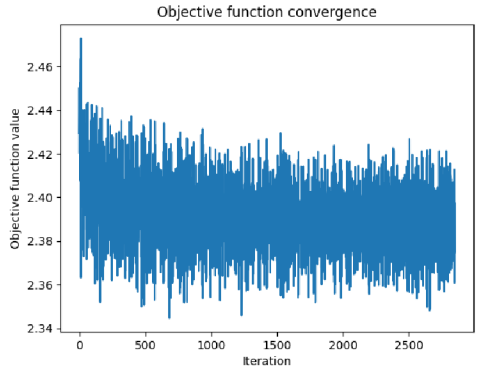
Obr. 17: Výsledky 20ti běhů algoritmů Carola3, CMS a SPSA

Ukončovacím kritériem bylo pro všechny metody počet iterací. Ten byl dle průběhu algoritmu CMA stanoven na 3000 iterací pro každý běh. Z Obr. 18 je vidět, že zhruba u této hodnoty se hodnota kriteriální funkce ustálí. Pro porovnání byl ukončovací parametr pro všechny metody stejný.

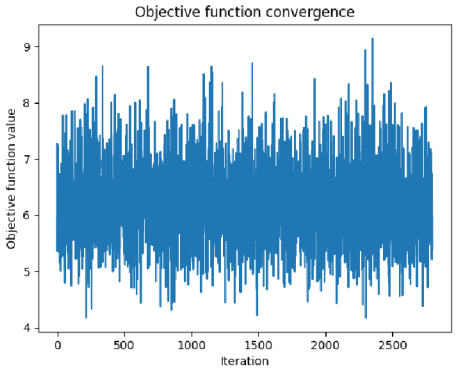


Obr. 18: Konvergence kriteriální funkce pro algoritmus CMA

Zvolený počet iterací v případě algoritmů SPSA a cGA neměl na optimalizaci dle Obr. 19 a 20 významný vliv. Hodnota kriteriální funkce s velkými výkyvy osciluje okolo hodnoty cca 2.29 m, resp. 6.2 m a nic nenasvědčuje tomu, že navýšení počtu iterací by mělo vliv na výsledek.

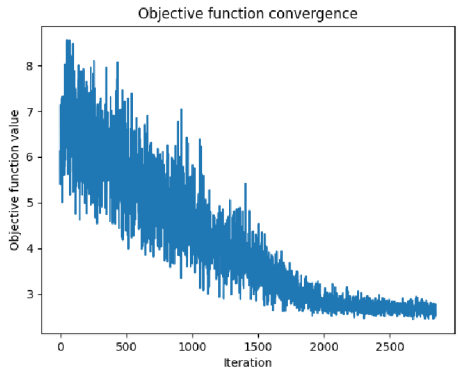


Obr. 19: Konvergence kriteriální funkce pro algoritmus SPSA

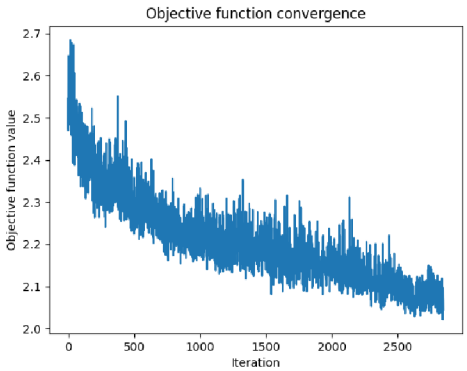


Obr. 20: Konvergence kriteriální funkce pro algoritmus cGA

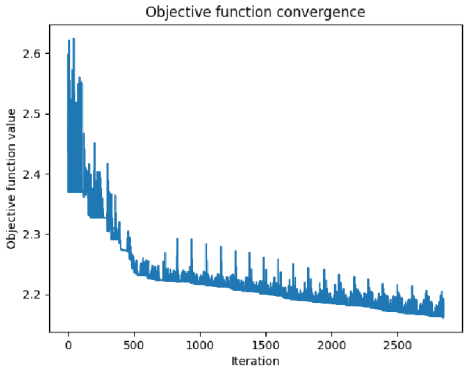
Pro diferenciální evoluci je podle Obr. 21 počet iterací taktéž poměrně vhodně zvolený, protože hodnota kriteriální funkce se zhruba od dvoutisící iterace ustaluje. Naopak v případě algoritmů Carola3, Cobyla a NGOp16 počet iterací je nedostatečný, protože z průběhu kriteriální funkce na Obr. 22, 23 a 24 je vidět, že do ustálení hodnoty je třeba ještě několik iterací. Zároveň Obr. 23 a 24 dokazují předpoklad z kapitoly 4.1, že optimalizátor NGOpt16 si díky vnitřní logice vybral k řešení této úlohy algoritmus Cobyla.



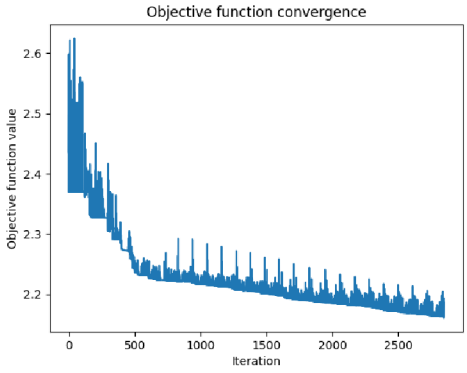
Obr. 21: Konvergence kriteriální funkce pro algoritmus DE



Obr. 22: Konvergence kriteriální funkce pro algoritmus Carola3



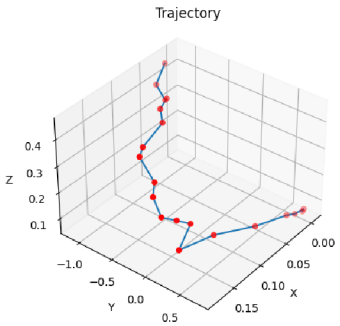
Obr. 23: Konvergence kriteriální funkce pro algoritmus Cobyla



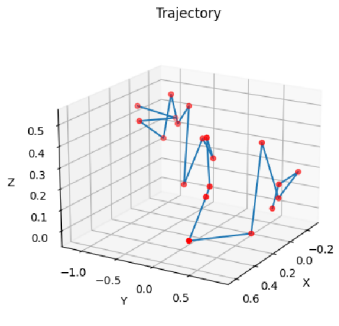
Obr. 24: Konvergence kriteriální funkce pro algoritmus NGOpt16

Na následujících obrázcích Obr. 25 - Obr. 31 jsou zobrazeny trajektorie nejlepších běhů pro každý algoritmus. I z nich je patrné, že algoritmus CMA dosáhl poměrně dobrých výsledků, zatímco například algoritmus cGA pro danou úlohu vhodný není.

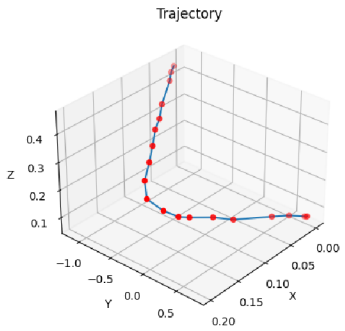
Veškeré výsledky jsou v příloze E. Pro každý běh je k dispozici graf výsledné trajektorie a průběh kriteriální funkce v průběhu iterací. Dále jeden textový soubor `results.txt`, kde jsou výsledné odchylky pro každý původní bod referenční trajektorie. A v neposlední řadě csv soubor `optim.csv` s body optimální trajektorie.



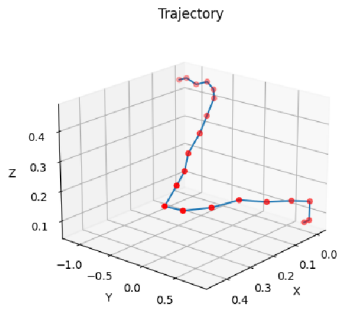
Obr. 25: Optimální trajektorie algoritmu Carola3



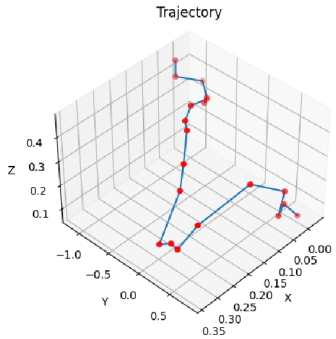
Obr. 26: Optimální trajektorie algoritmu cGA



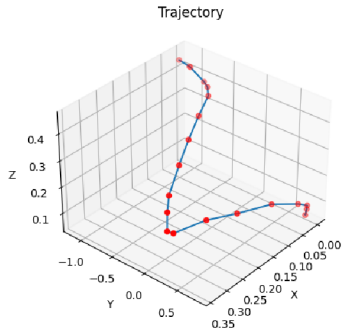
Obr. 27: Optimální trajektorie algoritmu CMA



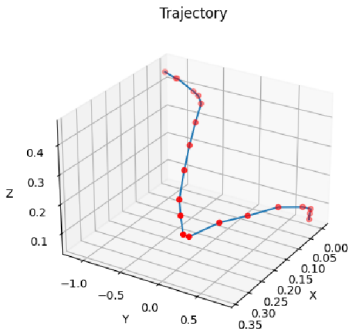
Obr. 28: Optimální trajektorie algoritmu SPSA



Obr. 29: Optimální trajektorie algoritmu DE



Obr. 30: Optimální trajektorie algoritmu NGOpt16



Obr. 31: Optimální trajektorie algoritmu Cobyla

6 ZÁVĚR

Diplomová práce se zabývala systémovou integrací s ohledem na principy čtvrté průmyslové revoluce, jimiž mimo jiné jsou automatizace, digitalizace a propojování všech zařízení výrobního procesu tak, aby byla spolu schopna komunikovat ve vertikálním i horizontálním směru napříč celým podnikem.

První část práce se zabývala řízením kolaborativního robota Universal Robots UR10e prostřednictvím PLC od společnosti Siemens. Vlastnímu řešení předcházela rešerše v oblasti vertikálních komunikačních protokolů, na základně které se pro komunikaci mezi robotem a PLC vybral komunikační protokol PROFINET. PROFINET je nejrozšířenějším průmyslovým protokolem vhodným pro komunikaci v reálném čase, což pro danou aplikaci byla klíčová vlastnost. Dále neméně důležitým faktem bylo, že se jednalo o protokol, který podporovala obě zařízení, tj. jak UR robot, tak i PLC Siemens.

Hlavním cílem byl návrh a implementace modulární knihovny, která by sloužila k řízení robota prostřednictvím PLC i ke sběru dat z robota do PLC, kde jsou připravena k odeslání kamkoli v rámci podniku. Knihovna byla vyvíjena v licencovaném vývojovém prostředí TIA Portal V18. Architektura knihovny je koncipována jako stavový automat. Výsledný formát je funkční blok (FB), který je snadno přenositelný mezi různými projekty s PLC od společnosti Siemens.

Výsledné řešení bylo úspěšně otestované v rámci *pick&place* úlohy v interakci s reálným robotem, která vytvořenou knihovnu využívá. Výstup je ve formátu videa v příloze D. Testování v prostředí simulace nebylo možné realizovat kvůli nedostupným licencím.

V druhé části diplomové práce bylo cílem navrhnout program pro statickou optimalizaci trajektorie robota. Prvním krokem bylo získat informace o bodech trajektorie, které sloužily jako referenční dráha, která byla následně optimalizována. Za tímto účelem byl vytvořen OPC UA klient, pomocí něhož bylo možné vyčítat z PLC jakákoli potřebná data, která vytvořená modulární knihovna z robota sbírala v reálném čase.

Ve druhém kroku byl navrhnut samotný program. Program se prvně implementoval pro úlohu minimalizace spotřeby energie. V tomto případě nebylo možné experimenty realizovat kvůli nedostatečné funkcionalitě simulace, která z bezpečnostních důvodů byla pro optimalizaci nutná.

Navržený program byl použit i pro minimalizaci ujeté dráhy robota. Experimenty se zkoušely se sedmi různými algoritmy dostupnými v Python knihovně *Nevergrad* od společnosti Meta (dříve Facebook). Tři algoritmy byly za použitého nastavení parametrů vyhodnoceny jako nedostačující, protože dráhu prodloužily.

Nejlépe z použitých algoritmů vyšla metoda CMA, která dokázala dráhu zkrátit až o 12,19 %.

Všechny předem stanovené cíle práce byly splněny. Vytvořená řešení spolu s výsledky experimentů jsou přiložena v přílohách A až E.

SEZNAM POUŽITÉ LITERATURY

[1] KAGERMANN, Henning; WOLF-DIETER, Lukas a WAHLSTER, Wolfgang. Industrie 4.0: Mit dem Internet der Dinge auf dem Weg zur 4. industriellen Revolution. VDI nachrichten. 2011, č. 13.

[2] ZHANG, Caiming; CHEN, Yong; CHEN, Hong a CHONG, Dazhi. Industry 4.0 and its Implementation: a Review. Online. *Information Systems Frontiers*. ISSN 1387-3326. Dostupné z: https://doi.org/10.1007/s10796-021-10153-5. [cit. 2024-02-09].

[3] KADA, Belkacem; ALZUBAIRI, Ahmed a TAMEEM, Abdullah. Industrial Communication Networks and the Future of Industrial Automation. Online. In: *2019 Industrial & Systems Engineering Conference (ISEC)*. IEEE, 2019, s. 1-5. ISBN 978-1-7281-0145-3. Dostupné z: https://doi.org/10.1109/IASEC.2019.8686664. [cit. 2024-02-09].

[4] GUPTA, Vibhoosh. Jaký ethernetový protokol je pro vaši podnikovou komunikační síť ten pravý? Online. *Automa*. 2020, roč. 13, č. 7, s. 26-28. Dostupné z: https://www.automa.cz/Aton/FileRepository/pdf_articles/13047.pdf. [cit. 2024-02-09].

[5] *Industrial Ethernet and Wireless are growing fast — Industrial network market shares 2017 according to HMS*. Online. HMS Networks. C2022. Dostupné z: https://www.hms-networks.com/news-and-insights/news-from-hms/2017/02/20/industrial-ethernet-and-wireless-are-growing-fast-industrial-network-market-shares-2017-according-to-hms. [cit. 2024-02-09].

[6] CARLSSON, Thomas. *Industrial network market shares 2023*. Online. HMS Networks. C2022. Dostupné z: https://www.hms-networks.com/news-and-insights/news-from-hms/2023/05/05/industrial-network-market-shares-2023. [cit. 2024-02-09].

[7] DIAS, Andre Luis; SESTITO, Guilherme Serpa; TURCATO, Afonso Celso a BRANDAO, Dennis. Panorama, challenges and opportunities in PROFINET protocol research. Online. In: *2018 13th IEEE International Conference on Industry Applications (INDUSCON)*. IEEE, 2018, s. 186-193. ISBN 978-1-5386-7995-1. Dostupné z: https://doi.org/10.1109/INDUSCON.2018.8627173. [cit. 2024-02-10].

[8] *Technology Description*. Online. PROFIBUS Nutzerorganisation e.V. C2024. Dostupné z: https://www.profinet.com/profinet-explained/technology-description. [cit. 2024-02-10].

[9] PROFINET Technology The Easy Way to PROFINET. Online. 2018, s. 4-15. Dostupné z: https://www.profibus.com/index.php?eID=dumpFile&t=f&f=85452&token=b6aa92b46b7e239aa199e94421269a2a7acd5014. [cit. 2024-02-10].

[10] NASRALLAH, Ahmed; THYAGATURU, Akhilesh S.; ALHARBI, Ziyad; WANG, Cuixiang; SHAO, Xing et al. Ultra-Low Latency (ULL) Networks: The IEEE TSN and IETF DetNet Standards and Related 5G ULL Research. Online. *IEEE Communications Surveys & Tutorials*. 2019, roč. 21, č. 1, s. 88-145. ISSN 1553-877X. Dostupné z: https://doi.org/10.1109/COMST.2018.2869350. [cit. 2024-02-10].

[11] ZEZULKA, František a HYNČICA, Ondřej. *Průmyslový Ethernet IX: EtherNet/IP, EtherCAT*. Online. Automa. 2008, roč. 1, č. 10, s. 60-64. Dostupné z: https://www.automa.cz/cz/casopis-clanky/prumyslovy-ethernet-ix-ethernet/ip-ethercat-2008_10_37910_6510/. [cit. 2024-02-12].

[12] *TECHNOLOGY OVERVIEW SERIES: CIP on Ethernet Technology*. Online. ODVA, 2021. Dostupné z: https://www.odva.org/wp-content/uploads/2021/05/PUB00138R7_Tech-Series-EtherNetIP.pdf. [cit. 2024-02-12].

[13] *EtherNet/IP uses the lower layers of standard Ethernet, with the Common Industrial Protocol (CIP) over the TCP/UDP transport layer*. Online. In: Motion Control Tips. 2024. Dostupné z: https://www.motioncontroltips.com/ethernet-ip-versus-ethercat-whats-the-difference/. [cit. 2024-02-12].

[14] BROOKS, P. Ethernet/IP-industrial protocol. Online. In: *ETFA 2001. 8th International Conference on Emerging Technologies and Factory Automation. Proceedings (Cat. No.01TH8597)*. IEEE, 2001, s. 505-514. ISBN 0-7803-7241-7. Dostupné z: https://doi.org/10.1109/ETFA.2001.997725. [cit. 2024-02-12].

[15] *EtherCAT - the Ethernet Fieldbus*. Online. EtherCAT Technology Group. 2022. Dostupné z: https://www.ethercat.org/en/technology.html#1.11. [cit. 2024-02-12].

[16] SOURY, Ayoub; CHARFI, Melek; GENON-CATALOT, Denis a THIRIET, Jean-Marc. Performance analysis of Ethernet Powerlink protocol: Application to a new lift system generation. Online. In: *2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA)*. IEEE, 2015, s. 1-6. ISBN 978-1-4673-7929-8. Dostupné z: https://doi.org/10.1109/ETFA.2015.7301492. [cit. 2024-02-13].

[17] WLAS, Miroslaw; GACKOWSKI, Marek a KOLBUSZ, Wojciech. The Ethernet POWERLINK Protocol for smart grids elements integration. Online. In: *2011*

IEEE International Symposium on Industrial Electronics. IEEE, 2011, s. 2070-2075. ISBN 978-1-4244-9310-4. Dostupné z: https://doi.org/10.1109/ISIE.2011.5984479. [cit. 2024-02-13].

[18] BARRIOS-AVILÉS, Juan; ROSADO-MUÑOZ, Alfredo; IAKYMCHUK, Taras a GARCÍA-CHULBI, Marcos. POWERLINK and Ethernet/IP Comparison as Robust Industrial Ethernet Protocols. Online. *IFAC-PapersOnLine*. 2017, roč. 50, č. 1, s. 363-368. ISSN 24058963. Dostupné z: https://doi.org/10.1016/j.ifacol.2017.08.159. [cit. 2024-02-13].

[19] KARABEGOVIĆ, I. Comparative analysis of automation of production process with industrial robots in Asia/Australia and Europe. Olnine. In: *International Journal of Human Capital in Urban Management*, 2017, s. 29-39. Dostupné z: https://www.ijhcum.net/article_24992_b8ba9f742551a33ee595a321bedd8b07.pdf. [cit. 2024-02-15].

[20] SHERWANI, F.; ASAD, Muhammad Mujtaba a IBRAHIM, B.S.K.K. Collaborative Robots and Industrial Revolution 4.0 (IR 4.0). Online. In: *2020 International Conference on Emerging Trends in Smart Technologies (ICETST)*. IEEE, 2020, s. 1-5. ISBN 978-1-7281-7113-5. Dostupné z: https://doi.org/10.1109/ICETST49965.2020.9080724. [cit. 2024-02-15].

[21] BRAGANÇA, Sara; COSTA, Eric; CASTELLUCCI, Ignacio a AREZES, Pedro M. A Brief Overview of the Use of Collaborative Robots in Industry 4.0: Human Role and Safety. Online. In: AREZES, Pedro M.; BAPTISTA, João S.; BARROSO, Mónica P.; CARNEIRO, Paula; CORDEIRO, Patrício et al. (ed.). *Occupational and Environmental Safety and Health*. Studies in Systems, Decision and Control. Cham: Springer International Publishing, 2019, s. 641-650. ISBN 978-3-030-14729-7. Dostupné z: https://doi.org/10.1007/978-3-030-14730-3_68. [cit. 2024-02-15].

[22] KHALID, A., et al. Towards implementing safety and security concepts for human-robot collaboration in the context of Industry 4.0. In: *39th International MATADOR Conference on Advanced Manufacturing*. 2017. p. 55-63. Dostupné z: https://www.researchgate.net/profile/Azfar-Khalid/publication/318340673_Implementing_Safety_and_Security_Concepts_for_Human-Robot_Collaboration_in_the_context_of_Industry_40/links/59f31941aca272607e27035a/Implementing-Safety-and-Security-Concepts-for-Human-Robot-Collaboration-in-the-context-of-Industry-40.pdf. [cit. 2024-02-15].

[23] ROBLA-GOMEZ, S.; BECERRA, Victor M.; LLATA, J. R.; GONZALEZ-SARABIA, E.; TORRE-FERRERO, C. et al. Working Together: A Review on

Safe Human-Robot Collaboration in Industrial Environments. Online. *IEEE Access*. 2017, roč. 5, s. 26754-26773. ISSN 2169-3536. Dostupné z: https://doi.org/10.1109/ACCESS.2017.2773127. [cit. 2024-02-15].

[24] MUNOZ, Luis Miguel. Ergonomics in the Industry 4.0: Exoskeletons. Online. *Journal of Ergonomics*. 2017, roč. 08, č. 01. ISSN 21657556. Dostupné z: https://doi.org/10.4172/2165-7556.1000e176. [cit. 2024-02-13].

[25] *UR10e*. Online. Universal Robots A/S. 2024. Dostupné z: https://www.universal-robots.com/products/ur10-robot/. [cit. 2024-02-17].

[26] *CNC frézka*. Online. Solid Vision. 2024. Dostupné z: https://www.solidvision.cz/dalsi-produkty-a-sluzby/cnc-frezka. [cit. 2024-02-17].

[27] *Basic Controller SIMATIC S7-120*. Online. Siemens, 2017. Dostupné z: https://www.eandm.com/Products/Content/Siemens/PDFs/simatic_basic_controller_s7-1200_en.pdf. [cit. 2024-02-17].

[28] RATIU, M; GREBENISAN, G; PRICHICI, M A a BOGDAN, S. Optimization methods of the industrial robots' trajectory. Online. *IOP Conference Series: Materials Science and Engineering*. 2019, roč. 568, č. 1. ISSN 1757-8981. Dostupné z: https://doi.org/10.1088/1757-899X/568/1/012009. [cit. 2024-05-11].

[29] PARYANTO; BROSSOG, Matthias; BORNSCHLEGL, Martin a FRANKE, Jörg. Reducing the energy consumption of industrial robots in manufacturing systems. Online. *The International Journal of Advanced Manufacturing Technology*. 2015, roč. 78, č. 5-8, s. 1315-1328. ISSN 0268-3768. Dostupné z: https://doi.org/10.1007/s00170-014-6737-z. [cit. 2024-05-11].

[30] MEUNIER, Laurent; RAKOTOARISON, Herilalaina; WONG, Pak Kan; ROZIERE, Baptiste; RAPIN, Jeremy et al. Black-Box Optimization Revisited: Improving Algorithm Selection Wizards Through Massive Benchmarking. Online. *IEEE Transactions on Evolutionary Computation*. 2022, roč. 26, č. 3, s. 490-500. ISSN 1089-778X. Dostupné z: https://doi.org/10.1109/TEVC.2021.3108185. [cit. 2024-05-06].

[31] Nevergrad. Online. 2019. Dostupné z: https://facebookresearch.github.io/nevergrad/. [cit. 2024-05-06].

[32] RAPONI, Elena; RAKOTONIRINA, Nathanaël Carraz; RAPIN, Jérémy; DOERR, Carola a TEYTAUD, Olivier. Optimizing With Low Budgets: A Comparison On the Black-Box Optimization Benchmarking Suite and OpenAI Gym.

Online. *IEEE Transactions on Evolutionary Computation*. S. 1-1. ISSN 1089-778X. Dostupné z: https://doi.org/10.1109/TEVC.2023.3346788. [cit. 2024-05-06].

[33] GHOSH, Saurav; DAS, Swagatam a ROY, Subhrajit et al. A Differential Covariance Matrix Adaptation Evolutionary Algorithm for real parameter optimization. Online. *Information Sciences*. 2012, roč. 182, č. 1, s. 199-219. Dostupné z:https://doi.org/10.1016/j.ins.2011.08.014. [cit. 2024-05-07].

[34] LIANG, Yajun; WANG, Xiaofei; ZHAO, Hui; HAN, Tong; WEI, Zhenglei et al. A covariance matrix adaptation evolution strategy variant and its engineering application. Online. *Applied Soft Computing*. 2019, roč. 83. ISSN 15684946. Dostupné z: https://doi.org/10.1016/j.asoc.2019.105680. [cit. 2024-05-07].

[35] MAYER, D.G.; KINGHORN, B.P. a ARCHER, A.A. Differential evolution – an easy and efficient evolutionary algorithm for model optimisation. Online. Agricultural Systems. 2005, roč. 83, č. 3, s. 315-328. ISSN 0308521X. Dostupné z: https://doi.org/10.1016/j.agsy.2004.05.002. [cit. 2024-05-07].

[36] BILAL; PANT, Millie; ZAHEER, Hira; GARCIA-HERNANDEZ, Laura a ABRAHAM, Ajith. Differential Evolution: A review of more than two decades of research. Online. *Engineering Applications of Artificial Intelligence*. 2020, č. 90. Dostupné z: https://doi.org/10.1016/j.engappai.2020.103479. [cit. 2024-05-07].

[37] HARIK, G.R.; LOBO, F.G. a GOLDBERG, D.E. The compact genetic algorithm. Online. *IEEE Transactions on Evolutionary Computation*. Roč. 3, č. 4, s. 287-297. ISSN 1089778X. Dostupné z: https://doi.org/10.1109/4235.797971. [cit. 2024-05-05].

[38] RIBEIRO E SILVA, S.; GOMES, R.P.F. a FALCÃO, A.F.O. Hydrodynamic optimization of the UGEN: Wave energy converter with U-shaped interior oscillating water column. Online. *International Journal of Marine Energy*. 2016, roč. 15, s. 112-126. ISSN 22141669. Dostupné z: https://doi.org/10.1016/j.ijome.2016.04.013. [cit. 2024-05-08].

[39] SPALL, James C. Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control. John Wiley & Sons, 2003. ISBN 0-471-33052-3. [cit. 2024-05-08].

[40] LI, Lianlin; JAFARPOUR, Behnam a MOHAMMAD-KHANINEZHAD, M. Reza. A simultaneous perturbation stochastic approximation algorithm for coupled well placement and control optimization under geologic uncertainty.

Online. Computational Geosciences. 2013, roč. 17, č. 1, s. 167-188. ISSN 1420-0597. Dostupné z: https://doi.org/10.1007/s10596-012-9323-1. [cit. 2024-05-08].

[41] ŠTĚRBOVÁ, Nina. Řízení kolaborativního robota Universal Robots UR10e prostřednictvím PLC od společnosti Siemens. Online. *Zenodo*. Dostupné z: https://doi.org/10.5281/zenodo.11214894. [cit. 2024-05-19].

SEZNAM ZKRATEK

AMQT Advanced Message Queuing Protocol

AR Application Relationship

CIP Common Industrial Protocol

CN Controlled Node

CPS Kyber-fyzikální systémy (Cyber-Physical Systems)

CR Communication Relationships

DB Data blok

ESC EtherCAT Slave Controller

FB Funkční blok

GB Gigabyte

HMI Human Machine Interface

I/O Vstup/výstup (Input/Output)

IoT Internet věcí (Internet of Things)

IRT Isochronous Real Time

IT Informační technologie

MAC Media Access Controller

MN Managing Node

MQTT Message Queuing Telemetry Transport

ODVA Open DeviceNet Vendors Association

OSI Open Systems Interconnection

PLC Programovatelný logický automat (Programmable Logic Controller)

RT Real Time

SCADA Supervisory Control and Data Acquisition

SCNM Slot Communication Network Management

SOA Architektura orientovaná na služby (Service-oriented Architecture)

TCP To Center Point - jedná se o pohybový mod robota

TSN Time Sensitive Networking

VPN Virtuální privátní síť (Virtual Private Network)

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

A Vstupní data z UR do PLC

Tato příloha obsahuje tabulky 9 až 12, ve kterých je kompletní seznam dat, která se přenášejí přes PROFINET z UR robota do PLC Siemens.

Tab. 9: Vstupy do PLC z robota - IO

Název	**Datový typ**	**Adresa**
Standard digital inputs	USInt	%IB68
Configurable digital inputs	USInt	%IB69
Standard digital outputs	USInt	%IB67
Configurable digital outputs	USInt	%IB71
Analog I/O types	USInt	%IB72
Reserved_8_15(1)	USInt	%IB73
Reserved_16_31_3	UInt	%IW74
Standard analog input [A or V][0]	Real	%ID76
Standard analog input [A or V][1]	Real	%ID80
Standard analog output [A or V][0]	Real	%ID84
Standard analog output [A or V][1]	Real	%ID88
I/O current [A]	Real	%ID92
Euromap67 24V voltage	Real	%ID104
Euromap67 24V current	Real	%ID108
Tool mode	USInt	%IB112
Reserved_7_15	USInt	%IB113
Reserved_16_31_4	UInt	%IW114
TDI: Tool Digital Inputs	USInt	%IB116
TDO: Tool Digital Outputs	USInt	%IB117
TAIT: Tool Analog Input Types	USInt	%IB118
Reserved_24_31	USInt	%IB119
Tool analog input [A or V][0]	Real	%ID120
Tool analog input [A or V][1]	Real	%ID124
Tool output voltage [V]	Real	%ID128
Tool current [A]	Real	%ID132

Tab. 10: Vstupy do PLC z robota - Klouby

Název	**Datový typ**	**Adresa**
Joint position [rad][0]	Real	%ID136
Joint position [rad][1]	Real	%ID140
Joint position [rad][2]	Real	%ID144
Joint position [rad][3]	Real	%ID148
Joint position [rad][4]	Real	%ID152
Joint position [rad][5]	Real	%ID156
Joint velocity [rad/s][0]	Real	%ID160
Joint velocity [rad/s][1]	Real	%ID164
Joint velocity [rad/s][2]	Real	%ID168
Joint velocity [rad/s][3]	Real	%ID172
Joint velocity [rad/s][4]	Real	%ID176
Joint velocity [rad/s][5]	Real	%ID180
Joint current [A][0]	Real	%ID184
Joint current [A][1]	Real	%ID188
Joint current [A][2]	Real	%ID192
Joint current [A][3]	Real	%ID196
Joint current [A][4]	Real	%ID200
Joint current [A][5]	Real	%ID204
Joint temperature [°C][0]	Real	%ID208
Joint temperature [°C][1]	Real	%ID212
Joint temperature [°C][2]	Real	%ID216
Joint temperature [°C][3]	Real	%ID220
Joint temperature [°C][4]	Real	%ID224
Joint temperature [°C][5]	Real	%ID228
Joint mode [0]	USInt	%IB232
Joint mode [1]	USInt	%IB233
Joint mode [2]	USInt	%IB234
Joint mode [3]	USInt	%IB235
Joint mode [4]	USInt	%IB236
Joint mode [5]	USInt	%IB237
Reserved_9	UInt	%IW238

Tab. 11: Vstupy do PLC z robota - Stav

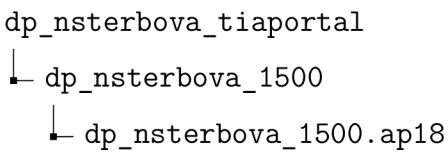
Název	**Datový typ**	**Adresa**
Controller major version	USInt	%IB2
Controller minor version	USInt	%IB3
Reserved	UInt	%IW4
Robot mode	USInt	%IB6
Real time machines seconds	USInt	%IB7
Real time machines milliseconds	UInt	%IW8
Real time machines minutes	USInt	%IB10
Real time machines hours	USInt	%IB11
Real time machines days	UInt	%IW12
Robot current [A]	Real	%ID14
PW: Is power on	Bool	%I18.0
PR: Is program running	Bool	%I18.1
TB: Is teach button pressed	Bool	%I18.2
PB: Is power button pressed	Bool	%I18.3
Reserved_4	Bool	%I18.4
Reserved_5	Bool	%I18.5
Reserved_6	Bool	%I18.6
Reserved_7	Bool	%I18.7
Reserved_8_15	USInt	%IB19
Reserved_16_31	UInt	%IW20
Speed slider fraction	Real	%ID22
Safety mode	USInt	%IB26
Reserved_1	USInt	%IB27
Reserved_2	UInt	%IW28
NO: Is normal mode	Bool	%I30.0
RD: Is reduced mode	Bool	%I30.1
PS: Is protective stopped	Bool	%I30.2
RC: Is recovery mode	Bool	%I30.3
SS: iS safeguard stopped	Bool	%I30.4
SES: Is system emergency stopped	Bool	%I30.5
RES: Is robot emergency stopped	Bool	%I30.6
ES: Is emergency stopped	Bool	%I30.7
VL: Is violation	Bool	%I31.0
FT: Is fault	Bool	%I31.1
ST: Is stopped due to safety	Bool	%I31.2

Tab. 12: Vstupy do PLC z robota - TCP

Název	**Datový typ**	**Adresa**
TCP Position (X,Y,Z) [m][0]	Real	%ID240
TCP Position (X,Y,Z) [m][1]	Real	%ID244
TCP Position (X,Y,Z) [m][2]	Real	%ID248
TCP Position (RX,RY,RZ) [rad][0]	Real	%ID252
TCP Position (RX,RY,RZ) [rad][1]	Real	%ID256
TCP Position (RX,RY,RZ) [rad][2]	Real	%ID260
TCP velocity (X,Y,Z) [m/s][0]	Real	%ID264
TCP velocity (X,Y,Z) [m/s][1]	Real	%ID268
TCP velocity (X,Y,Z) [m/s][2]	Real	%ID272
TCP velocity (RX,RY,RZ) [rad/s][0]	Real	%ID276
TCP velocity (RX,RY,RZ) [rad/s][1]	Real	%ID280
TCP velocity (RX,RY,RZ) [rad/s][2]	Real	%ID284
TCP force (X,Y,Z) [N][0]	Real	%ID288
TCP force (X,Y,Z) [N][1]	Real	%ID292
TCP force (X,Y,Z) [N][2]	Real	%ID296
TCP torque (X,Y,Z) [Nm][0]	Real	%ID300
TCP torque (X,Y,Z) [Nm][1]	Real	%ID304
TCP torque (X,Y,Z) [Nm][2]	Real	%ID308
TCP force scalar [N]	Real	%ID312

B TIA Portal projekt

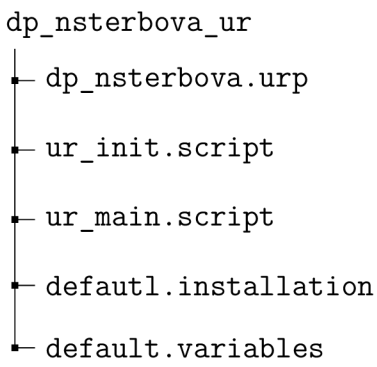
Příloha B obsahuje podadresář složky `NinaSterbova_218023_DP_2024_UAI.zip`, který je dostupný v [41]. Podadresář znázorněný na Obr. 32 obsahuje projekt vytvořené v prostředí TIA Portal V18.



Obr. 32: Struktura podadresáře `dp_nsterbova_tiaportal`

C UR skripty

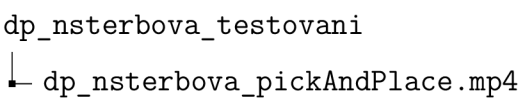
Příloha C obsahuje podadresář složky `NinaSterbova_218023_DP_2024_UAI.zip`, který je dostupný v [41]. Podadresář znázorněný na Obr. 33 obsahuje skripty vytvořené pro UR robota UR10e.



Obr. 33: Struktura podadresáře `dp_nsterbova_ur`

D Testování řešení

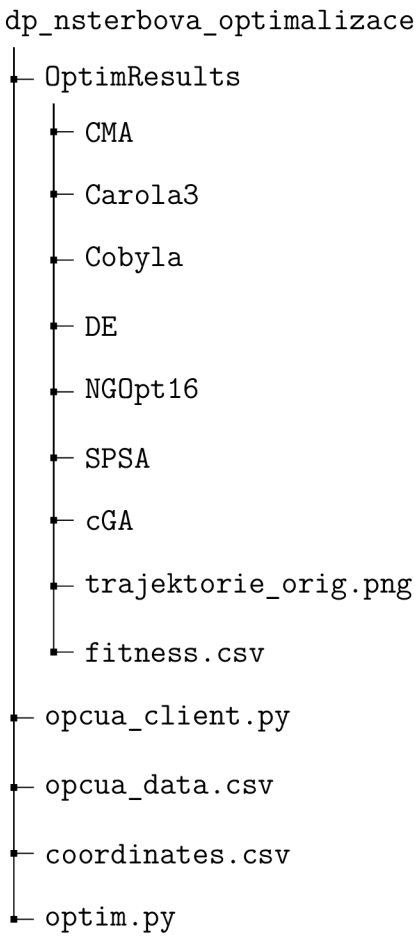
Příloha D obsahuje podadresář složky `NinaSterbova_218023_DP_2024_UAI.zip`, který je dostupný v [41]. Podadresář znázorněný na Obr. 34 obsahuje nahrané video zachycující jeden cyklus úlohu *pick&place* úlohy, kde kolaborativní robot vezme do gripperu kostku, založí ji do frézky a vrácí se do výchozí polohy.



Obr. 34: Struktura podadresáře `dp_nsterbova_testovani`

E Optimalizace

Příloha E obsahuje podadresář složky `NinaSterbova_218023_DP_2024_UAI.zip`, který je dostupný v [41]. Podadresář znázorněný na Obr. 35 obsahuje Python skript vytvořeného OPC UA klienta, který sloužil ke sběru dat z reálného robota. Dále Python script pro optimalizaci trajektorie definovanou přiloženými souřadnicemi. A v neposlední řadě složku `OptimResults` s výsledky optimalizace ze všech běhů pro všechny použité algoritmy.



Obr. 35: Struktura podadresáře `dp_nsterbova_optimalizace`