UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

Robotický manipulátor ovládaný strojovým viděním

Pavel Kotomtsev

Bakalářská práce
2024

Univerzita Pardubice

Fakulta elektrotechniky a informatiky

Robotický manipulátor ovládaný strojovým viděním

Bakalářská práce

2024 Kotomtsev Pavel

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: 2023/2024

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Pavel Kotomtsev**
Osobní číslo:	**I21050**
Studijní program:	**B0714A150008 Automatizace**
Téma práce:	**Robotický manipulátor ovládaný strojovým viděním**
Zadávající katedra:	**Katedra řízení procesů**

Zásady pro vypracování

Cílem bakalářské práce je konstrukce robotického manipulátoru, ovládaného kamerovým systémem se strojovým viděním, s využitím algoritmů umělé inteligence. Sestava robotického manipulátoru a kamerového systému bude testována na úloze třídění cílových objektů. Tyto objekty budou tříděny dle třídicích kritérií, kterými bude např. barva, velikost, nebo tvar. Ke konstrukci řídicí jednotky zařízení bude použit vybraný typ jednočipového mikropočítače, např. řady ATmega, který může být součástí kompletního vývojového kitu (např. Arduino Mini, UNO, DUE atp.). Zařízení bude řízeno nadřazeným řídicím systémem osobního počítače. Komunikace řídicí jednotky s nadřazeným řídicím systémem bude realizována vybraným typem bezdrátového komunikačního rozhraní.
Teoretická část bude obsahovat rešerši zadaného tématu se zaměřením na hlavní směry konstrukčního řešení srovnatelných technických zařízení a použitých konstrukčních prvků ve vlastním návrhu.
Praktická část bude zaměřena na podrobný popis konstrukce zařízení a jeho testování, včetně příslušného zhodnocení dosažených výsledků. Konstrukční řešení bude zpracováno převážně ve formě 3D modelů, ve vybraném typu návrhového 3D software a následně realizováno s využitím technologie 3d tisku. Firmware řídicí jednotky bude navržen a realizován v jazyce C, C++ pro mikrokontrolery (případně jejich klonů) a software nadřazeného řídicího systému osobního počítače bude realizován v jazyce Python.
Nedílnou součástí práce bude i podrobně zpracovaná výrobní dokumentace, zdokumentované zdrojové kódy firmware mikropočítače a osobního počítače a uživatelský manuál.

Rozsah pracovní zprávy: **40 – 50 stran A4**
Rozsah grafických prací:
Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

MATOUŠEK, D. Práce s mikrokontroléry ATMEL AVR-3.díl, edice uP a praxe, 2. vydání, BEN – technická literatura, 2006, ISBN 80-7300-209-4
MAIXNER, L. a kol., Mechatronika, Brno, Computer Press, 2006, ISBN 80-251-1299-3
SZELISKI, Richard. Computer Vision: Algorithms and Applications. Texts in computer science. London: Springer, 2010. ISBN 978-1-84882-934-3.
RASCHKA, Sebastian a MIRJALILI, Vahid. Python machine learning: machine learning and deep learning with Python, scikit-learn, and TensorFlow. Second edition. Expert insight. Birmingham: Packt, 2017. ISBN 978-1-78712-593-3.

Vedoucí bakalářské práce: **Ing. Libor Havlíček, Ph.D.**
Katedra řízení procesů

Datum zadání bakalářské práce: **15. prosince 2023**
Termín odevzdání bakalářské práce: **10. května 2024**

L.S.

Ing. Zdeněk Němec, Ph.D. v.r.
děkan

Ing. Daniel Honc, Ph.D. v.r.
vedoucí katedry

V Pardubicích dne 12. ledna 2024

Prohlašuji:

Práci s názvem „Robotický manipulátor ovládaný strojovým viděním“

jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 7/2019 Pravidla pro odevzdávání, zveřejňování a formální úpravu závěrečných prací, ve znění pozdějších dodatků, bude práce zveřejněna prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 02. 05. 2024 Kotomtsev Pavel

PODĚKOVÁNÍ

Chtěl bych vyjádřit svou upřímnou vděčnost všem, kteří mi pomohli a podpořili mě během psaní této bakalářské práce.

Nejprve bych rád poděkoval svému vedoucímu práce, Ing. Liboru Havlíčku Ph.D, za jeho cenné rady, trpělivost a odborné vedení v průběhu celého výzkumu a psaní. Jeho podnětné myšlenky a konstruktivní kritika mi poskytly neocenitelné směřování a povzbuzení. Děkuji také své rodině a přátelům za jejich neustálou podporu, pochopení a motivaci během celého procesu psaní. Jejich povzbuzení mi dodávalo sílu a energii k dokončení tohoto důležitého kroku v mé akademické kariéře.

Nakonec bych rád poděkoval všem ostatním, kteří mě podpořili a inspirovali svou prací, literaturou nebo osobními zkušenostmi.

Vaše podpora a pomoc mi byly nepostradatelné při dosahování tohoto cíle, a za to jsem vám nesmírně vděčný. Díky vám všem!

ANOTACE

V této bakalářské práce bude uděláno realizace manipulátoru, který bude ovládán kamerovým systémem se strojovým viděním. Cil práce realizovat algoritmus pro označeni objektu a udělat akční člen pro realizace tříděni podle požadavku.

KLÍČOVÁ SLOVA

Robot, počítač, uměla inteligence, řízeni, strojové viděni.

ANNOTATION

In this bachelor's thesis, a manipulator will be realized, which will be controlled by a camera system with machine vision. The aim of the work is to implement an algorithm for marking the object and to make an actuator for the realization of sorting according to the requirement.

KEYWORDS

Robot, computer, artificial intelligence, control, machine vision.

Obsah

SEZNAM ILUSTRACÍ

TERMINOLOGIE

Rekurze – je definice, popis nebo zobrazení určitého objektu nebo procesu uvnitř tohoto samého objektu nebo procesu, což znamená, že objekt je součástí sám sebe. Termín "rekurze" pochází z latiny a znamená "vracet se".

Aproximace – je proces nahrazení složitých matematických funkcí nebo objektů jednoduššími, ale blízkými k nim. Například když se používá kusově lineární aproximace k přibližnému popisu spojitě diferencovatelné funkce nahrazením několika lineárními úseky. Při aproximaci se upřesňují struktura a parametry systému, což často výrazně zjednodušuje řešení optimalizačních problémů. V ekonomii se aproximace často používá k zjednodušení modelování ekonomických objektů sloučením jejich charakteristik.

Aktuátor – je podtržený kloub robotu.

SEZNAM ZKRATEK A ZNAČEK

AC	střídavé napětí
A/D	analogový/digitální převodník
DC	stejnosměrné napětí
EEPROM	elektricky vymazatelná programovatelná paměť s možností čtení
GND	zem
IP	ochrana proti vniknutí
RESET	resetování programu
FLASH	nevolatilní elektricky programovatelná paměť
SRAM	statická paměť
USB	univerzální sériová sběrnice
Vin	vstupní napětí
SDK	software development kit
CGK	chebychev-grübler-kutzbachova formula
RRR	rotace – rotace – rotace
RTR	rotace – posun – rotace
DOF	počet stupňů volnosti (Degrees of Freedom)
SCARA	selective compliance assembly robot arm
HSV	převládající barva, sytost barvy, hodnota jasu (Hue,Saturation,Value)
RAM	pracovní paměť procesoru
IDLE	integrované vývojové a výukové prostředí
IDE	integrované vývojové prostředí

ÚVOD

V dnešní době umožňují technologie automatizovat většinu procesů jak v průmyslu, tak i v domácnosti. Avšak automatizace těchto procesů je momentálně nákladným procesem a aby byla výroba zisková, musí uplynout určitý čas po zavedení nových technologií, a v krátkodobé perspektivě je výhodnější využívat lidskou práci. Avšak technologie se neustále rozvíjejí a díky větší konkurenci se stávají levnějšími a pohodlnějšími. Pro mě se stalo zajímavým úkolem sestavit manipulátor pro domácí potřeby a používat ho k provádění rutinních úkolů. Na internetu je již velké množství různých konstrukcí manipulátorů a bylo rozhodnuto použít hotový rámec pro tento účel.

Cílem bakalářské práce je vytvoření robotického manipulátoru, který bude schopen rozpoznávat objekty a jejich barvy a něco s nimi dělat. Bakalářská práce zahrnuje kameru, samotný manipulátor vytištěný na 3D tiskárně, softwarové zpracování obrazu, desky a vysílače pro výměnu informací.

Aby bakalářská práce fungovala, je třeba se vypořádat s mnoha úkoly, jako je tisk dílů, nastavení a kalibrace servopohonů, instalace spojení Bluetooth modulů a psaní samotného programu pro rozpoznávání objektů.

Je zřejmé, že moderní společnost stále více směřuje k automatizaci a robotizaci různých oblastí života. Tento trend otevírá řadu možností pro využití technologií v běžném prostředí, jako je domácnost. Manipulátor pro domácí potřeby může značně usnadnit každodenní činnosti a přinést do domova další úroveň efektivity a komfortu.

Pokrok v oblasti robotiky a automatizace nejenže přináší nové možnosti pro zlepšení života, ale také otevírá dveře pro inovativní projekty a kreativní řešení v mnoha oblastech lidské činnosti. Bakalářská práce není jen o konkrétním zařízení, ale také o větším trendu přizpůsobování se technologickému pokroku a jeho využití ke zlepšení každodenního života.

1.Teoretická část

Očividně je před tím abych začít takový velký a složitý proces jako výroba manipulátora s řízením pomoci umele inteligence potřeba provést analýzu projektu který už máme v realizaci a na zaklade tech technických řešeni už mužem dělat něco svoje. Existuje hodně různých designových řešení a před tím abych vybrat nejvhodnější potřeba vědět jaký má silný a slabý strany různý konstrukce a na zaklade toho udělat vyber.

1.1 Historie vzniku

Koncept průmyslového robotického manipulátoru poprvé představil George Devol. V roce 1959 realizoval svůj nápad a vytvořil první průmyslového robota s názvem Unimate. Devol je také držitelem prvního amerického patentu na robotický manipulátor (Patent USA 2988237, 1961, s prioritou z roku 1954). Na začátku 50. let Devolovi nebylo možné upoutat pozornost k jeho nápadům. Nicméně potkal inženýra, fyzika a podnikatele Josepha Engelbergera, který jeho koncept podpořil. Společně založili společnost Unimation v roce 1962 - prvního výrobce průmyslových robotů. Ačkoli Engelberger řídil podnikání a zajišťoval financování, autorství myšlenky a obchodní značky firmy patřily Devolovi. První sériově vyráběný robot společnosti Unimation, nazvaný PUMA, se stal široce rozšířeným po celém světě. V roce 1966 se Engelberger přesunul za hranice obchodního a akademického prostředí a vystupoval s roboty Unimation v televizních pořadech, čímž se stal "otec robotiky" v očích veřejnosti. Avšak Devol a Engelberger se setkali s negativními stereotypy o robotech v americké společnosti. V té době byli roboti často spojováni s mystikou a fantazií, což ztěžovalo jejich přijetí ve společnosti. Vzhledem k nedostatku nadšení ve Spojených státech se Devol a Engelberger obrátili k Japonsku, kde byla robotika přijata s nadšením. Japonské automobilové výrobce se stali prvními, kteří používali roboty Unimate. Později japonská korporace Kawasaki získala licenci na technologie Unimate a začala vlastní výzkum a vývoj v oblasti průmyslové robotiky. To se stalo výchozím bodem pro robotickou revoluci jak v Japonsku, tak po celém světě.

1.2 Základní pojmy v kinematice robotů

Základní principy mechaniky

Mechanika, včetně obecné mechaniky, se dělí na několik disciplín.

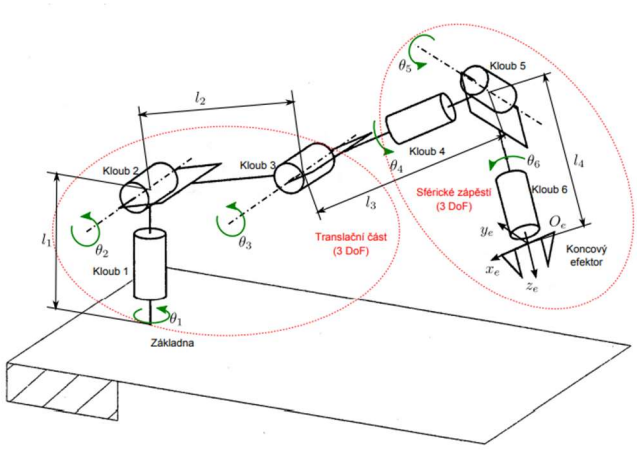
Statika zkoumá působení sil na robota v klidu a analyzuje deformace jednotlivých částí. Na rozdíl od dynamiky se zde nezkoumá samotný pohyb robota a vliv setrvačných a tlumicích sil. Klíčovým pojmem v oblasti statické mechaniky je pružnost.

Kinematika se zabývá geometrií pohybu robota a analyzuje trajektorie, po kterých se robot pohybuje, aniž by zkoumala příčiny samotného pohybu. Zde hraje klíčovou roli pojem poloha.

Dynamika se pak zaměřuje na analýzu působení sil a momentů na robota během pohybu. Všechny tyto pojmy a zákony lze aplikovat na širokou škálu mechanických zařízení a strojů.

Terminologie v kinematice robotů

Základna (base) je částí robota, která je pevně spojena se zemí. Rameno (arm) je pevná část robota, zatímco kloub (joint) umožňuje ovládaný nebo volný pohyb mezi spojenými rameny, tvořící kinematickou dvojici (kinematic pair). Chapadlo (end effector) slouží k manipulaci s předměty, zatímco příruba chapadla (gripper flange) slouží k montáži chapadla nebo dalších nástrojů. V případě šestiosých angulárních průmyslových robotů, jako je například na (obrázku 1.1), se používají kloubové souřadnice (kloub 1 až 6). kloub 1 označí otáčení základny, kloub 2 otáčení prvního ramene dopředu a dozadu, kloub 3 otáčení druhého ramene nahoru a dolů, kloub 4 otáčení zápěstí doprava a doleva, 5 otáčení zápěstí nahoru a dolů a 6 otáčení efektoru.



Obrázek 1 Antropomorfní manipulátor (Švejda 2011)

U takového robota máme k dispozici 6 úhlů natočení jednotlivých kloubů (souřadnice kloub 1 až 6), které jasně určují polohu koncového členu a ostatních členů. Posunutí a orientace koncového členu vzhledem k základně rovněž jasně definují jeho polohu, avšak může být dosaženo téže polohy více způsoby.

Kinematický řetězec je soubor ramen spojených klouby, který může být znázorněn graficky. Uzly tohoto grafu reprezentují ramena a hrany klouby. Například RRR (rotace – rotace – rotace) nebo RTR (rotace – posun – rotace), kde podtržený kloub bývá často označen jako aktivní nebo aktuátor. Pokud je rameno kinematického řetězce spojeno se zemí, nazývá se mechanismus. Otevřený kinematický řetězec je charakterizován acyklickým grafem (bez smyček), což je typické pro sériové manipulátory. Naopak smíšený kinematický řetězec obsahuje smyčky a v případě paralelního manipulátoru může obsahovat více ekvivalentních smyček.

Počet stupňů volnosti (Degrees of Freedom – DOF) představuje minimální počet nezávislých parametrů, které jednoznačně popisují analyzovaný systém. Například bod v rovině má 2 DOF (posun podél osy x a y), bod v prostoru má 3 DOF (posuny podél os x, y, z), tuhé těleso v rovině má 3 DOF (posuny a otočení kolem bodu) a v prostoru má 6 DOF (posuny a otočení kolem os x, y, z).

Počet stupňů volnosti může být redukován v závislosti na typu kinematické vazby. Pokud má robotický systém více než 6 DOF, má větší pracovní prostor, což umožňuje práci i za překážkami. Avšak výroba takového systému je složitější, má vyšší hmotnost, nižší nosnost a přesnost, a plánování trajektorií pohybu je náročnější

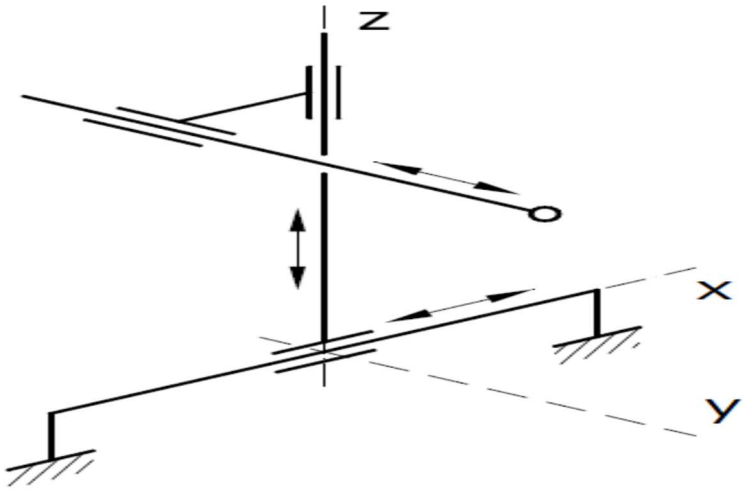
1.3 Klasifikace robotů dle kinematické struktury

Sériová kinematická struktura je popsána jako otevřený kinematický řetězec, který lze reprezentovat acyklickým grafem. Zdůrazňuje se, že každé rameno manipulátoru je spojeno s jinými rameny prostřednictvím určité vazby, přičemž koncový efektor a základna mají pouze jednu vazbu s dalšími rameny. Takové manipulátory jsou nejrozšířenější typy, mají relativně jednoduchou mechanickou konstrukci a velký pracovní prostor, ale mají nižší tuhost, vyšší hmotnost a nižší přesnost polohování.

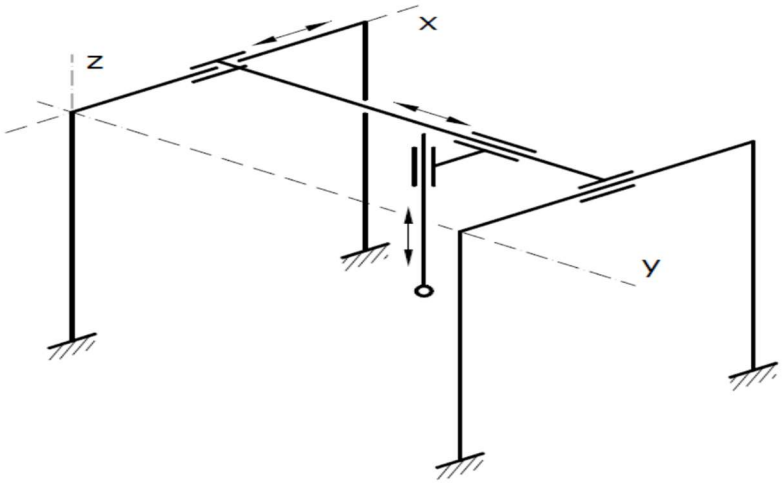
Naopak, paralelní kinematické struktury využívají uzavřený kinematický řetězec, kde je efektor spojen s bází minimálně dvěma otevřenými kinematickými řetězci. Tyto manipulátory, které byly původně výzkumným předmětem, jsou nyní často používány v průmyslu, zejména v potravinářském odvětví.

Kinematické struktury manipulátorů a robotů

V průmyslové robotice se používají kinematické řetězce s rotačními a posuvnými vazbami, označenými podle orientace pohybu. Pro určení polohy jsou potřebné 6 souřadnic. Roboty obvykle mají 6 stupňů volnosti, které ovlivňují velikost pracovního prostoru, přesnost a umístění pohonů. (Obrázek 2), Kinematické struktury jsou popsány posloupností kinematických párů od základny k efektoru. Manipulátory s kartézskou kinematickou (Obrázek 3), strukturou pracují v oblasti ve tvaru kvádru a udržují orientaci břemene. Jejich nevýhodou jsou nepříznivé vlastnosti posuvných mechanismů. Často jsou používány jako podavače a sázecí stroje.



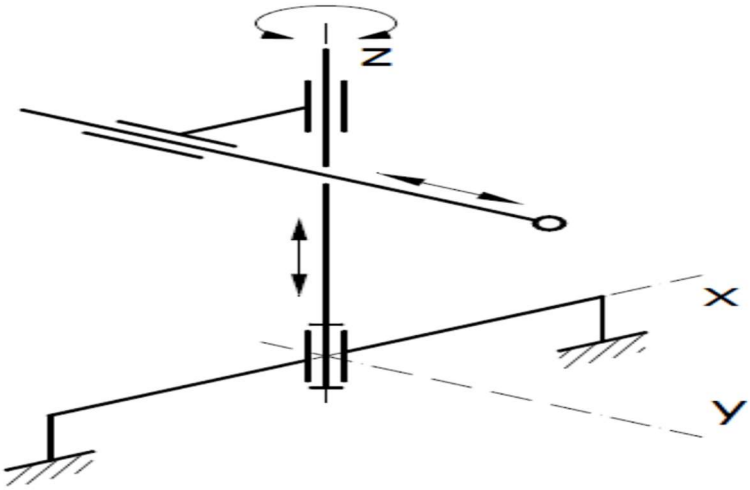
Obrázek 2 Kartézská kinematická struktura Tx, Tz, Ty (Skařupa, 2007)



Obrázek 3 Stojanové i portálové (Skařupa, 2007)

Cylindrická

Cylindrická (nebo válcová) kinematická struktura se skládá z dvou translačních a jedné rotační kinematické dvojice (RTT). Manipulátor tohoto typu (Obrázek 4) operuje v oblasti ve tvaru válcového prstence a během manipulace s břemenem dochází ke změně jeho orientace.



Obrázek 4 Cylindrická kinematická struktura Rz, Tz, Ty (Skařupa, 2007)

SCARA

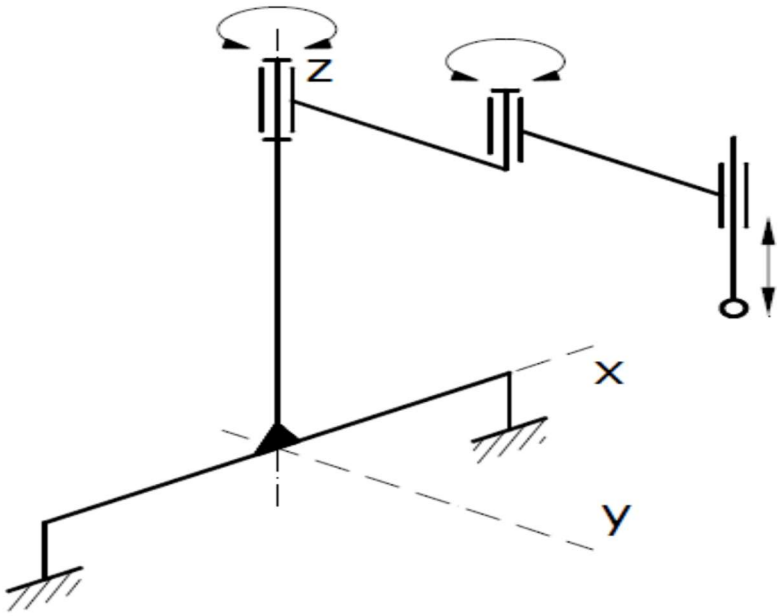
Kinematický řetězec na (Obrázek 5) se používá u manipulátorů, které se označují SCARA (Selective Compliance Assembly Robot Arm nebo Selective Compliance Articulated Robot Arm), SCARA manipulátory jsou charakterizovány velkou rychlostí a vyšší přesností pohybu kolem vertikálních os. Díky absenci pružných prvků (pohonových řemenů) v konstrukci je mechanismus SCARA charakterizován vysokou opakovatelností výsledků pohybu bez změny přesnosti. To znamená, že SCARA roboty mohou provádět po sobě jdoucí identické operace bez nejmenších odchylek. Jsou kompaktní a snadno se montují, což usnadňuje práci v

omezených prostředích. Díky flexibilitě v manipulaci s objekty pod libovolným úhlem a absenci pružných prvků dosahují konzistentních výsledků.

Většina mechanismů SCARA může být instalována na libovolnou plochu (stěna, strop, podlaha) bez změny jejich provozních vlastností. Tato výhoda je široce využívána v provozních prostorách s omezeným objemem.

Pro mechaniku navrženou podle principu SCARA je typická nehomogenita rozlišovací schopnosti pohybu v rovině X-Y. Pro mechanismy SCARA se hovoří o gradientu rozlišovací schopnosti v dané rovině. Maximální přesnost (nejmenší absolutní chyba a největší rozlišovací schopnost) je pozorována na začátku souřadnic (v centru mechanismu). S postupným vzdalováním od středu (s nárůstem délky páku, tj. prodloužením "ruky" SCARA), se rozlišovací schopnost zhoršuje.

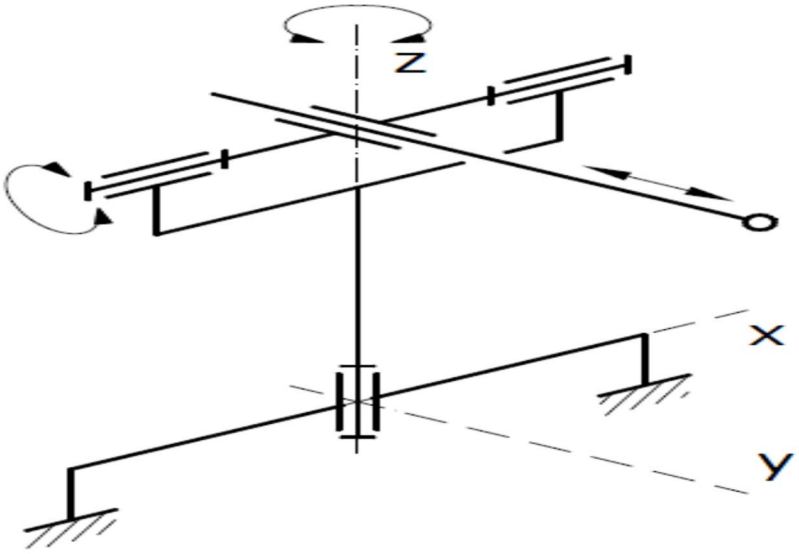
Přípustné zatížení se skládá ze dvou složek – hmotnosti pracovního nástroje a hmotnosti zátěže (nebo úsilí působícího v pracovní oblasti mechanismu). Při použití mechaniky SCARA je zatížení připojeno k pracovní oblasti na konci prodlouženého pracovního ramene mechanismu (prodloužené "ruky" robota). To vede k určitým omezením v zatížení a nutnosti zvýšit pevnost a tuhost prvků mechanismu.



Obrázek 5 Kinematická struktura Rz, Rz, Tz (Skařupa, 2007)

Sférická

Sférická (kulová) kinematická struktura je tvořena dvěma rotačními a jednou translační kinematickou dvojicí (RRT), podobně jako u systémů SCARA, ale rotace jsou v tomto případě prováděny podle různých os, (Obrázek 6)

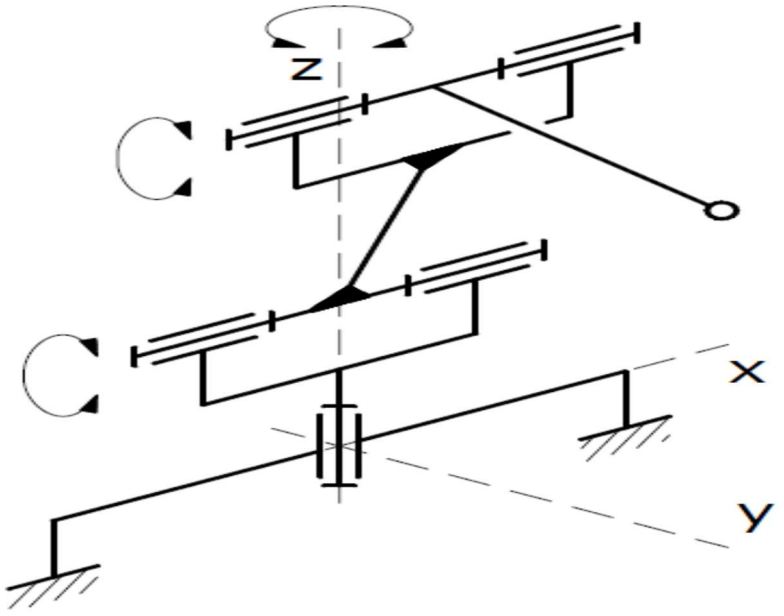


Obrázek 6 Sférická kinematická struktura Rz, Rx, Ty (Skařupa, 2007)

U manipulátorů se sférickou kinematickou strukturou dochází ke změně orientace objektu. Pracovním prostorem je v tomto případě kulový vrchlík. Nejrozšířenější aplikační oblastí jsou svařovací roboty.

Angulární

Angulární kinematická struktura (obrázek 7) umožňuje manipulaci s objekty ve všech směrech kolem pevného středu. Používá pouze rotační vazby pro nižší cenu a vyšší spolehlivost. I když má dobrou dynamiku a univerzální využití, opakovatelná přesnost klesá s rostoucí délkou ramen. Pro rozšíření pracovního prostoru lze angulární manipulátory osadit na portál nebo pojezd, nebo umístit výrobek na otočný stůl.



Obrázek 7 Angulární kinematická struktura Rz, Rx, Rx (Skařupa, 2007)

1.4 Přehled algoritmů umělé inteligence v kontextu řízení manipulátorů

OpenCV

V bakalářské práce bylo implementováno řízení pomocí knihovny OpenCV, což je knihovna počítačového vidění.

OpenCV, zkratka pro Open Source Computer Vision Library, je široce používaná open-source knihovna pro úkoly počítačového vidění. Je dostupná ze zdrojů [12] a je psána převážně v jazycích C a C++, s podporou pro platformy Linux, Windows a Mac OS X. OpenCV je navržen s důrazem na výpočetní efektivitu a aplikace v reálném čase.

Knihovna je neustále vyvíjena, s aktivními rozhraními dostupnými pro jazyky jako Python, Ruby, Matlab a další. OpenCV využívá optimalizovaný C kód a může využít vícejádrové procesory pro zlepšení výkonu. Kromě toho, pro další optimalizaci na architekturách Intel mohou uživatelé využívat knihovny Intel's Integrated Performance Primitives (IPP), které nabízejí optimalizované rutiny nízko úrovňových operací v různých oblastech algoritmů. OpenCV se bezproblémově integruje s knihovnami IPP za běhu, pokud jsou nainstalovány.

OpenCV si klade za cíl poskytnout uživatelsky přívětivou infrastrukturu pro počítačové vidění, umožňující vývojářům rychle vytvářet sofistikované aplikace pro vidění. S více než 500 funkcemi pokrývajícími různé oblasti vidění, jako jsou inspekce produktů, lékařské zobrazování, bezpečnost, uživatelské rozhraní, kalibrace kamer, stereo vidění a robotika, nabízí OpenCV rozsáhlé možnosti.

Kromě toho, vzhledem k uznání symbiotického vztahu mezi počítačovým viděním a strojovým učením, OpenCV obsahuje knihovnu strojového učení (MLL). Tato podknihovna se zaměřuje na statistické rozpoznávání vzorů a shlukování a poskytuje robustní podporu pro základní úkoly v oblasti vidění, přičemž zůstává dostatečně všestranná k řešení různých výzev strojového učení.

MobileNetV2

Dále byla použita neuronová síť MobileNetV2, která využívá hloubkově oddělitelné konvoluce. Model MobileNet je založen na faktorizovaných konvolucích, které rozdělují běžnou konvoluci na hloubkovou a bodovou konvoluci. Hloubková konvoluce používá jeden filtr na každý vstupní kanál, zatímco bodová konvoluce sloučí výstupy hloubkové konvoluce. Tento přístup výrazně redukuje výpočetní nároky a velikost modelu.

Běžná konvoluční vrstva filtruje a sloučí vstupy v jednom kroku, což vede k vysokému výpočetnímu nákladu. Model MobileNet využívá hloubkově oddělitelné konvoluce, která tento proces rozděluje do dvou vrstev, což vede k výrazné redukci výpočetní náročnosti.

Hloubková konvoluce se skládá z jednoho filtru na každý vstupní kanál a je následována bodovou konvolucí, která vytváří lineární kombinaci výstupů. Tento model používá dávkovou

normalizaci a ReLU nelinearity. Použitím hloubkově oddělitelných konvolucí dochází ke snížení výpočetní náročnosti.

MobileNet 3x3 hloubkově separabilní konvoluce

Zvyšují efektivitu s minimálním snížením přesnosti. Hloubkově separabilní konvoluce je technika používaná v konvolučních neuronových sítích pro redukci výpočetní náročnosti a parametrů modelu.

Klasická konvoluce provádí operaci průchodu filtru přes všechny kanály vstupního obrázku. Hloubkově separabilní konvoluce rozděluje tento proces do dvou kroků:

Hloubková konvoluce (Depthwise convolution): V této fázi je použit samostatný konvoluční filtr na každý kanál vstupního obrázku. Místo jednoho filtru pro každý vstupní kanál jsou použity oddělené filtry pro každý kanál.

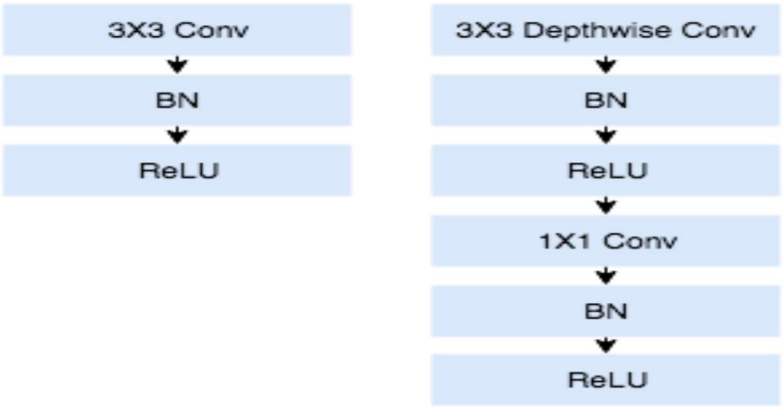
Konvoluce 1x1 (Pointwise convolution): Výstupy z předchozí fáze jsou následně kombinovány pomocí konvoluce 1x1, která sloučí výstupy do jednoho výstupního obrázku.

Tímto způsobem hloubkově separabilní konvoluce výrazně snižuje výpočetní náročnost a počet parametrů sítě, aniž by příliš ovlivnila výkonnost modelu.

Například, 3x3 hloubkově separabilní konvoluce použitá v MobileNetu znamená, že pro každý pixel ve vstupním obrázku se nejprve aplikuje 3x3 hloubková konvoluce a poté konvoluce 1x1 pro sloučení výstupů. Tímto způsobem MobileNet dosahuje vysoké účinnosti při zachování přijatelné úrovně přesnosti.

Struktura sítě MobileNet je založena na hloubkově separabilních konvolucích s výjimkou první vrstvy, která je plná konvoluce. Všechny vrstvy jsou následovány dávkovou normalizací a ReLU nelinearitou.

Síť MobileNet má 28 vrstev, přičemž hloubkové a bodové konvoluce jsou počítány jako samostatné vrstvy.



Obrázek 8 Pravidelná konvoluční vrstva s batchnorm a ReLU. Vpravo: hloubkově oddělitelné konvoluce s hloubkovými a bodovými vrstvami následované dávkovou normou a ReLU (Indonesian Journal of Electrical Engineering and Computer Science,2019)

Batch Normalization (BatchNorm)

Je technika používaná v neuronových sítích, která pomáhá zlepšit stabilitu a rychlost učení tím, že normalizuje každou vrstvu v neuronové síti tak, aby měla průměr nula a rozptyl jedna. Tímto způsobem BatchNorm pomáhá předejít problémům jako je mizící/explodující gradient a umožňuje rychlejší učení sítě.

Základní myšlenka za BatchNorm spočívá v tom, že během tréninku se normalizují vstupy každé vrstvy podle statistických vlastností (průměr a rozptyl) všech aktivací v minulém minibatchi. Tím se dosáhne toho, že každá vrstva bude pracovat s podobně velkými čísly, což způsobuje, že se učení sítě stává stabilnějším.

Konkrétně se Batch Normalization skládá ze dvou kroků:

Normalizace: Vstupy do každé vrstvy jsou normalizovány odečtením průměru minibatche a následným dělením standardní odchylkou minibatche.

Škálování a posunutí: Normalizované hodnoty jsou následně zváženy a posunuty pomocí parametrů, které se učí během tréninku.

Použití Batch Normalization pomáhá urychlit učení neuronových sítí, umožňuje použití vyšších learning rate, což může vést k rychlejší konvergenci a zlepšení výkonnosti sítě. Taktéž může fungovat jako regulátor, což umožňuje lepší generalizaci a snižuje přesnost na validačních datech.

ReLU (Rectified Linear Unit)

Je aktivační funkce používaná v neuronových sítích. Je definována jednoduše jako max (0, x), kde x je vstupní hodnota.

Matematicky lze ReLU funkci definovat jako (1.1):

$$f(x)=\max(0, x) \tag{1.1}$$

ReLu funkce přiřadí nule všechny negativní hodnoty a ponechá všechny kladné hodnoty nezměněné. ReLu je oblíbená pro svou jednoduchost a efektivitu. Má tendenci urychlovat učení neuronové sítě tím, že umožňuje průchod signálu bez jakéhokoliv transformačního zpoždění, které by bylo spojeno s jinými aktivačními funkcemi jako je například sigmoida nebo tanh.

Jedna z nevýhod ReLu je však v tom, že může vést k "mrtvým neuronům" (neurony, které nevyjadřují žádnou aktivitu, protože mají negativní váhy). To znamená, že pokud je výstup z neuronu ReLu v tréninkové fázi vždy záporný, gradient váhy spojené s tímto neuronem bude vždy nula a váhy se nebudou aktualizovat. Tento problém může být řešen použitím dalších variant ReLu, jako je například Leaky ReLU, která umožňuje malé záporné hodnoty, nebo Exponential Linear Unit (ELU), která má některé výhody ve výkonnosti a stability.

Operace formless light matrix (bez tvaru světelné matice) jsou obvykle ne rychlejší než husté matice, dokud nenastane velmi vysoká úroveň řídkosti. Tato struktura modelu přenáší výpočty do hustých 1×1 konvolucí, které lze provést s vysoce optimalizovanými funkcemi obecného

násobení matic (GEMM). 1 × 1 konvoluce nevyžadují počáteční přeuspořádání v paměti a lze je implementovat přímo pomocí GEMM.

Modely MobileNet byly trénovány v TensorFlow pomocí RMSprop s asynchronním gradientním sestupem. Oproti trénování velkých modelů se používá méně regularizace a technik augmentace dat. Při trénování MobileNetů se používají vedlejší hlavy nebo vyhlazování značek a snižuje se množství deformací obrazu omezením velikosti malých výřezů.

Je důležité dát velmi málo nebo žádný pokles váhy na hloubkové filtry, protože jich je tak málo.

1.5 Barevné masky

Byl stanoven úkol určit barvu a byl vyřešen pomocí tohoto postupu. Při určování barev objektu bylo vytvářeno dvě masky pro obě barvy.

Určení rozsahu červené barvy v HSV

```
lower_red = np.array([0, 100, 100])
upper_red = np.array([10, 255, 255])
mask_red1 = cv2.inRange(hsv, lower_red, upper_red)
lower_red = np.array([160, 100, 100])
upper_red = np.array([180, 255, 255])
mask_red2 = cv2.inRange(hsv, lower_red, upper_red)
```

Kombinujeme masky pro červenou barvu.

```
mask_red = cv2.bitwise_or(mask_red1, mask_red2)
```

Určete rozsah modré barvy v HSV.

```
lower_blue = np.array([100, 100, 100])
upper_blue = np.array([140, 255, 255])
mask_blue=cv2.inRange(hsv,lower_blue,upper_blue)
```

Kód obsahuje dvě masky pro červenou barvu, protože červená se nachází na hranici barevného kruhu v prostoru HSV. V tomto prostoru jsou barvy reprezentovány jako kruh, takže některé odstíny červené mohou být blíže k jedné hranici (odstíny od 0 do 10) a jiné k druhé (odstíny od 160 do 180). Aby byly zahrnuty všechny odstíny červené, používají se dvě masky.

1.6 Algoritmy pro vyhledávání objektů

Tento úsek kódu provádí detekci čtverců na obrázku a přidává obdélníky kolem nalezených čtverců. Kromě toho přidává popisky označující barvu každého čtverce. Algoritmus funguje následujícím způsobem:

```
contours, _ = cv2.findContours(mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
```

contours jsou kontury nalezené pomocí funkce findContours.

max_contour = max(contours, key=cv2.contourArea):

Tato řádka hledá největší obrys (konturu) v obraze.

cv2.contourArea vrací plochu kontury. Funkce max vybere konturu s největší plochou.

x, y, w, h = cv2.boundingRect(max_contour):

Tato řádka vypočítá ohraničující obdélník (bounding box) pro největší nalezený obrys.

x, y jsou souřadnice levého horního rohu obdélníka.

w je šířka obdélníka a h je výška.

cv2.rectangle(snapshot, (x, y), (x+w, y+h), (0, 255, 0), 2):

Tato řádka kreslí obdélník kolem nalezeného objektu na snímku.

(x, y) jsou souřadnice levého horního rohu obdélníka.

(x+w, y+h) jsou souřadnice pravého dolního rohu obdélníka.

(0, 255, 0) je barva obdélníka (zelená).

2 je tloušťka obrysu obdélníka.

cv2.putText(snapshot, 'Čtverec', (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, (0, 255, 0), 2):

Tato řádka přidává popisek "Čtverec" nad obdélníkem.

'Cetverec' je text popisku.

(x, y - 10) jsou souřadnice, kde se text zobrazí.

cv2.FONT_HERSHEY_SIMPLEX je font textu.

0.9 je velikost textu.

(0, 255, 0) je barva textu (zde zelená).

2 je tloušťka písma.

color_points = [(x+w//4, y+h//4), (x+w//2, y+h//2), (x+3w//4, y+3h//4)]:

Tato řádka definuje body uvnitř obdélníka, které budou kontrolovány na přítomnost červené nebo modré barvy.

for point in color_points:

Tato část kódu projde všechny body uvnitř obdélníka.

if mask_red[point[1], point[0]] == 255:

Tato podmínka kontroluje, zda je bod červený.

mask_red je binární maska, která ukazuje červené oblasti na snímku.

[point[1], point[0]] jsou souřadnice bodu.

if mask_blue[point[1], point[0]] == 255:

Tato podmínka kontroluje, zda je bod modrý.

mask_blue je binární maska, která ukazuje modré oblasti na snímku.

[point[1], point[0]] jsou souřadnice bodu.

if red_count > blue_count:

Pokud je počet červených bodů větší než počet modrých bodů, barva objektu se označí jako červená.

elif blue_count > red_count:

Pokud je počet modrých bodů větší než počet červených bodů, barva objektu se označí jako modrá.

else:

Pokud je počet červených bodů roven počtu modrých bodů, barva objektu se označí jako neznámá.

cv2.putText(snapshot, f'Color: {color_label}',(x,y+h+30), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 255, 0), 2):

Tato řádka přidává popisek barvy objektu pod obdélník.

f'Color: {color_label}' je text popisku, který obsahuje označení barvy.

(x, y+h+30) jsou souřadnice, kde se text zobrazí.

cv2.FONT_HERSHEY_SIMPLEX je font textu.

0.7 je velikost textu.

(0, 255, 0) je barva textu (zde zelená).

2 je tloušťka písma.

if color_label == 'cerveny': ser.write(b'1') elif color_label == 'modry': ser.write(b'0'):

Tato část kódu odesílá signál přes sériovou linku na základě barvy objektu.

Pokud je objekt červený, odesílá se b'1'.

Pokud je objekt modrý, odesílá se b'0'.

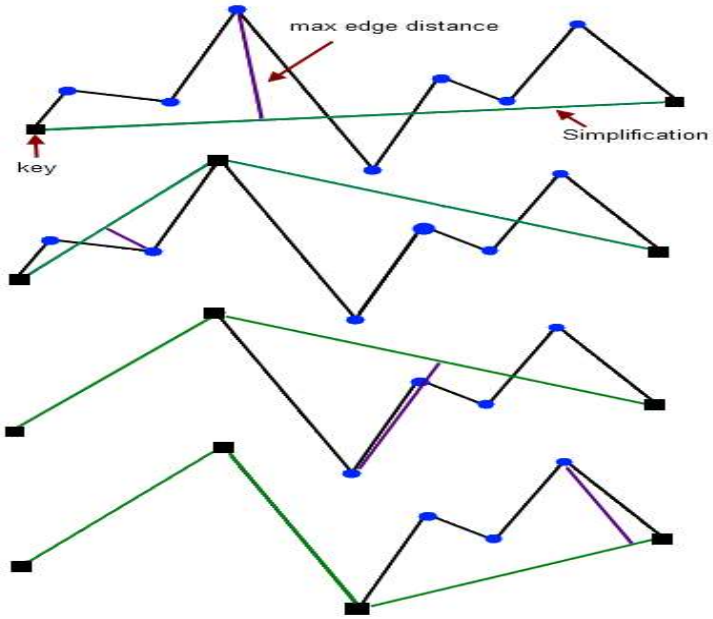
cv2.imshow("Snapshot", snapshot):

Tato řádka zobrazí upravený snímek s obdélníkem a popisky.

metody vybraný pomoci zdrojů [23] a [24].

Algoritmus Douglase-Peuckera

Metoda cv2.CHAIN_APPROX_SIMPLE implementuje algoritmus Douglase-Peuckera, který aproximuje konturu pomocí zjednodušení polygonu zachováním pouze důležitých vrcholů. Tento algoritmus snižuje počet bodů, které tvoří konturu, a tím zlepšuje výkon algoritmů zpracování obrazu.Spočívá v tom, že pro danou lomenou čáru, která aproximuje křivku, se vytvoří lomená čára s menším počtem bodů. Algoritmus určuje odchylku, která je vypočítána jako maximální vzdálenost mezi původní a zjednodušenou křivkou. Zjednodušená křivka se skládá z podmnožiny bodů, které jsou vybrány z původní křivky. Původní křivka je zobrazena jako uspořádaná sada bodů nebo linií a daná vzdálenost ε > 0. Počáteční křivka je zobrazena na hoře , zjednodušená úplně dole (obrázek 9).



Obrázek 9 Počáteční křivka a zjednodušená (psimpl Copyright ,2010-2011)

Algoritmus rekurzivně dělí úsečku. Vstupem algoritmu jsou souřadnice všech bodů mezi prvním a posledním bodem. První a poslední bod zůstávají nezměněny. Poté algoritmus hledá bod, který je nejdále od úsečky spojující první a poslední bod. Pokud je bod ve vzdálenosti menší než ε, pak všechny body, které ještě nebyly označeny k zachování, mohou být odebrány ze sady a výsledná přímka vyhlazuje křivku s přesností ne menší než ε. Pokud je vzdálenost větší než ε, pak algoritmus rekurzivně volá sám sebe s množinou bodů od počátečního bodu po tento bod a od tohoto bodu po koncový bod (což znamená, že tento bod bude označen k zachování). Po dokončení všech rekurzivních volání je výstupní lomená čára postavena pouze z těch bodů, které byly označeny k zachování [13].

1.7 Analýza dostupných mikropočítačů a vývojových kitů pro řízení zařízení

Analýza dostupných mikropočítačů a vývojových kitů pro řízení zařízení je klíčovým krokem při navrhování a implementaci řídicí jednotky pro robotický manipulátor s kamerovým

systémem a strojovým viděním. Analýza byla udělaná podle toho abych splnilo funkcionalitu zařízeny.

Pro mikropočítač potřeba abych měl:

Parametry jako frekvence procesoru, paměťové kapacity (RAM, flash) protože musit vypracovávat data v reálnem čase, a dostatečný počet I/O pinů, pro to abych bylo možné připojit akční členy. Taky bylo bz dobry mít analogovy piny pro zapojeny některých senzoru prý dalších rozvojů.

Podpora komunikace (UART, SPI, I2C, Ethernet, Wi-Fi, Bluetooth). Protože bude používáno spojeni přes Bluetooth a taky potřeba mít prostor pro další modifikace.

Dostupnost periférií pro senzory a aktuátory.

Dostupnost vývojových kitů: Existují vývojové kity obsahující mikropočítače spolu s dalšími perifériemi a rozhraními což do cela důležitá věc pro to abych realizace byla minimálně problematická. Vývojové kity mohou urychlit vývoj a testování řídicí jednotky.

Podpora softwarových nástrojů a vývojového prostředí:

Existence SDK (Software Development Kit) a knihoven pro rychlý vývoj aplikací.

Podpora různých programovacích jazyků (C, C++, Python) a vývojových prostředí (Arduino IDE, PlatformIO, MPLAB X, Atmel Studio).

Cenová dostupnost mikropočítačů a vývojových kitů pro studenty a výzkumníky tak jak celková hodna peněz musit být kolem 5-7 tisíc korun.

Možnost zakoupení vývojových kitů a mikropočítačů na trhu a jejich dostupnost v různých regionech pro různý lidi z celého světa.

Komunita a podpora taky potřebná a pro to vyber musí být směrován ohledné na dostupnost

aktivní komunity uživatelů a vývojářů, kteří sdílejí znalosti, projekty a podporu.

Přítomnost online dokumentace a tutoriálů pro vývoj s danými mikropočítači a vývojovými kity.

Možnost rozšiřování funkcionality mikropočítačů pomocí externích modulů a periférií.

Dostupnost rozšiřujících desek (shields) a modulů pro různé účely (senzory, komunikační rozhraní, displeje).

Po provádění analýzy trhu takových zařízeni bylo vybráno pro realizace bakalářské práce využít dvě desky arduino který budou mezi sobou komunikovat, protože při realizace vyšlo že blutus modul HC=05 komunikuje s zařízeni jenom na androidů, ale bylo potřeba realizace komunikace s počítačem a kvůli tomu bylo využíváno řešeni použit dvě desky jako Master aduino nano a jako Slave arduino uno.

2. Konstrukční návrh

2.1 Popis návrhu robotického manipulátoru a kamerového systému

Konstrukce manipulátoru antropomorfní manipulátor se sférickým zápěstím byla vybrána kvůli velké flexibilitě a pohyblivosti. 3D model tohoto konstrukce byla převzata od projektu Arduino Robot [14]. Bylo rozhodnuto použít hotový rámec, protože bakalářská práce je poměrně rozsáhlá a vyžaduje mnoho časových prostředků.

Kamera

Webkamera kvalitou rozlišení (1280×720 px) a 30 fps. Kamera do cela příjemná ji úhel záběru 52° a taky má v sobě mikrofon, tlačítko pro snímek až 8 Mpx. Velky bonus pro to abych bylo pěkné nastavena poloha flexibilní stojánek • úchyt na notebook • automatické balancování bílé barvy. Konektor USB 2.0.Velikost kamery není velká, jenom 81 × 70 × 62 mm a váha 88 gramů.

Obrázek 10 Webkamera Trust Trino HD

V bakalářské práce bylo zvoleno toto řešení nákupu této kamery kvůli nízké ceně. Hlavním úkolem zařízení je zajistit barevný obraz a dobře viditelný objekt.

2.2 Výběr konstrukčních prvků a technologií pro realizaci zařízení

MG996R Servo Motor

Obrázek 11 MG996R Servo Motor

Napájecí napětí: 4,8V - 6,0V a provozní proud je 500 mA. Úhel otáčení: 0° až 180° víc než 180 není potřeba protože nemáme v manipulátoru ramen který bych potřebovali otečeni na 360 stupňů. Servo schopno dosáhnout rychlosti (6,0V): 0.14 sec/60° a točivého moment při (6,0V) v 11 kg*cm. Teplotní rozsah: 0 °C až 55 °C. Délka kabelu: 30 cm kabel s 3 pinovou zásuvkou typu "S". Hmotnost: 55 g. Rozměry: 40,7 mm x 19,7 mm x 42,9 mm

Hlavní nosné servopohony, které budou schopné unést hmotnost objektu, kvůli tomu musí mít dobru mocnost a s tím pohony schopny zvládnout, protože jejich rychlost (4,8V): 0.17 sec/60° a točivý moment: (4,8V) 9 kg*cm, pohony mají na sobě celkovou hmotu konstrukce. Potřeba ve třech kusech, ale je lepší vzít je s rezervou kvůli možné vadě z výroby.

SG90 Micro Servo Motor

Obrázek 12 SG90 Micro Servo Motor

Digitální servo s nylonovými převody, ovládané přes PWM signál. Pracovní rozsah napětí 3,5 ÷ 6 V. Rychlost náměru: 0,12 s / 60 stupňů (4,8 V); 0,10 s / 60 stupňů (6,0 V). Kroutící moment: 1,6 kg·cm (4,8 V). Hmotnost: 9 g. Rozměry: 23x12, 2x29mm.V balení obvykle má různé nástavce pro uchycení předmětů a šrouby.

Vedlejší servopohony vybraný kvůli jejich použití, v RC modelech, robotech či různorodých aplikacích, velikosti a ceně. Potřeba také tři kusy.

Bluetooth modul

Bluetooth modul HC-05 pro komunikace mezi sobou (Obrázek 1.14). Modul podporuje protokoly Bluetooth verse 2.0 a EDR. Má v sobě sériové rozhraní což znamená, že přenos dat funguje tak že jeden bit posíláme postupně po druhému přes jeden komunikační kanál. Komunikace nastavována na přenos až 9600 (bpn) což znamená schopen předávat 9600 bit za jednu vteřinu. Signál rozšiřuje se na vzdálenost 10 metrů. Sam modul docela malý 3,2 x 1,6 x 0,3 cm, a je možnost jeho krásné schovat do nějakého těla. Modul má piny

STATE: Stav jako Master nebo Slave

VCC: Napájení

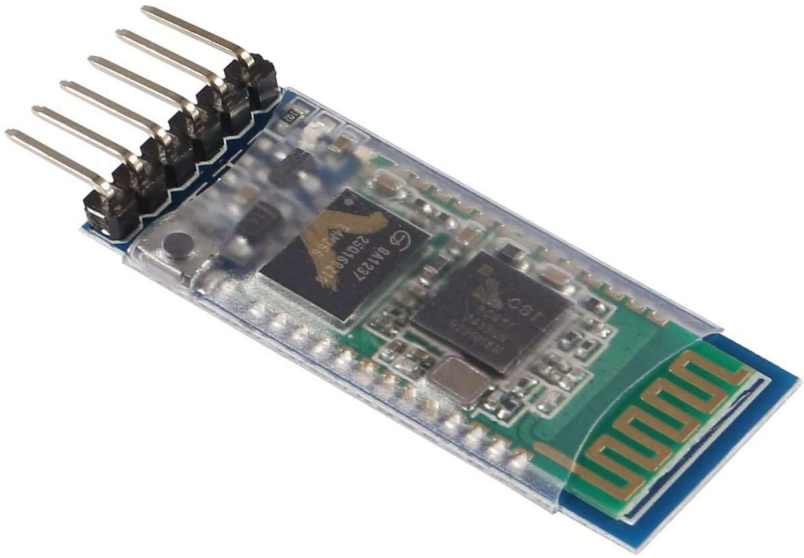
GND: Zemný kontakt

TXD: Vysílací pin

RXD: Přijímací pin

KEY: Klíč (Může sloužit k nastavení režimu)

Indikaci provozu modulu lze kontrolovat pomocí LED na přední části desky. HC-05 schopen pracovat na platformách jako Arduino a Raspberry Pi a pro to je možnost využit ho v jiných projektech a různých deskách. Napájený potřeba mít od 4,5 – do 6 voltu. Hlavní čipy: 29LV800, BC417 .29LV800: Jedná se o typ flash paměti. Konkrétně 29LV800 je čip typu NOR Flash s kapacitou 8 megabitů. Tato paměť slouží k ukládání programového kódu, dat nebo konfigurace. BC417: Tento čip je často používán pro implementaci Bluetooth funkcionality. Je to Bluetooth 2.0 EDR (Enhanced Data Rate), který poskytuje bezdrátovou komunikaci v krátkém dosahu. BC417 je součástí Bluetooth modulu HC-05 a umožňuje modulu komunikovat s jinými zařízeními pomocí Bluetooth.



Obrázek 13 Bluetooth modul HC-05

Při nákupu Bluetooth modulu je důležité, aby moduly podporovaly přechod do režimu AT příkazů, abych bylo možné nastavit pro vzájemnou výměnu informací. V opačném případě naprogramovat není možné, ne pomůže ani připojit při nuceném napájení na pinu. V bakalářské práce budou potřeba 2 kusy.

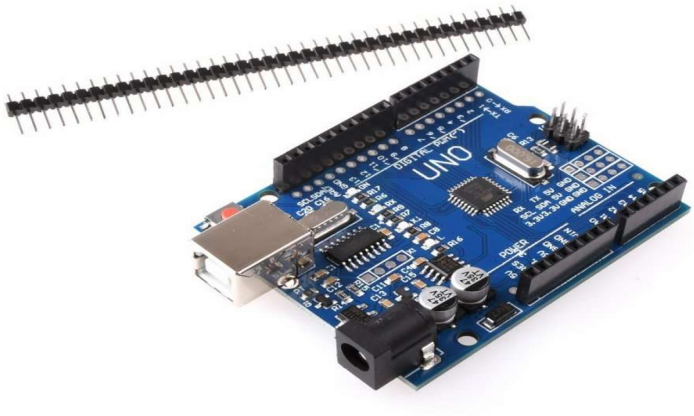
Napájecí zdroj zásuvkový 5 V, 2 A, 4.0x1.75 mm

Vstupní napětí 100–240 V střídového proudu 50/60 Hz, na výstupu 5 V stejnosměrného proudu a výstupného proudu 2 A. Výkon 10 W. Délka kabelu 100 cm. Konektor 4.0x1.7 mm. Krytí IP22, tech parametru bude dost pro splníváni úkolu napájení manipulátoru.

Obrázek 14 Napájecí zdroj zásuvkový

Zdroj napájení byl vybrán na základě původního projektu. Konkrétně v bakalářské práce je doporučen zdroj napětí od 7 do 12 voltů, ale lze ho napájet i z tohoto.

Arduino UNO



Obrázek 15 Arduino UNO R3, ATmega328P

Vývojová deska má procesor s frekvencí CPU až 20MHz, s FLASH pamětí o velikosti 32KB, EEPROM pamětí o velikosti 1KB a SRAM pamětí o velikosti 2KB. Nabízí 14 digitálních vstupů/výstupů, 6 kanálů 10-bitového analogově-digitálního převodníku a 6 PWM kanálů. Dále obsahuje 2 8-bitové časovače s odděleným Prescaler + Compare Mode, 1 16-bitový časovač s odděleným Prescaler + Compare Mode + Capture Mode a 1 programovatelný Watchdog časovač s odděleným integrovaným oscilátorem. K dispozici je také 1 USART, 1 SPI, 1 I2C, 1 analogový komparátor a 1 obvod reálného času – RTC s odděleným oscilátorem. Deska umožňuje interrupt a Wake-up při změně úrovně na pinu.

Oscilátory, reset a správa napájení zahrnují možnost napájení v rozmezí 1.8–5.5V, reset při zapnutí napájení (Power-on Reset) a programovatelný detektor nízkého napětí (Brown-out). Externí až 20MHz oscilátor pro CPU, interní 8MHz oscilátor pro CPU a interní oscilátor pro RTC jsou také k dispozici. Deska nabízí režimy nízké spotřeby včetně Idle, ADC Noise Reduction, Power-save, Power-down, Standby a Extended Standby.

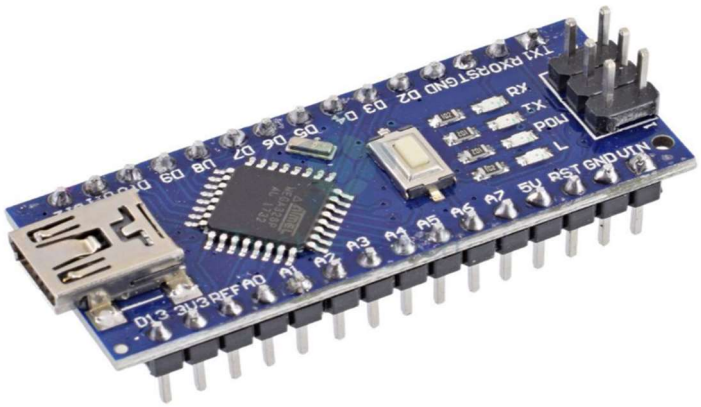
Vývojová deska má stejné rozměry a rozložení vstupních/výstupních konektorů jako originální Arduino UNO R3. Je možné ji napájet v rozmezí 6 – 12V pomocí napájecího souosého konektoru 5,5/2,1. Maximální proudový odběr je 800mA pro zdroj 5V a 180mA pro zdroj 3,3V. Obsahuje integrovaný ISP programátor s převodníkem CH340G a USB-B konektorem, stejně jako integrovaný ICSP konektor pro externí programátor. Deska má resetovací tlačítko, 16MHz krystal a integrované LED pro RX, TX a napájení. Integrovaná LED L je ovládaná programově. Možnost napájení je možná pomocí USB konektoru, napájecího konektoru nebo napájecích pinů. Celková váha desky je 22 g.

Resetovací tlačítko: Slouží k resetování programu v případě zacyklení a je označeno nápisem RESET.USB konektor: Typicky typu B, slouží k možnosti programování a napájení desky.

Napájecí konektor: Umožňuje napájení Arduino desky, pokud není napájena skrze USB konektor.ICSP hlavice: Používá se k externímu programování USB-serial převodníku.

USB-serial převodník: Zajišťuje komunikaci hlavního čipu s počítačem. Indikační LED diody (L, Rx, Tx): L značí výstup číslo 13, zatímco Rx a Tx signalizují sériovou komunikaci. Čip desky: Hlavní procesor desky. Indikační LED dioda (ON): Signalizuje připojení zařízení. ICSP hlavice (druhá): Také se využívá pro externí programování a některé shieldy. Digitální piny: Některé umožňují PWM modulaci. Napájecí piny Arduina (IOREF, RESET, 3,3 V, 5 V, GND, GND, Vin). Analogové vstupy. U různých typů desek se může umístění tlačítka restartu lišit, stejně jako typ USB konektoru. ICSP hlavici většinou běžní uživatelé nevyužívají, a u některých modelů desek není USB-seriál převodník přítomen. [27]

ARDUINO NANO



Obrázek 16 Arduino Nano V3.0

Vývojová deska Arduino Nano je postavena na čipu ATmega328 s frekvencí 16 MHz a nabízí kompaktní rozměry 45 x 19 mm, což ji skvěle hodí do konstrukcí Mastera. Integrovaný USB port a integrovaný, což zvětšuje celkové provedení. Díky tomu odpadá potřeba externího USB-Serial převodníku, což usnadňuje použití a připojení k počítači umožňuje snadné nahrávání

programů na desku a zároveň poskytuje napájecí napětí 5 V. Deska disponuje sériovým rozhraním včetně UART, SPI a I2C.

Co se týče portů, deska Arduino Nano nabízí 14 pinů I/O, 8 analogových vstupů a 6 PWM výstupů.

2.3 Specifikace použitých materiálů a komponent

Creality ENDER 3

3D tiskárna FDM, tiskový prostor 220 mm×220 mm×250 mm, tiskový materiál 1.75mm ABS, ASA, FLEX, Nylon, PETG, PLA, PVA, TUP a Wood, tloušťka vrstvy 0.1mm-0.4mm, rychlost tisku 50mm/s, možnost tisku: PC přes USB 2.0 a SD kartu. Pro tisk na 3D tiskárně byl využit rozsah pro PLA je 190 °C až 220 °C.

3D tisk vyžaduje určité dovednosti k používání, a je doporučeno se seznámit s technickými parametry vaší tiskárny a základy 3D tisku před zahájením tisku, aby byl proces co nejefektivnější a nemuseli jste opakovaně tisknout díly. Je to skutečně důležité, protože proces tisku trvá poměrně dlouho a není vhodné plýtvat časem a materiálem. Výška tisku vrstvy 0,2 mm, (Příloha 2) došlo k ucpání extruderu, proto je při tisku důležité, aby byla tryska čistá.

Při montáži součástek je také nezbytné mít nástroje. K dispozici musí být sada šroubováků malého průměru pro práci s elektronikou. Při realizaci byla použita sada šroubováků Ya Xun YX-8022 C, [6] ale vhodná je jakákoli sada s dostatečně malým průměrem. Stejně tak byl použit multimetr [7]. Stejně jako v případě šroubováku není důležité, jaký konkrétní nástroj to je, hlavní je, aby bylo možné měřit napětí a proud a drátky [21] pro zapojeny všech komponentu.

3. Experimentální část

3.1 Popis postupu implementace firmware a software

Pro software byly využity Arduino IDE [22] a IDLE (python 3.11 64-bit) [23]

Téhle prostředí splnili všechny požadavky a pro ne využíval jsem nějaký externí prostředí. Byla provedena instalace vývojového prostředí a nastavení knihoven a vizuálního zobrazení kódu při psaní.

Taky byl prováděn vyber algoritmu pro označeni objektu a metoda pro označeni barvy objektu a pro to hledaní knihoven. Při hledání vhodného konceptu práce byl napsán program, který vytvořil okno, ve kterém by uživatel mohl stisknout mezerník a pořídit snímek. Původně bylo plánováno, že detekce bude probíhat v reálném čase, ale po několika experimentech bylo rozhodnuto vrátit se k detekci objektu na snímku kvůli velkému objemu informací, který silně zatěžoval systém trvalým výpočtem vektorů. Po pořízení snímku uživatelem dochází k zpracování fotografie a po hledání průsečíků vektorů bude objeven čtverec a označen rámečkem s popiskem názvu objektu a jeho barvou.

3.2 Testování jednotlivých komponent zařízení a jejich integrace

Během testování bylo provedeno mnoho změn a vymyšlených řešení problémů. Poté, co byla konstrukce manipulátoru hotová, bylo nutné provést montáž, kalibraci a zkušební spuštění. Prvním krokem byl montáž servopohonů.

Servopohony musí být napájeny zdrojem napětí 5 voltů a proudem 2 ampéry. Toto napájení plně postačí pro napájení jak všech šesti servopohonů, a Bluetooth modulu HC-05. Nejprve je třeba se ujistit, že desky v pořádku.

K tomu byl vyžit jednoduchý testovací skript, blikání LED na desce, a jsou obě desky v pořádku, nastal další krok. Instalace servopohonů na rámu je žádoucí provést zkušební spuštění každého z nich, abychom nedostat k demontáži a výměně servopohonu v pozdějších fázích, stejně jako primární kalibraci, tj. nastavení všech servo na polohu 90 stupňů. K tomu poslouží běžný skript pro ovládání servopohonu. Při montáži je třeba mít na paměti, že servo se pohybují o 90 stupňů vpravo a vlevo od výchozí polohy. Poté, co je úspěšně dokončena kontrola funkčnosti, je třeba servopohony instalovat na rám. Montáž by měla začínat zdola nahoru, tj. od servo, které ovládá celý manipulátor a je označeno jako základní servo, a pokračovat nahoru podél kloubu, konče horním držákem. Během zkušebního spuštění byl zjištěn problém s manipulací těžkých předmětů a kvůli tomu, bylo nutné nainstalovat na tělo kompenzátor. Původně byla vybrána pryž, ale empiricky bylo nalezeno lepší řešení. Nejlepším kandidátem pro toto byla železná pružina, která je schopna se natáhnout na dostatečnou vzdálenost pro práci manipulátoru, a pro její instalaci na tělo bylo zvoleno použití stavebních zipů, jak je ukázáno na fotce. Tímto se zajišťuje, že pružina (Příloha 3) je pevně připevněna a umožňuje manipulátoru úspěšně zdvihat těžší předměty. Tato změna v konstrukci eliminovala problém s manipulací těžkých předmětů a zvýšila spolehlivost manipulátoru jako celku.

Stejně tak byl rámový rám připevněn na desku, ale místo ní bylo možné vybrat jakýkoliv jiný materiál, což bylo nezbytné pro stabilitu konstrukce. Pro instalaci manipulátoru bylo zapotřebí čtyři šrouby nebo hřeby odpovídající průměru děr na těle manipulátoru. Při montáži bylo také nutné sestavit mechanismus uchopení, pro to mohl perfektně posloužit šroub a matice. V tomto případě to byl šroub o délce 2 cm a šířce 0,2 mm a s šířkou hlavy 0,6 mm, díky čemuž byl mechanismus schopen volně provádět proces uchopení. Kabely nebylo doporučeno zavádět dovnitř těla konstrukce, protože v této fázi mohla nastat potřeba odstranit servopohon a provést přestavbu celé konstrukce.

Po montáži bylo třeba provést druhou fázi kalibrace a doladit celkovou pozici manipulátoru tak, aby bylo pohodlné provádět přesun předmětů. Polohy serv pro výchozí pozici byly následující: první servo bylo samozřejmě základna a poté nahoru po kloubu až k šestému servu, což byl samotný uchopovač (Příloha 2).

servo1=90 stupňů

servo2=80 stupňů

servo3=90 stupňů

servo4=60 stupňů

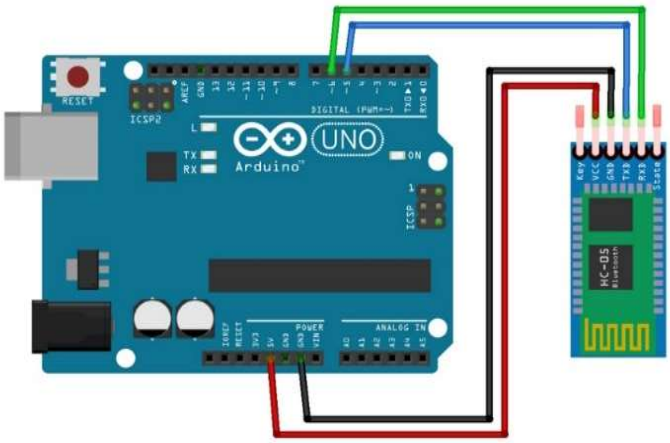
servo5=90 stupňů

servo6=100 stupňů

Manipulátor připraven a bylo nastaveno Bluetooth spojení a prováděno testovací ovládání přes Bluetooth pomocí jednoho ze servopohonů. Původně bylo plánováno, že v bakalářské práce bude pouze jeden modul Bluetooth a integrovaný Bluetooth v notebooku, ale během vývoje nebylo možné spojit modul Bluetooth s integrovaným v notebooku, a problém byl vyřešen přidáním dalšího modulu Bluetooth.

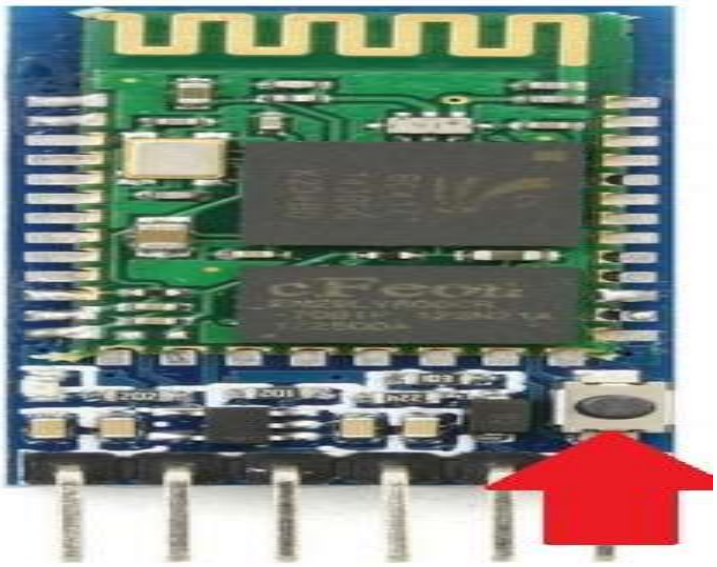
Před propojením všeho je nutné, aby Bluetooth moduly byly nastaveny tak, aby spolu pracovaly. Nejprve byl nastaven režim fungování, kde jeden modul pracuje jako master a druhý jako Slave. Tyto nastavení lze provést pomocí AT příkazů; bez této akce moduly nebudou moci vzájemně komunikovat a toto opatření také zabezpečí spojení před jinými zařízeními. Pomocí AT příkazů je možné zjistit a změnit jméno, heslo a ID modulu. Pro provedení této činnosti je nejprve bylo třeba vzít desku Arduino a nahrát na ni skript AT příkazů. Program pro práce s AT (Příloha 1).

Po nahrání kódu na monitoru sériového portu objevila zpráva "start", pak byla sestavena schéma, jak je ukázáno na (obrázku 17).



Obrázek 17 Připojování Bluetooth modulu (Kolotushkin,2024)

Důležitým detailem je, že při připojování Bluetooth modulu je třeba podržet tlačítko (obrázek 18).

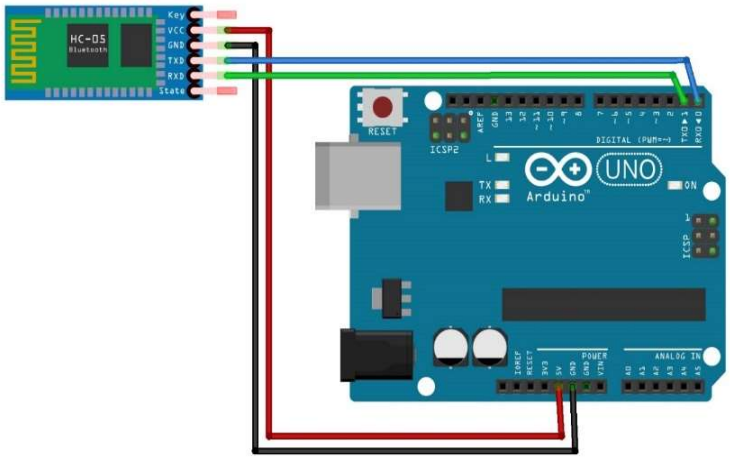


Obrázek 18 Tlačítko

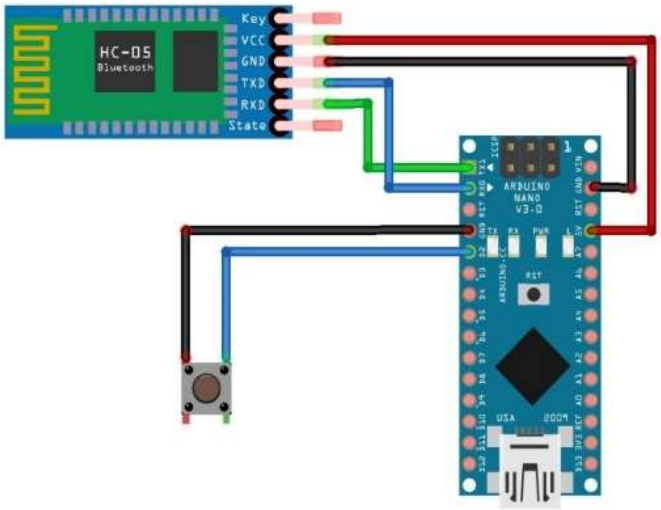
Zapojeny proběhlo úspěšně, LED na desce Bluetooth modulu začala blikat s intervaly 2 sekundy. Může nastat situace, když blikat ne bude, třeba zkusit znovu vyndat modul a znovu ho připojit. Při restartu monitoru sériového portu Arduino by se měla objevit zpráva "OK". Následně pomocí AT příkazů nastaveno jméno a heslo zařízení.

Příkazem AT+NAME=SLAVE nebo AT+NAME=MASTER nastavíme jméno zařízení a heslo AT+PSWD="1234". Tyto příkazy se zadávají do monitoru sériového portu a stiskem Enteru přiřadíme nová data modulu. Dále nastaveni roli modulu pomocí příkazu AT+CMODE=1 pro Slave a AT+CMODE=0 pro Master. Po zjištěni adresu zařízení Slave, abychom mohli mastera najít a připojit k němu, pomocí příkazu AT+ADDR?. Adresa získána a role Slave je nastavena, bylo ověřeno příkazem AT+ROLE?; pro mastera by odpověď měla být AT+ROLE=1 a pro Slave AT+ROLE=0.Pak změna Bluetooth modulu na druhý. Restart monitoru sériového portu a stejné jako kroky pro roli mastera, ale třeba ještě zadat adresu prvního modulu pomocí příkazu AT+BIND=18,E4,400006 a zadanou adresu Slave. Při psaní adresy je důležité, aby byla správně formátovaná, místo dvojteček se používají čárky. Zkontrolovaný roli modulu a heslo bylo stejné pro oba moduly.

Pro testování byl použít následující příklad (Příloha 1): Na začátku nainstalujeme Software a poté je propojíme podle následujícího příkladu (obrázek 19) a (obrázek 20).



Obrázek 19 Deska v roli Slave (Kolotushkin,2024)



Obrázek 20 Deska v roli mastera (Kolotushkin,2024)

Po ověření správného fungování bylo nutné navázat spojení s počítačem.

Hlavní Python skript (Příloha 1), po jehož spuštění dal zkušebny signál tým, že na kameře ukázal červený čtverec, detekovány červený objekt, odeslal signál na desku mastera a poté přes Bluetooth modul na desku Slave. Po testu správnou funkci spojení experimentální část bakalářské práce byla ukončena.

3.3 Zhodnocení výsledků testování a vyhodnocení úspěšnosti dosažení cílů

Po všech testech a kalibracích byl dosažen konečný cíl, ale chtěl bych uvést všechny problémy v každé fázi montáže.

Koupil bych zdroj napájení o napětí 7-12 voltů, aby nebylo nutné připojovat přímo, ale použít vestavěný napájecí konektor Arduino.

Při realizaci bakalářské práce je také doporučeno nakupovat více součástek, než je uvedeno, protože často dochází ke problémům ze strany výrobce a součástky mohou být vadné ještě před odchodem z výroby.Dále je doporučeno provádět vše po etapách, ne sbírat jako celek, protože konstrukce je dost složitá a je třeba mít jistotu, že každá fáze funguje správně.

Výsledky plně splňují technický úkol. Rám, zejména jeho konstrukce, funguje úplně. Servopohony mají dostatečný výkon pro manipulaci s lehkými objekty. Arduino desky jsou schopny provádět výpočty. Bluetooth moduly po nastavení fungují spolehlivě a jsou schopny rychle a bez ztrát vyměňovat data. Algoritmus pro detekci čtverce a určení jeho barvy funguje dostatečně dobře, ale pro správnou funkci je nutné dodržovat podmínku, že na pracovní ploše nejsou žádné jiné objekty podobné čtverci, jinak budou také detekovány, a to by mohlo vést k chybnému signálu pro manipulátor. Ideální by bylo pokrýt pracovní plochu nějakým bílým materiálem. Například čistý list papíru se osvědčil jako dobrá volba, jak je znázorněno (Příloha 2).

Výhody:

Manipulátor jako celková bakalářské práce představuje cenově dostupnou alternativu k dostupným možnostem na trhu. Jeho hodnota spočívá nejen v úspoře nákladů, ale i v jeho univerzálnosti a širokém rozsahu aplikací. I když původně slouží jako vstupní bod pro zájemce o Arduina a robotiku.

Manipulátor je navržen s ohledem na jednoduchost a srozumitelnost, což z něj činí skvělý nástroj pro výuku a experimentování nejen pro jednotlivce, ale i pro vzdělávací instituce. Studenti mohou zkoumat principy programování, bezdrátové komunikace a počítačového vidění v praxi, což posiluje jejich dovednosti a schopnosti.

Díky svému modulárnímu designu je manipulátor snadno upravitelný a rozšiřitelný, což z něj dělá atraktivní možnost nejen pro domácí uživatele, ale i pro vývojáře a průmyslové aplikace. Možnost jeho použití v inteligentních domácnostech, vzdělávacích institucích nebo dokonce v průmyslových procesech dává bakalářské práce potenciál dosáhnout širšího publiku a trhu.

Vzhledem k jeho všestrannosti a potenciálu pro výuku a vývoj lze očekávat, že manipulátor bude nadále inspirativní a relevantní pro komunitu zájemců o robotiku a technologii. Možnosti jeho rozšíření a dalšího vývoje jsou neomezené.

Nevýhody:

Je pochopitelné, že pevnost konstrukce a nosnost jsou klíčovými faktory, které ovlivňují praktické využití manipulátoru. Je důležité, aby manipulátor dokázal zvládnout širší spektrum úkolů a manipulovat s různými předměty, včetně těch těžších. Zatímco v současné době může

být použitelný pro lehčí předměty a základní úkoly, jeho schopnost by měla být rozšířena tak, aby mohl být efektivně využíván i ve složitějších situacích.

Servomotory jsou nedílnou součástí manipulátoru a jejich výkon ovlivňuje možnosti a efektivitu pohybu. Zvýšení výkonu servomotorů by umožnilo manipulátoru řešit náročnější úkoly a pracovat s těžšími předměty, což by rozšířilo jeho možnosti využití.

Aplikace pro programování pohybů manipulátoru by značně zjednodušila proces vytváření a úpravy pohybových sekvencí. Uživatelsky přívětivé prostředí s intuitivním rozhraním by umožnilo uživatelům snadno definovat pohyby a přizpůsobit je svým potřebám, což by výrazně zlepšilo uživatelskou zkušenost.

Optimalizace procesu 3D tisku by snížila čas a náklady spojené s výrobou dílů manipulátoru a zvýšila by kvalitu výsledných komponent. Minimalizace chyb při výrobě by zase snížila riziko selhání manipulátoru a zvýšila jeho spolehlivost.

Celkově lze říct, že vylepšení v těchto oblastech by mohlo výrazně posílit schopnosti a využitelnost manipulátoru, což by jej učinilo atraktivnějším a užitečnějším nástrojem pro uživatele v různých oblastech, jako je robotika, domácí automatizace a vzdělávání.

3.4 Možnosti budoucího rozvoje a vylepšení zařízení

Při dalším zlepšování této konstrukce manipulátoru by bylo vhodné vzít v úvahu zkušenosti z jeho realizace. Pro zlepšení stability a efektivity práce manipulátoru je doporučeno obložit základnu, na které je umístěn, krytem. V tomto krytu by mohl být horní pohyblivý část manipulátoru upevněn. To by umožnilo zabránit naklánění osy Y při zdvihu těžších předmětů, což někdy brání otáčení nosné části.

Kromě toho by bylo vhodné vyvinout softwarové vybavení ve formátu aplikace pro pohodlné nastavení pohybů manipulátoru. Například by bylo možné implementovat funkci ukládání pohybů uživateli pro následné samostatné opakování těchto pohybů. To by zjednodušilo proces nastavování a zpřístupnilo ho koncovým uživatelům.

Dalším logickým krokem po realizaci vylepšení by bylo začít pracovat na kovové součástky a výkonné servopohony. To by umožnilo vytvořit robustnější a výkonnější manipulátor, který by byl schopen zvládat složitější úkoly a zátěže.

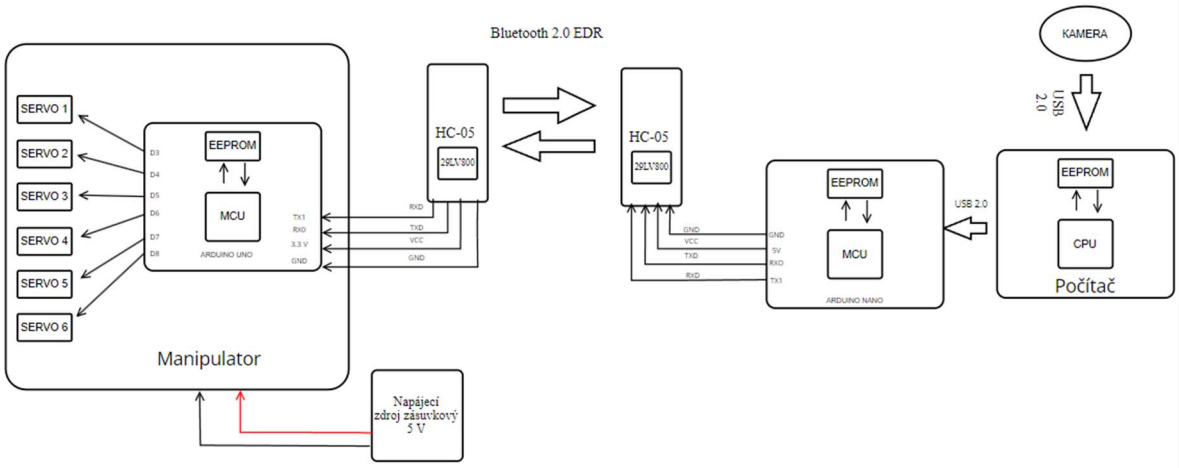
4. Realizace

Při realizaci této finální sestavy je důležité zajistit správnou synchronizaci a komunikaci mezi jednotlivými částmi systému. Začínáme získáním obrazových dat z kamery, která jsou následně zpracována na počítači pomocí speciálního softwaru. Tento software identifikuje barvu a tvar objektu podle předem definovaných kritérií. Pokud je objekt v souladu s těmito kritérii, generuje se signál, který je poslán na desku Arduino Nano přes USB kabel.

Arduino Nano, která je zároveň napájena z tohoto USB kabelu, zpracovává příchozí signál a předává ho pomocí Bluetooth modulu na desku pro spolupráci. Na desce pro spolupráci je

umístěna Arduino Uno, která na základě příchozího signálu rozhodne, jakou část kódu spustit. Poté přivede napětí na servopohony, které provádějí fyzickou manipulaci s objektem.

V tomto procesu je klíčové zajistit spolehlivou a rychlou komunikaci mezi jednotlivými komponenty systému. Je také důležité implementovat robustní softwarové řešení pro zpracování obrazových dat a řízení pohybu manipulátoru. Výsledkem je plně automatizovaný systém, který dokáže identifikovat, manipulovat a reagovat na objekty podle předem stanovených parametrů.



Obrázek 21 Hardver diagram

- servo 1 označí otáčení základny
- servo 2 otáčení prvního ramene dopředu a dozadu
- servo 3 otáčení druhého ramene nahoru a dolů
- servo 4 otáčení zápěstí doprava a doleva
- servo 5 otáčení zápěstí nahoru a dolů
- servo 6 otáčení efektoru

4.1 Práce s počítačovým viděním

Pro správné fungování kódu bakalářské práce potřeba po instalaci vývojového prostředí v Pythonu také nainstalovat externí knihovny, jako například:

import cv2 #OpenCV

import numpy as np #čislicove operace v Python

import warnings #pro chyby

import tensorflow as tf

To lze provést přes příkazový řádek pomocí příkazu pip install, například instalace tensorflow by vypadala takto:

pip install tensorflow

Poté třeba ověřit funkci kódu v Pythonu, ale pro spuštění kódu je nutné, aby kamera a deska Arduina byly připojeny k počítači. V mém případě je pro Arduinu použit port:

arduino_port = 'COM8'

a pro kameru:

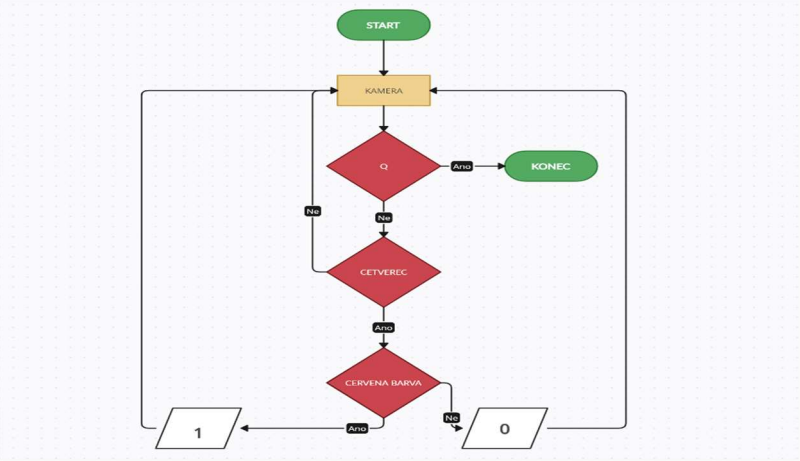
usb_camera_index = 1

Portová čísla se mohou lišit pro zjištěni čísla portu je moznost podívat v nastavení kamery na počítači. V případě portu Arduino bude po připojení k počítači ve vývojovém prostředí v sekci Nástroje v podsekci Port uvedeno číslo pracovního portu, přes který deska komunikuje s počítačem, a tyto hodnoty potřeba nahradit v kódu, ve kterém bylo implementováno počítačové vidění. Po spuštění se zobrazí okno, ve kterém je zobrazen obraz z kamery, a následně se celý proces spustí stisknutím mezerníku:

if key == 32: # ASCII tlačítko mezera

Volitelně můžete vybrat jakoukoli jinou klávesu. Stisknutí této klávesy simuluje signál ze senzoru o tom, že objekt dorazil, a s ním lze provádět manipulace. Po stisknutí se objeví snímek, ve kterém je objekt již ohraničen čtvercem a pod rámečkem je napsáno, o jaký objekt jde, a jaká má barvu. Poté manipulátor přesune objekt do úložiště podle jeho barvy a po dokončení manipulace lze proces detekce opakovat. Pro uzavření programu bylo vybráno stisknutí klávesy "q", tato klávesa je také nastavitelná a lze vybrat jakoukoli klávesu.

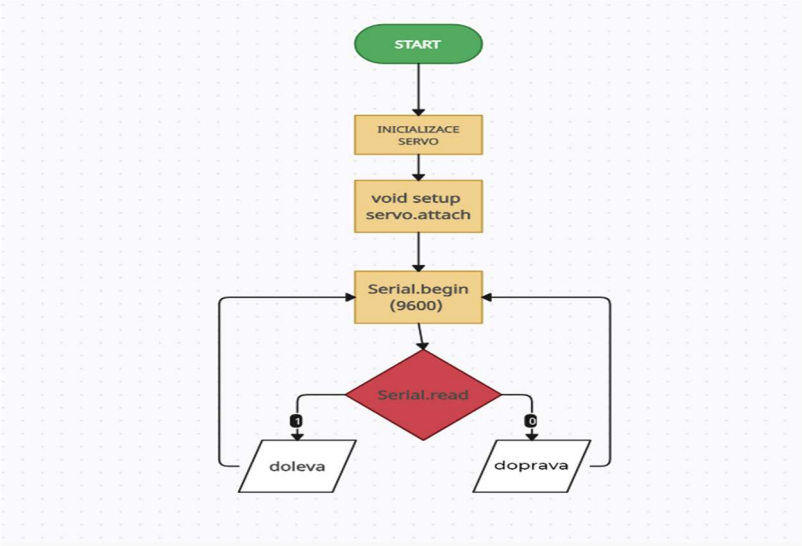
Níž ukázán vývojový diagram SW PC (Python).



Obrázek 22 Vývojový diagram SW PC (Python)

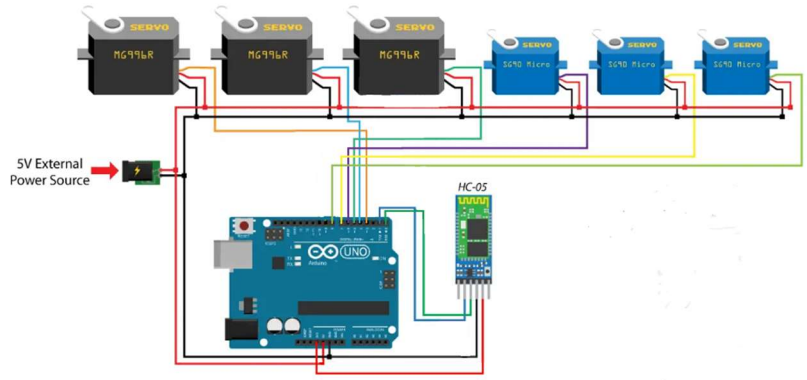
4.2 Blok manipulátoru a desky v roli Slave

Je doporučeno začít s montáží postupně, nejprve sestavíme samotný manipulátor, nahráváme finální skript pro Slave na desku (Příloha 1). Níž ukázán vývojový diagram pro Slave na desku.



Obrázek 23 Vývojový diagram pro Slave

propojíme vše podle příkladu (obrázek 24).



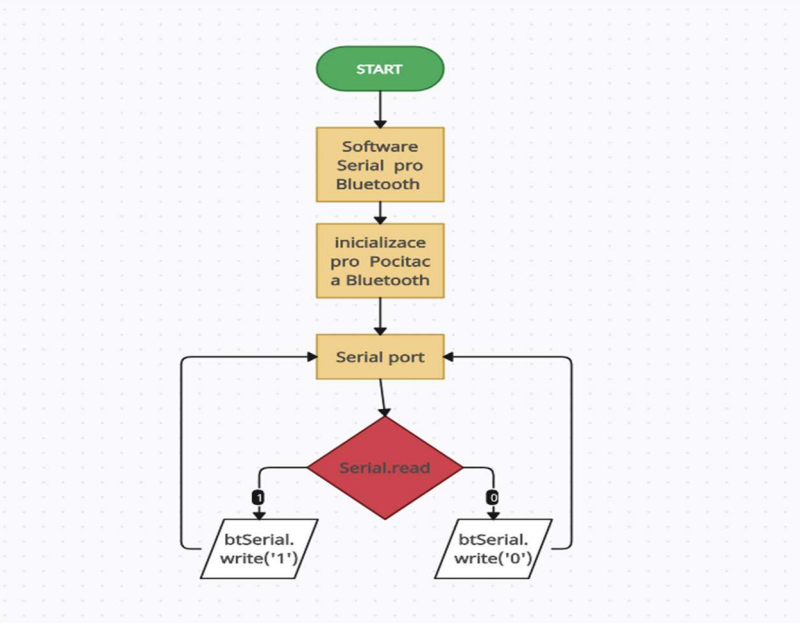
Obrázek 24 Blok manipulátoru jako Slave (HowToMechatronics,2024)

Při připojení napájení je třeba připojit vstup na 5 voltů do desky Arduino místo portu DC Power Jack. To je proto, že je navržen pro napájení od 7 do 12 voltů a poté stabilizuje napětí na 5 voltů. Pokud je deska přímo napájena ze zdroje na 5 voltů přes konektor na desce, na výstupu dostaneme pouze 3 voltů, což nestačí k napájení všech servo. Manipulátor by měl po montáži vypadat jako na (Příloha 2).

4.3 Deska Master

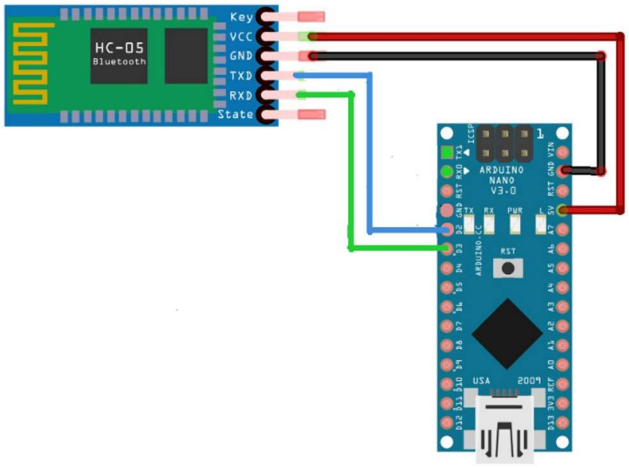
Desku v roli mastera připojíme k počítači a nahrajeme nový kód pro mastera (Příloha 1).

Při práci s tímto kódem inicializujeme dva sériové porty pro komunikaci s počítačem a Bluetooth modulem. Na (obrázku 25) ukázán vývojový diagram.



Obrázek 25 Vývojový diagram Master

Po nahrání skriptu Master (Příloha 1) na desku provedeme instalaci a sestavíme desku (obrázek 26).



Obrázek 26 Blok manipulátoru jako Master

4.4 Spuštění bakalářské práce

Dalším krokem je instalace kamery, připojeni desky Master a napájeni desky Slave. Po tomu, když Bluetooth moduly spojí, poznat to možné, pokud vše LED diody blikají jednou za 2 sekundy, spuštěny kódu počítačového vidění v Pythonu. Poté je bakalářské práce připravená k použití.

5. Závěr

5.1 Shrnutí dosažených výsledků

Bakalářská práce byla dostatečně zajímavá a poskytuje mnoho znalostí v oblasti mechatroniky, programování a práce s umělou inteligencí. Manipulátor je skutečně jedním z největších úspěchů lidstva. Je to prostřední, které se nachází mezi lidskou populací a umělou inteligencí. Je to počáteční fáze k vyřešení kompletní konstrukce robota, který již bude schopen myslet pomocí umělé inteligence. Manipulátor je věc, která již zjednodušila téměř veškeré průmyslové odvětví a umožnila firmám snížit náklady na výrobu, což vede ke zvýšení objemu výroby, rychlosti a kvality. Roboti jsou dalším stupněm pro všechna lidská odvětví a my jako lidé budeme moci trávit více času věnováním se věcem, které jsou zajímavější a méně nebezpečné. Například již nyní robot Spot dokáže zachytit úniky plynu na závodě. V budoucnu však bude možné nahradit mnoho dalších nebezpečných profesí, jako jsou armáda, hasiči a mnoho dalších. Rozvoj automatizace a robotiky člověka pomůže společnosti přinášet do tohoto množství různorodých robotů a technologií jejich realizace. A čím více lidí se tímto směrem bude zabývat, tím rychleji se celá naše společnost stane lepší, a veškerá monotónní a nebezpečná práce půjde do zapomnění, jako je například profese komináře. My sami však získáme obrovské množství zboží a produktů, které budou dostupnější pro všechny. Tento pokrok v oblasti technologie a automatizace slibuje lepší a bezpečnější budoucnost pro lidstvo. Celkem cena byla přibližně 3000 korun českých. Je to do cela moc ale manipulátor byl udělán se začátku a bez potřebných znalostí, a proto vyšel dražším, než bylo očekáváno. Ale taková cena taky do cela dobra, protože stejné výrobky můžeme najit od 5+ tisíc.

5.2 Doporučení pro další výzkum a vývoj v dané oblasti

V budoucnu by bylo vhodné zaměřit se na manipulátor schopný asistovat v domácích úkolech. Konstrukce bakalářské práce by mohla být výrazně vylepšena, například přidáním pohyblivé základny, která by umožňovala autonomní pohyb po domě a ovládání operátorem skrze helmu rozšířené reality či joystick. Také by bylo užitečné modernizovat kód počítačového vidění, aby bylo možné rozpoznávat i další barvy a tvary. Pro zvýšení uživatelského komfortu by se mohl přidat i způsob ukončení programu pomocí tlačítka pro uzavření okna kamery. Ještě jeden návrh pro zlepšeni kódu je to realizace automatického pochybu bez závislosti od počátečními polohy objektu například pomoci rozděleny počátečního obrazu maticové a v závislosti na toma v jakém čtverců matice objekt je posílat různý signály na desku manipulátoru. V desce že realizovat pokrokový pochyb a svázat stupni otáčeny pohonu z virtuální matice jako například máme objekt v levém čtverců a posíláme na desku od začátku polohu počáteční pohon který rotuje celkový systém do leva nebo do pravá stupeň otáčeny a pak odeslat signál o spuštěny hlavního kódu manipulace s objektem.

6. Použita literatura

[1] Webkamera Trust Trino HD video [online]. In: . [cit. 2024-04-15]. Dostupné z: https://www.datart.cz/webkamera-trust-trino-hd-video-cerna.html

[2] Servo MG996R (MG996) s kovovými převody 13kg 360° [online]. In: . [cit. 2024-04-15]. Dostupné z: https://dratek.cz/arduino/122950-servo-mg996r-s-kovovymi-prevody-13kg-360.html?gad_source=1&gclid=CjwKCAjwoPOwBhAeEiwAJuXRh1ptP3oKq0YQu3E8NxDLECRC7GbLkus88X6XXf7sG18Ltq4o6cl0JRoCq2QQAvD_BwE

[3] Servomotor mikro - 180° [online]. In: . [cit. 2024-04-15]. Dostupné z: https://dratek.cz/arduino/897-eses-servo-motor-9g.html?gad_source=1&gclid=CjwKCAjwoPOwBhAeEiwAJuXRhxnfOOpKOvP4OLGJM9uyPjmetG_dcF7LlKdMw1lY9ubc7Rw_I-waiRoCoEgQAvD_BwE

[4] Bluetooth modul HC-05 TTL [online]. In: . [cit. 2024-04-15]. Dostupné z: https://www.laskakit.cz/bluetooth-modul-hc-05-ttl/?gad_source=1&gclid=CjwKCAjwoPOwBhAeEiwAJuXRh-xb1dk94rLnU5i3ZVB0YQhHbuOo4eqbRFKvLFguAMSJaA_PkQgxkBoCgO8QAvD_BwE

[5] Napájecí zdroj zásuvkový 5V, 2A, 4.0x1.75mm [online]. In: . [cit. 2024-04-15]. Dostupné z: https://ampul.eu/cs/zasuvkove-zdroje/4263-napajeci-zdroj-zasuvkovy-5v-2a-40x175mm?SubmitCurrency=1&id_currency=1&gad_source=1&gclid=CjwKCAjwoPOwBhAeEiwAJuXRh8QZhZHzFEUg3KADDFAWIKjnFD1UNov57mcpgIAX_zQXI7rcGKuB_RoCA50QAvD_BwE

[6] Ya Xun YX-8022C [online]. In: . [cit. 2024-04-15]. Dostupné z: https https://mobiround.ru/catalog/p11884/

[7] UT 131C, UNI-T multimetr s měřením teploty [online]. In: . [cit. 2024-04-15]. Dostupné z: https://www.hitobchod.cz/ut-131c--uni-t-multimetr-s-merenim-teploty/?gad_source=1&gclid=CjwKCAjwoPOwBhAeEiwAJuXRh4FXcXpxj0WCiDw0ziFujOvQgk4k-gXtQaj5O0cXBYtyENxPuRS-fBoCuB0QAvD_BwE

[8] ŠVEJDA, Martin,2011. Kinematika robotických architektur, Plzeň

Práce ke státní doktorské zkoušce. ZÁPADOČESKÁ UNIVERZITA V PLZNI Fakulta aplikovaných věd Katedra kybernetiky

[9] Indonesian Journal of Electrical Engineering and Computer Science Vol. 16, No. 1, October 2019, pp. 389~394 ISSN: 2502-4752, DOI: 10.11591/ijeecs.v16.i1.pp389-394

[10] Slovník [online]. In: . [cit. 2024-04-15]. Dostupné z: http://lopatnikov.pro/slovar/a/approksimaciya/

[11] SKAŘUPA, J. 2007. Průmyslové roboty a manipulátory. Ostrava: Ediční středisko VŠB – TUO.

ISBN 978-80-248-1522-0.

[12] OpenCV [online]. In: . [cit. 2024-04-15]. Dostupné z: https://sourceforge.net/projects/opencvlibrary/

[13] Douglas-Peucker [online]. In: . [cit. 2024-04-15]. Dostupné z: https://psimpl.sourceforge.net/douglas-peucker.html

[14] Arduino Robot [online]. In: . [cit. 2024-04-15]. Dostupné z: https://howtomechatronics.com/tutorials/arduino/diy-arduino-robot-arm-with-smartphone-control/

[15] DuPont kabel M-F - 40x, 20 cm [online]. In: . [cit. 2024-04-15]. Dostupné z: https://dratek.cz/arduino/1214-40-x-m-f-dupont-kabel-20-cm.html?_gl=1*q7opm2*_up*MQ..&gclid=CjwKCAjw5v2wBhBrEiwAXDDoJRIHDyIdvh8g-aNC_BAWUQ4mMz99Y8bu8XCwue5kwJRQ_YYJRaxNVhoC2MMQAvD_BwE

[16] Arduino IDE 2.3.2 [online]. In: . [cit. 2024-04-15]. Dostupné z: https://www.arduino.cc/en/software

[17] Python 3.11.0 [online]. In: . [cit. 2024-04-15]. Dostupné z: https://www.python.org/downloads/release/python-3110/

[18] Arduino UNO R3, ATmega328P [online]. In: . [cit. 2024-04-15]. Dostupné z: https://www.laskakit.cz/arduino-uno-r3--atmega328p--klon/?gad_source=1&gclid=CjwKCAjw5v2wBhBrEiwAXDDoJaKQU3zNaVmj38eyUCEYo_1fGo5-ieODudYuqUfD5WYIPvmbmdOlnBoCU0sQAvD_BwE

[19] ARDUINO NANO V3.0 [online]. In: . [cit. 2024-04-15]. Dostupné z: https://www.turtle3d.cz/cz/494/13/arduino-nano-v3-0?gad_source=1&gclid=CjwKCAjw5v2wBhBrEiwAXDDoJQcOeXmo_syFv_XjeaPHDdOgmkxzgcvcGVf8E_kAdrsQ5v6aevv-lhoCc80QAvD_BwE

[20] T. Tieleman and G. Hinton. Lecture 6.5 rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 4(2), 2012.

[21] VODA, Zbyšek. Průvodce světem Arduina. Vydání druhé. Bučovice: Martin Stříž, 2017. ISBN 978-80-87106-93-8.

[22] HC05 Bluetooth [online]. In: . [cit. 2024-04-15]. Dostupné z: https://kolotushkin.com/article.php?id=30

[23] OpenCV [online]. In: . [cit. 2024-04-15]. Dostupné z: https://docs.opencv.org/4.x/d1/d8f/tf_cls_tutorial_dnn_conversion.html

[24] TensorFlow [online]. In: . [cit. 2024-04-15]. Dostupné z: https://www.tensorflow.org/tutorials?hl=ru

7. Seznam příloh

Příloha 1: Program

Deska v roli Slave

```
char command;
#include <Servo.h>
Servo servo1;
Servo servo2;
Servo servo3;
Servo servo4;
Servo servo5;
Servo servo6;

int pozverch = 90;
int pozniz = 90;
int pozbaza = 90;
int chapadlo=0;
int vertchap=0;
int gorztchap=0;
const int led_Pin = 13;

void setup()  {
  servo1.attach(3);
  servo2.attach(4);
  servo3.attach(5);
  servo4.attach(6);
  servo5.attach(7);
  servo6.attach(8);

  servo1.write(90);
```

```
  servo2.write(80);
  servo3.write(90);
  servo4.write(60);
  servo5.write(90);
  servo6.write(100);
  pinMode( led_Pin,OUTPUT);
  Serial.begin(9600);
}

void loop() {
  if (Serial.available()) {
    char c = Serial.read();

if (c == '1'){
  for (pozniz = 62 ; pozniz <= 80; pozniz += 1) {
    servo2.write(pozniz);  // +
   delay(60);
    }
for (vertchap = 90 ; vertchap <= 100; vertchap += 1) {
    servo5.write(vertchap);  // +
   delay(60);
    }


for (pozverch = 90 ; pozverch >= 60; pozverch -= 1) {
    servo3.write(pozverch);  // +
   delay(60);
    }

for (chapadlo =100 ; chapadlo >= 50; chapadlo -= 1) {  //bodjem i spusk
```

```
   servo6.write(chapadlo);  // +
  delay(60);
   }
for (vertchap = 100 ; vertchap >= 90; vertchap -= 1) {
   servo5.write(vertchap);  // +
  delay(60);
   }
for (pozverch = 60 ; pozverch <= 90; pozverch += 1) {
   servo3.write(pozverch);  // +
  delay(60);
   }
for (pozniz = 80 ; pozniz >= 62; pozniz -= 1) {
   servo2.write(pozniz);  // +
  delay(60);
   }
 ////////////


for (pozbaza = 90 ; pozbaza >= 0; pozbaza -= 1) {
   servo1.write(pozbaza);  // +
  delay(60);
   }


   ////////////////////////////////////
for (pozniz = 62 ; pozniz <= 80; pozniz += 1) {
   servo2.write(pozniz);  // +
  delay(60);
   }
for (vertchap = 90 ; vertchap <= 100; vertchap += 1) {
   servo5.write(vertchap);  // +
  delay(60);
```

```
    }

for (pozverch = 90 ; pozverch >= 60; pozverch -= 1) {
    servo3.write(pozverch);  // +
   delay(60);
    }

for (chapadlo =50 ; chapadlo <= 100; chapadlo += 1) {
    servo6.write(chapadlo);  // +
   delay(60);
    }
for (vertchap = 100 ; vertchap >= 90; vertchap -= 1) {
    servo5.write(vertchap);  // +
   delay(60);
    }
for (pozverch = 60 ; pozverch <= 90; pozverch += 1) {
    servo3.write(pozverch);  // +
   delay(60);
    }
for (pozniz = 80 ; pozniz >= 62; pozniz -= 1) {
    servo2.write(pozniz);  // +
   delay(60);
    }
for (pozbaza = 0; pozbaza <= 90; pozbaza += 1) {
    servo1.write(pozbaza);  // --
   delay(60);
}
  digitalWrite(led_Pin,HIGH);
    }
if (c == '0'){ digitalWrite(led_Pin, LOW);
```

```
 for (pozniz = 62 ; pozniz <= 80; pozniz += 1) {
    servo2.write(pozniz);  // +
   delay(60);
    }
for (vertchap = 90 ; vertchap <= 100; vertchap += 1) {
    servo5.write(vertchap);  // +
   delay(60);
    }
for (pozverch = 90 ; pozverch >= 60; pozverch -= 1) {
    servo3.write(pozverch);  // +
   delay(60);
    }
for (chapadlo =100 ; chapadlo >= 50; chapadlo -= 1) {  //bodjem i spusk
    servo6.write(chapadlo);  // +
   delay(60);
    }
for (vertchap = 100 ; vertchap >= 90; vertchap -= 1) {
    servo5.write(vertchap);  // +
   delay(60);
    }
for (pozverch = 60 ; pozverch <= 90; pozverch += 1) {
    servo3.write(pozverch);  // +
   delay(60);
    }
for (pozniz = 80 ; pozniz >= 62; pozniz -= 1) {
    servo2.write(pozniz);  // +
   delay(60);
    }
 ////////////
```

```
   for (pozbaza = 90; pozbaza <= 180; pozbaza += 1) {
   servo1.write(pozbaza);  // --
  delay(60);
    }

   ////////////////////////////////////
for (pozniz = 62 ; pozniz <= 80; pozniz += 1) {
   servo2.write(pozniz);  // +
  delay(60);
   }
for (vertchap = 90 ; vertchap <= 100; vertchap += 1) {
   servo5.write(vertchap);  // +
  delay(60);
   }

for (pozverch = 90 ; pozverch >= 60; pozverch -= 1) {
   servo3.write(pozverch);  // +
  delay(60);
   }

for (chapadlo =50 ; chapadlo <= 100; chapadlo += 1) {
   servo6.write(chapadlo);  // +
  delay(60);
   }
for (vertchap = 100 ; vertchap >= 90; vertchap -= 1) {
   servo5.write(vertchap);  // +
  delay(60);
   }
for (pozverch = 60 ; pozverch <= 90; pozverch += 1) {
   servo3.write(pozverch);  // +
```

```
   delay(60);
    }
for (pozniz = 80 ; pozniz >= 62; pozniz -= 1) {
    servo2.write(pozniz);  // +
   delay(60);
    }
 for (pozbaza = 180 ; pozbaza >= 90; pozbaza -= 1) {
    servo1.write(pozbaza);  // +
   delay(60);
    }
    }
}
}
```

Deska v roli mastera

```
#include <SoftwareSerial.h>
const int BT_TX_PIN = 2; // TX pin Bluetooth
const int BT_RX_PIN = 3; // RX pin Bluetooth
//  SoftwareSerial  pro  Bluetooth
SoftwareSerial btSerial(BT_TX_PIN, BT_RX_PIN);
void setup() {                    // inicializace pro  počítač
  Serial.begin(9600);
// inicializace pro Bluetooth
  btSerial.begin(9600);
}
void loop() {
  if (Serial.available() > 0) {
    char signal = Serial.read();
    // pokud máme 1 tak přepošlu dal
    if (signal == '1') {
      btSerial.write('1');
```

```
    }
    else if (signal == '0') {
      btSerial.write('0');
    }
  }
  // ctěný s druhého portu
  if (btSerial.available() > 0) {
    char signal = btSerial.read();
    // ukázka pro údržbu
    Serial.println(signal);
  }
}
```

Kód pro počítač

```
import cv2 #OpenCV
import numpy as np #cislove operace в Python
import warnings #pro chybu
import tensorflow as tf #instalace  MobileNetV2.
from tensorflow.keras.applications import MobileNetV2
#model hlubokého učení, pro mobilní a vestavěné aplikace strojového vidění
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input #
from sklearn.cluster import KMeans #klasterizace barev.
import serial
import time


#načteny předtrénovane verze modelu MobileNetV2 bez posledních vrstev. Tento model byl předtrénován na souboru dat ImageNet.
base_model = MobileNetV2(weights="imagenet", include_top=False, input_shape=(224, 224, 3))
frame_width = 640
frame_height = 480
```

```
usb_camera_index = 1
cap = cv2.VideoCapture(usb_camera_index)
cap.set(3, frame_width)
cap.set(4, frame_height)

arduino_port = 'COM7' # port arduino
arduino_baudrate = 9600
ser = serial.Serial(arduino_port, arduino_baudrate, timeout=1)
time.sleep(2)

while True:
    ret, frame = cap.read()

    cv2.imshow("Frame", frame)

    key = cv2.waitKey(1)
    if key == 32:  # ASCII mezera
        snapshot = frame.copy()
        hsv = cv2.cvtColor(snapshot, cv2.COLOR_BGR2HSV)

        # kombinace masek pro cerveny
        lower_red1 = np.array([0, 100, 100])
        upper_red1 = np.array([10, 255, 255])
        mask_red1 = cv2.inRange(hsv, lower_red1, upper_red1)

        lower_red2 = np.array([160, 100, 100])
        upper_red2 = np.array([180, 255, 255])
        mask_red2 = cv2.inRange(hsv, lower_red2, upper_red2)

        mask_red = cv2.bitwise_or(mask_red1, mask_red2)
```

```
        # kombinace pro modry
        lower_blue = np.array([100, 100, 100])
        upper_blue = np.array([140, 255, 255])
        mask_blue = cv2.inRange(hsv, lower_blue, upper_blue)

        mask = cv2.bitwise_or(mask_red, mask_blue)

        object_region = cv2.bitwise_and(snapshot, snapshot, mask=mask)

        contours, _ = cv2.findContours(mask, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

        #  Přebírat kontury a najt max
        max_contour = max(contours, key=cv2.contourArea)

        #Definice pro  ohraničovací rámeček největšecho obrysu
        x, y, w, h = cv2.boundingRect(max_contour)

        cv2.rectangle(snapshot, (x, y), (x+w, y+h), (0, 255, 0), 2)
        cv2.putText(snapshot, 'Čtverec ', (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, (0,
255, 0), 2)

        # Kontrola několika bodů uvnitř objektu
        color_points = [(x+w//4, y+h//4), (x+w//2, y+h//2), (x+3*w//4, y+3*h//4)]

        red_count = 0
        blue_count = 0
        for point in color_points:
            if mask_red[point[1], point[0]] == 255:
                red_count += 1
```

```
            elif mask_blue[point[1], point[0]] == 255:
                blue_count += 1

        if red_count > blue_count:
            color_label = 'cerveny'
        elif blue_count > red_count:
            color_label = 'modry'
        else:
            color_label = 'unknown'

        cv2.putText(snapshot, f'Color: {color_label}', (x, y+h+30),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 255, 0), 2)



        if color_label == 'cerveny':
            ser.write(b'1')
        elif color_label == 'modry':
            ser.write(b'0')

        cv2.imshow("Snapshot", snapshot)

    elif key == ord('q'):
        break

cap.release()
ser.close()
cv2.destroyAllWindows()
```

Program pro práce s AT

```
const int arduino_rx = 5;
```

```
const int arduino_tx = 6;
SoftwareSerial mySerial (arduino_rx, arduino_tx);

void setup()  {
  pinMode( arduino_rx,INPUT); pinMode( arduino_tx,OUTPUT);
  Serial.begin(9600);
  mySerial.begin(38400);
  Serial.println( "<<< Start! >>>");
  mySerial.println("AT");
}

void loop() {
  if (mySerial.available()) {
    char c = mySerial.read();      // ctěny s portu
    Serial.print(c);   }           // ukázka Serial-porta
  if (Serial.available()) {
    char c = Serial.read();        // ctěný Serial-porta
    mySerial.write(c); }           // zápis  do Serial-porta
```

Zkoušková kalibrace pro mastera.

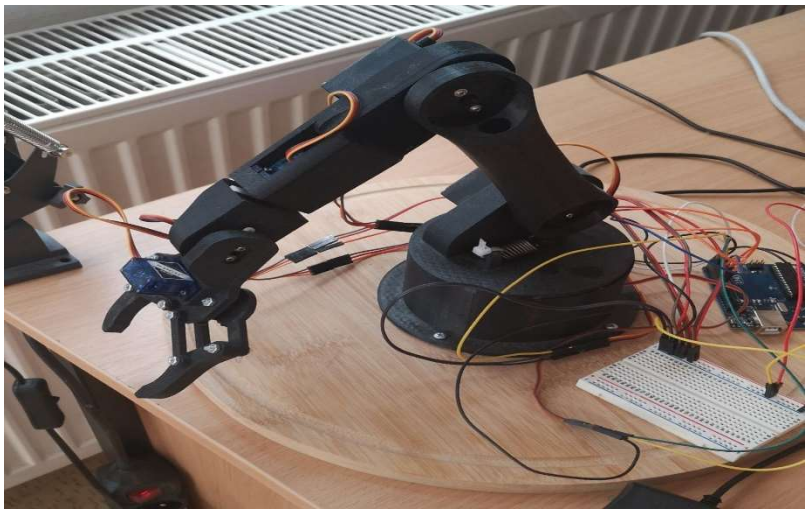
```
const int button = 2;
int button_state = 0;

void setup()  {
  pinMode( button,INPUT_PULLUP);
  Serial.begin(9600);
}

void loop() {
  button_state = digitalRead(button);
```

```
  if (button_state == LOW){ Serial.print("1"); }
  if (button_state == HIGH){ Serial.print("0"); }
}
```

Zkoušková kalibrace pro sleva

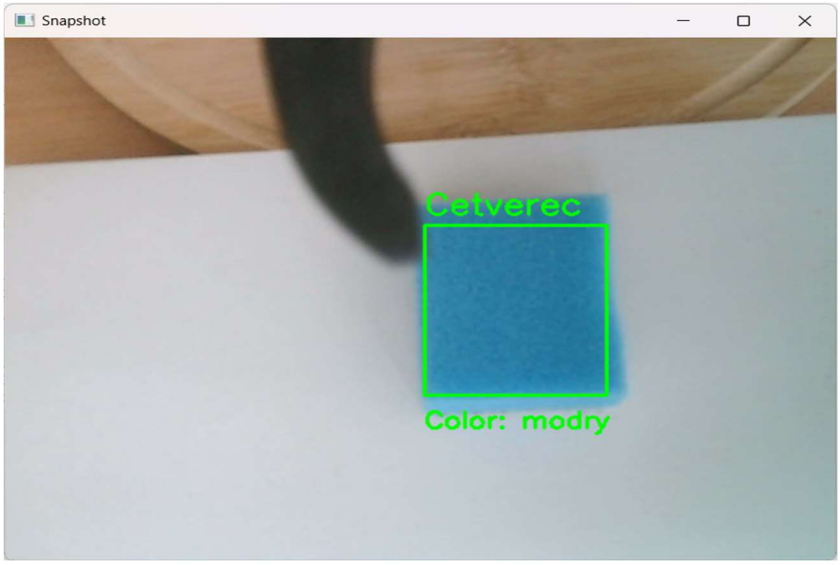
```
const int led_Pin = 13;
void setup()  {
  pinMode( led_Pin,OUTPUT);
  Serial.begin(9600);
}

void loop() {
  if (Serial.available()) {
    char c = Serial.read();

if (c == '1'){ digitalWrite(led_Pin, HIGH);}
if (c == '0'){ digitalWrite(led_Pin, LOW);}
  }
}
```

Příloha 2: Foto

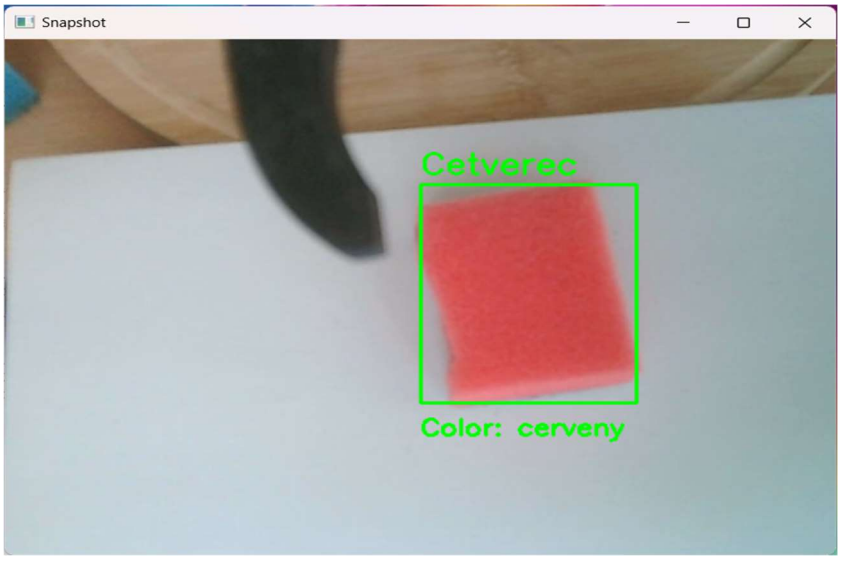


Obrázek 27 Manipulátor po montáži

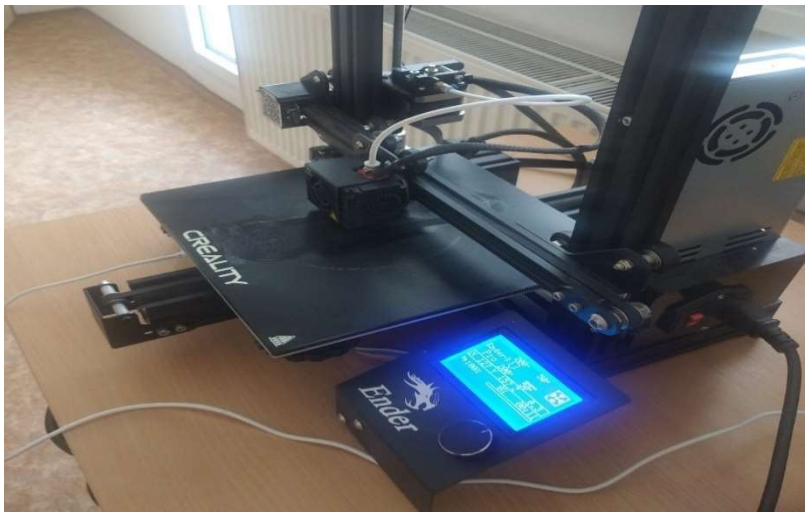
Obrázek 28 Kompenzátor



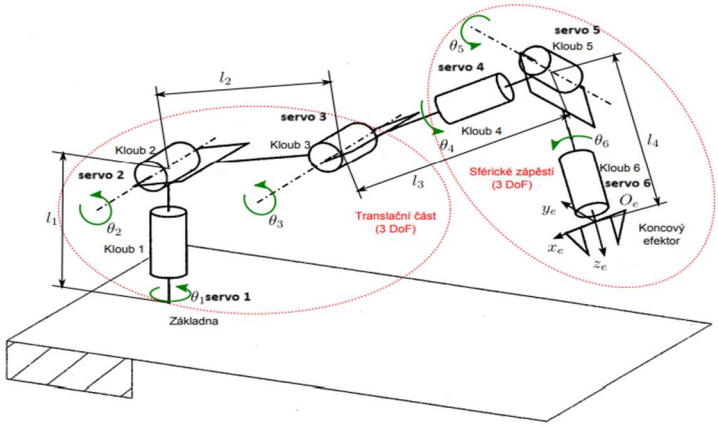
Obrázek 29 Detekce modrého čtverce



Obrázek 30 Detekce červeného čtverce



Obrázek 31 Creality ENDER 3



Obrázek 32 Polohy servo (Švejda,2011)