



AUTOMATICKY NAVÁDĚNÝ PODVOZEK POMOCÍ AI, UWB A GPS

Bc. Kryštof Vacek

Diplomová práce
Fakulta informačních technologií
České vysoké učení technické v Praze
Katedra číslicového návrhu
Studijní program: Informatika
Specializace: Návrh a programování vestavných systémů
Vedoucí: Ing. Pavel Kubalík, Ph.D.
1. května 2026



Zadání diplomové práce

Název:	Automaticky naváděný podvozek pomocí AI, UWB a GPS
Student:	Bc. Kryštof Vacek
Vedoucí:	Ing. Pavel Kubalík, Ph.D.
Studijní program:	Informatika
Obor / specializace:	Návrh a programování vestavných systémů
Katedra:	Katedra číslicového návrhu
Platnost zadání:	do konce letního semestru 2026/2027

Pokyny pro vypracování

Prozkoumejte existující řešení.

Navrhněte vlastní řešení pro platformu Raspberry Pi.

Řešení bude umožňovat:

- 1) režim sledování (follow me) pomocí kamery a umělé inteligence,
- 2) detekci překážek,
- 3) navádění pomocí technologie UWB,
- 4) navádění pomocí GPS,
- 5) vzdálené ovládání pomocí webové aplikace.

Navržené řešení realizujte a řádně otestujte.

České vysoké učení technické v Praze
Fakulta informačních technologií

© 2026 Bc. Kryštof Vacek. Všechna práva vyhrazena.

Tato práce vznikla jako školní dílo na Českém vysokém učení technickém v Praze, Fakultě informačních technologií. Práce je chráněna právními předpisy a mezinárodními úmluvami o právu autorském a právech souvisejících s právem autorským. K jejímu užití, s výjimkou bezúplatných zákonných licencí a nad rámec oprávnění uvedených v Prohlášení, je nezbytný souhlas autora.

Odkaz na tuto práci: Vacek Kryštof. *Automaticky naváděný podvozek pomocí AI, UWB a GPS*. Diplomová práce. České vysoké učení technické v Praze, Fakulta informačních technologií, 2026.

Chtěl bych především poděkovat vedoucímu práce panu Ing. Pavlu Kubalíkovi, Ph.D. za odborné vedení a vstřícný přístup. Dále bych chtěl poděkovat své rodině za podporu v době, kdy tato práce vznikala.

Prohlášení

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

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů, zejména skutečnost, že České vysoké učení technické v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 citovaného zákona.

Prohlašuji, že jsem v průběhu příprav a psaní závěrečné práce použil nástroje umělé inteligence. Vygenerovaný obsah jsem ověřil. Stvrzuji, že jsem si vědom, že za obsah závěrečné práce plně zodpovídám.

V Praze dne 1. května 2026

Abstrakt

Tato práce se zabývá návrhem a realizací řídicího systému a podvozku pro cenově dostupnou robotickou platformu schopnou autonomního navádění a dálkového ovládání. Cílem bylo vytvořit modulární řešení založené na mikropočítači Raspberry Pi 5, které umožní sledování osoby pomocí metod strojového vidění a současně využití GPS a UWB technologie jako záložní možnosti navádění. Teoretická část se zaměřuje na výběr sběrnic pro vzájemnou komunikaci jednotlivých modulů a na prozkoumání možností lokace platformy v prostoru. Praktická část se posléze zaměřuje na výběr správných komponent a na softwarovou implementaci. Výstupem této práce je plně funkční platforma pro stavbu robota, která je schopna navádění v prostoru jak pomocí následování tak pomocí dálkového ovládání. V příloze lze nalézt kompletní zdrojový kód programu v jazyce Python, programy pro obsluhu UWB modulů v jazyce C a STL soubory obsahující díly podvozku.

Klíčová slova autonomní řízení, umělá inteligence, Raspberry Pi 5, GPS, UWB, strojové vidění, YOLO

Abstract

This thesis deals with the design and implementation of a control system and chassis for an affordable robotic platform capable of autonomous guidance and remote control. The goal was to create a modular solution based on the Raspberry Pi 5 microcomputer, which will enable person tracking using machine vision methods and at the same time use GPS and UWB technology as backup guidance options. The theoretical part focuses on the selection of buses for mutual communication of individual modules and on exploring the possibilities of platform location in space. The practical part then focuses on the selection of the right components and software implementation. The output of this thesis is a fully functional platform for building a robot, which is capable of guidance in space both by following and by remote control. In the appendix you can find the complete source code of the program in Python, programs for operating UWB modules in C, and STL files containing chassis parts.

Keywords autonomous control, artificial intelligence, Raspberry Pi 5, GPS, UWB, machine vision, YOLO

Obsah

1	Úvod	1
2	Existující řešení	2
2.1	SunFounder Robotické auto Zeus Car, Arduino UNO	2
2.2	SunFounder Raspberry Pi AI Car Kit	2
2.3	4Tronix M.A.R.S. Rover Robot pro Raspberry Pi Zero	3
2.4	Waveshare JetRacer 2GB AI Kit	3
3	Teorie	5
3.1	Autonomní řízení	5
3.1.1	Úrovně autonomního řízení	5
3.2	Neuronové sítě pro zpracování obrazu	6
3.2.1	Konvoluční neuronové sítě (CNN)	6
3.2.2	Trénování konvoluční neuronové sítě	7
3.2.3	Detekce objektů v obraze	7
3.2.4	Porovnání kvality	8
3.3	Mikropočítače Raspberry Pi	9
3.3.1	Parametry Raspberry pi 5	9
3.4	Mikrokontroléry	10
3.4.1	Desky Arduino	11
3.5	Satelitní navigace	11
3.6	Technologie UWB	12
3.6.1	Metody pro určení polohy	14
3.7	Sběrnice SPI	15
3.8	Sběrnice I2C	15
3.8.1	Adresace	16
3.8.2	Princip přenosu	16
3.9	Standard RS-485	18
3.10	Standard RS-232	18
3.11	Pulzně šířková Modulace	19
3.11.1	Řízení pomocí PWM	19
4	Analýza	21
4.1	Natáčení kamery	21
4.1.1	Servomotory	21
4.1.2	Ovládání servomotorů	22

4.2	Motory	23
4.2.1	Ovládání motorů	24
4.3	Akcelerátor AI	25
4.4	Napájení	26
4.5	UWB Moduly	28
4.6	GPS	29
4.7	Podvozek	30
5	Návrh řešení	32
5.1	Řešení podvozku	34
5.1.1	Prvotní návrh	34
5.1.2	Návrh pomocí 3D tisku	35
5.1.3	Stranová deska	35
5.1.4	Držák strany	39
5.1.5	Spodní deska	40
5.1.6	Držák pro kameru	40
5.1.7	Držák pro GPS a magnetometr	42
5.1.8	Shrnutí návrhu podvozku	43
5.2	Řešení UWB modulů	43
5.3	Řešení webové aplikace	50
5.3.1	Kostra webové aplikace	50
5.3.2	Grafické rozhraní	51
5.3.3	Obsluha a komunikace	51
5.3.4	Shrnutí webové aplikace	53
5.4	Řešení řídicí aplikace	54
5.4.1	Architektura aplikace	55
5.4.2	Zpracování obrazu a detekce objektů	57
5.4.3	Navádění	57
5.4.4	Řízení hardwaru	59
5.4.5	Webová komunikace	60
5.4.6	Inicializace a spuštění systému	61
5.4.7	Shrnutí řídicí aplikace	62
6	Testování	63
6.1	Testování UWB modulů	63
6.1.1	Měření vzdálenosti jednotlivých modulů	63
6.1.2	Měření úhlu natočení	64
6.2	Testování inference strojového vidění	66
6.3	Testování GPS a magnetometru	67
6.4	Testování webové aplikace	67
6.5	Testování celého systému	68
6.5.1	Mechanické a napájecí vlastnosti	68
6.5.2	Ověření metod navádění	68
6.6	Shrnutí kapitoly	68

Obsah	ix
7 Závěr	70
A Přílohy	72
Obsah příloh	78

Seznam obrázků

2.1	Obrázky již existujících řešení [1] [2] [3] [4]	4
3.1	Porovnání jednoúrovňové a dvouúrovňové metody [6]	8
3.2	Popis pinů mikropočítače Raspberry pi 5 [12]	10
3.3	Výpočet souřadnic pomocí GNSS [17]	12
3.4	Porovnání vysílání pomocí technologie UWB a klasických rádiových sítí [20]	13
3.5	Výpočet vzdálenosti pomocí metody time of flight [22]	14
3.6	Možné schéma zapojení [25]	16
3.7	Počátek I2C komunikace [24]	17
3.8	Průběh I2C komunikace [25]	17
3.9	Výpočet průměrného napětí [31]	20
4.1	Blokové schéma servo motoru [32]	21
4.2	Řízení dvou servomotorů pomocí driveru PCA9685 [35]	22
4.3	Motor typu n20 uložený s hřídelí v pouzdře [36]	24
4.4	Popis pinů na vstupech a výstupech H můstku [38]	25
4.5	Akcelerátor umělé inteligence [41]	26
4.6	Schéma zapojení napájení 2S [42]	27
4.7	UPS modul x1205 již přišroubovaný k Raspberry Pi 5 [43]	28
4.8	Modul BU01 v SMD provedení [49]	29
4.9	GPS modul L75G [50]	30
5.1	Základní schéma zapojení	33
5.2	Přenos pomocí ozubených kol v horní části a pomocí řemenu v dolní části	36
5.3	Finální návrh hřídele	37
5.4	Finální návrh stranové desky	38
5.5	Finální návrh disku kola	38
5.6	Finální návrh držáku strany	39
5.7	Základna pro držák na kameru	41
5.8	Konstrukce uchycení serv	42
5.9	Výsledný návrh držáku GPS	42
5.10	Výsledný návrh podvozku	43
5.11	Prvotní zapojení modulu BU01	44
5.12	Schéma zapojení modulu BU01	45

5.13	Schéma zapojení mikrokontroléru Arduino Nano	45
5.14	Schéma zapojení lineárního regulátoru ASM1117	45
5.15	Schéma zapojení integrovaného převodníku TXS0108E	46
5.16	Navržená deska plošných spojů pro obsluhu BU01	47
5.17	Vývojový diagram kalibračního algoritmu Antenna Delay.	48
5.18	Schéma zapojení převodníku MAX485	49
5.19	Upravená deska plošných spojů o převodník MAX485	50
5.20	Webová aplikace	54
5.21	Schéma architektury Producent–Konzument v aplikaci.	56
6.1	Porovnání skutečné vzdálenosti s naměřenou pro první modul	64
6.2	Porovnání skutečné vzdálenosti s naměřenou pro druhý modul	64
6.3	Srovnání naměřených sad dat.	65
6.4	Porovnání skutečného úhlu natočení s naměřeným	66
6.5	Porovnání naměřené vzdálenosti při zvolených úhlech	66
A.1	Kompletní podvozek	72

Seznam tabulek

2.1	Srovnání vybraných modelů aut s umělou inteligencí [1] [2] [3] [4]	4
-----	--	---

Seznam výpisů kódu

1	Zjednodušená ukázka hlavní části HTML	51
2	Zjednodušená ukázka odesílání ve formátu JSON	52
3	Ukázka normalizování souřadnic	53
4	Zjednodušená ukázka této sdílené struktury	55
5	Zjednodušený princip callbacku	57
6	Ukázka principu výpočtu otočení kamery	58
7	Ukázka principu načítání dat ze sériové linky	59
8	Ukázka funkce <code>set_motor()</code>	59
9	Ukázka obsluhy endpointu pro telemetrii	60
10	Ukázka obsluhy telemetrie využívající WebSocket	61

11	Ukázka funkce využívající WebSocket v souboru <code>control.js</code> náležícím webové aplikaci	61
12	Zjednodušená ukázka spuštění jednotlivých vláken	62

Seznam zkratek

AI	umělá inteligence
CAD	počítačem podporované navrhování
CNN	konvoluční neuronová síť
CPU	centrální procesorová jednotka
GNSS	globální navigační družicový systém
GPIO	univerzální vstupně-výstupní port
GPS	globální polohový systém
GPU	grafická procesorová jednotka
HTML	hypertextový značkovací jazyk
HTTP	hypertextový přenosový protokol
I ² C	sériová sběrnice Inter-Integrated Circuit
JSON	textový formát JavaScript Object Notation
MJPEG	formát Motion JPEG
NPU	neuronová procesorová jednotka
PCB	deska plošných spojů
PID	proporcionálně-integračně-derivační regulátor
PWM	pulzně šířková modulace
RS-485	sériový komunikační standard RS-485
SPI	sériové periferní rozhraní
UWB	ultraširokopásmová technologie
USB	univerzální sériová sběrnice
YOLO	detekční model řady You Only Look Once



Kapitola 1

Úvod

Umělá inteligence je dnes běžnou součástí každodenního života většiny lidí. S jejími aplikacemi se setkáváme jak přímo (například ve formě chatbotů), tak i nepřímo, například při zobrazování cílené reklamy na internetu. Rozmach výpočetní techniky umožňuje nasazení umělé inteligence také ve vestavných systémech, kde nachází uplatnění například i v dostupných dronových technologiích – ty dnes často nabízejí režim "follow me", kdy dron rozpozná osobu a autonomně ji následuje. Přesto je tato funkce většinou dostupná jen v komerčně dražších modelech, případně se týká pouze vybraných funkcí.

Problematika autonomního navádění prostřednictvím umělé inteligence je proto aktuální a zasluhuje další výzkum zejména v oblasti malých a dostupných systémů.

Cílem této práce je navrhnout, realizovat a ověřit funkčnost podvozku a řídicího systému založeného na mikropočítači Raspberry Pi 5, který umožní jak autonomní ovládání podvozku pomocí umělé inteligence, tak i možnost uživatelského zásahu. Součástí práce je také prozkoumání a využití záložních metod navádění, jako jsou UWB a GPS. Výsledný podvozek a řídicí systém by měl být snadno adaptovatelný na různé robotické platformy a tvořit základ pro další rozvoj autonomních navigačních systémů.

[illegible]

Existující řešení

Pro přehled současného stavu trhu s auty řízenými umělou inteligencí byly vybráni čtyři zástupci zobrazení na obrázku 2.1, kteří se pohybují ve srovnatelné cenové relaci, ale každý využívá odlišný mikropočítač pro řízení.

2.1 SunFounder Robotické auto Zeus Car, Arduino UNO

- Cena: 2 909 Kč
- Řízení pomocí mikropočítače Arduino UNO
- Obsahuje sensory pro detekci překážek a kameru pro rozpoznávání obličejů
- Nedisponuje dostatečným výkonem pro běh detekce na zařízení [1]

Mezi hlavní výhody patří robustní konstrukce, omnidirekční kola, která umožňují jízdu do všech směrů, a velké množství různých senzorů. Nevýhodou je nutnost připojení telefonu pro vzdálenou detekci objektů z kamery – Arduino UNO nedisponuje dostatečným výpočetním výkonem. Kamera není otočná, je tedy nutné otáčet celým autem pro sledování objektu. [1]

2.2 SunFounder Raspberry Pi AI Car Kit

- Cena: 2 228 Kč
- Řízení pomocí mikropočítače Raspberry Pi 5/4/3B+/3B/Zero 2W (není součástí balení)
- Obsahuje sensory pro detekci překážek a kameru pro rozpoznávání obličejů
- Disponuje dostatečným výkonem pro běh detekce na zařízení [2]

Nevýhodou je absence mikropočítače v základní sadě, což zvyšuje celkovou cenu při pořízení. Tento model však vyniká nejlepším poměrem cena/výkon z uvedených variant. Slabinou je nízká terénní dostupnost a absence systémů pro navádění, jako je GPS. [2]

2.3 4Tronix M.A.R.S. Rover Robot pro Raspberry Pi Zero

- Cena: 3 999 Kč
- Řízení pomocí mikropočítače Raspberry Pi Zero
- Nedisponuje kamerou, řídí se pouze ultrazvukovým senzorem
- Má vyšší terénní prostupnost [3]

Výhodou tohoto modelu je dobrá terénní prostupnost a schopnost otáčení na místě. Nevýhodou v porovnání s ostatními je možnost navádění pouze pomocí ultrazvukového senzoru. [3]

2.4 Waveshare JetRacer 2GB AI Kit

- Cena: 5 969 Kč
- Řízení pomocí mikropočítače Nvidia Jetson Nano 2GB (není součástí balení)
- Disponuje pouze kamerou
- Disponuje dostatečným výkonem pro běh detekce na zařízení [4]

Po zakoupení řídicího mikropočítače cena tohoto auta přesahuje 10 000 Kč, jediným senzorem je kamera. Kamera není otočná, a proto je nutné pro sledování objektu otáčet celým autem. Mezi pozitiva patří robustní podvozek a možnost vyšší rychlosti jízdy. [4]

Společnou nevýhodou výše uvedených aut je absence navádění pomocí GPS nebo UWB technologie. Není tedy možné zadat souřadnice cílové destinace, navigace je realizována pouze následováním čáry na zemi nebo sledováním objektu prostřednictvím kamery. [1] [2] [3] [4]

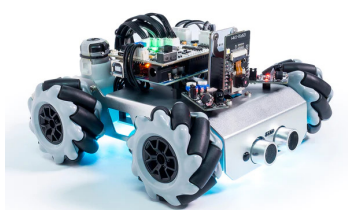
Shrnutí existujících řešení

Vybraná srovnávaná řešení demonstrují různé přístupy k softwarové i hardwarové stránce autonomního řízení vozidel v dané cenové kategorii. Zásadními omezeními současných modelů jsou zejména absence navigace pomocí GPS

nebo UWB a nutnost kompromisů mezi výpočetním výkonem, počtem senzorů a dalšími schopnostmi. Tabulka 2.1 uvádí základní technické parametry pro rychlejší srovnání:

Model	Cena	Počítač	Senzory	Otočná kamera
Zeus Car	2 909 Kč	Arduino UNO	senzory, kamera	ne
Raspberry Pi AI Car Kit	2 228 Kč (+RPI)	Raspberry Pi (volitelně)	senzory, kamera	ano
4Tronix M.A.R.S. Rover	3 999 Kč	Raspberry Pi Zero	ultrazvuk	ne
JetRacer 2GB AI Kit	5 969 Kč (+Jetson Nano)	Jetson Nano 2GB (volit.)	kamera	ne

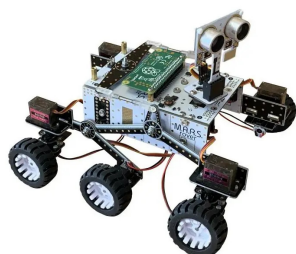
■ **Tabulka 2.1** Srovnání vybraných modelů aut s umělou inteligencí [1] [2] [3] [4]



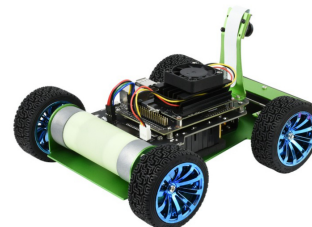
(a) SunFounder Robotické auto Zeus Car, Arduino UNO



(b) SunFounder Raspberry Pi AI Car Kit



(c) 4Tronix M.A.R.S. Rover Robot pro Raspberry Pi Zero



(d) Waveshare JetRacer 2GB AI Kit

■ **Obrázek 2.1** Obrázky již existujících řešení [1] [2] [3] [4]



Kapitola 3

Teorie

3.1 Autonomní řízení

Autonomní řízení je technologie, která umožňuje strojům, vozidlům nebo robotům pohybovat se a orientovat se v prostoru bez lidského zásahu, pomocí senzorů (radar, lidar, kamery) a pomocí počítačových systémů k detekci okolí a plánování trasy. Využívá se v samořízených automobilech, v zemědělských strojích pro precizní práci (např. setí, sklizeň) nebo v logistických AGV vozících, přičemž cílem je zvýšení bezpečnosti, efektivity a snížení lidské chyby a námahy. Princip autonomního řízení spočívá v načtení dat senzory, následném zpracování dat například pomocí umělé inteligence a následně provedením změn aktuátory (např. motory). [5]

3.1.1 Úrovně autonomního řízení

Existují různé stupně, které definovala organizace SAE (Society of Automotive Engineers) a liší se mírou nezbytné aktivity ze strany řidiče:

- Stupeň 0 (Žádná automatizace): Řidič dělá vše, systém jen varuje (např. před námrazou).
- Stupeň 1 (Asistence řidiče): Systém pomáhá s jednou funkcí (např. adaptivní tempomat nebo asistent jízdy v pruhu).
- Stupeň 2 (Částečná automatizace): Systém řídí zrychlování, brzdění i řízení v určitých situacích (např. při zácpě), ale řidič musí neustále sledovat provoz a být připraven zasáhnout (tzv. "hands-off", ale "eyes-on").
- Stupeň 3 (Podmíněná automatizace): Vozidlo řídí v omezeném prostředí (např. na dálnici), řidič nemusí sledovat provoz, ale musí být schopen převzít řízení na vyžádání systému (tzv. "eyes-off").

- Stupeň 4 (Vysoká automatizace): Vozidlo řídí autonomně téměř za všech okolností v definované oblasti (např. v určitém městě) a nepožaduje zásah řidiče, kromě extrémních podmínek (počasí, stav silnice).
- Stupeň 5 (Plná automatizace): Vozidlo je schopno řídit za všech podmínek a na jakémkoli místě. Řidič není vůbec potřeba, jen zadá cíl (např. autonomní taxíky). [5]

3.2 Neuronové sítě pro zpracování obrazu

Neuronové sítě zažívají v poslední době velký rozmach. K oblastem, kde v současnosti nacházejí uplatnění, patří i zpracování obrazu pro strojové vidění a následné hodnocení kvality průmyslových výrobků. Dominantní technologií jsou konvoluční neuronové sítě (CNN – Convolutional Neural Networks), které napodobují způsob, jakým lidský zrak zpracovává informace, a efektivně rozpoznávají vzory, hrany a textury v obrazových datech. [6]

3.2.1 Konvoluční neuronové sítě (CNN)

Konvoluční neuronové sítě jsou specializovaným typem hlubokých umělých neuronových sítí, které jsou navrženy pro efektivní zpracování dat s mřížkovou strukturou, jako jsou obrázky (2D mřížka pixelů) nebo videa. [6]

Jde o specializovaný typ umělé neuronové sítě, který zhruba napodobuje systém lidského vidění. Na rozdíl od plně propojených sítí využívají CNN lokální receptivní pole a sdílení vah, díky čemuž výrazně snižují počet parametrů a lépe zachovávají prostorové vztahy v obraze. Základní stavební prvky CNN tvoří zejména:

- **Konvoluční vrstvy** – aplikují sadu filtrů (jader) nad vstupní maticí a vytvářejí tzv. mapy příznaků (*feature maps*). Parametry konvoluce zahrnují velikost jádra, krok (*stride*) a případné doplnění okrajů (*padding*).
- **Aktivační funkce** – typicky ReLU, která zavádí nelinearitu a umožňuje síti učit se komplexní vztahy.
- **Pooling** (subsampling) – např. max pooling, který redukuje prostorové rozměry map příznaků a zvyšuje odolnost vůči drobným posunům a šumu.
- **Normalizační vrstvy** – často *batch normalization*, která stabilizuje a urychluje trénink.
- **Klasifikační část** – na konci sítě bývají plně propojené vrstvy (nebo globální průměrování) a výstupní funkce, typicky *softmax* pro vícetřídní klasifikaci. [6]

Operace vynásobení hodnot vstupních pixelů váhami filtru a jejich sčítání se nazývá „konvoluce“ (odtud název konvoluční neuronová síť). V praxi se v implementacích často používá korelace; terminologie „konvoluce“ se však v literatuře běžně zachovává. [6]

Při zadání obrázku do konvoluční sítě každá z jejích vrstev vygeneruje několik map příznaků (aktivačních map), které zvýrazňují relevantní vlastnosti obrazu. První vrstvy obvykle detekují základní prvky (vodorovné, svislé a úhlopříčné hrany), následné vrstvy pak skládají tyto jednoduché rysy do složitějších struktur (rohy, textury) až po abstraktní reprezentace na úrovni objektů nebo částí objektů. Tato hierarchická extrakce příznaků je jedním z hlavních důvodů, proč jsou CNN velmi úspěšné v úlohách jako klasifikace obrazu, detekce objektů nebo segmentace. [6]

3.2.2 Trénování konvoluční neuronové sítě

Jednou z velkých výzev při vývoji CNN je úprava vah jednotlivých neuronů, aby byly ze snímků získány správné rysy. Proces úpravy těchto vah se nazývá „trénink“ neuronové sítě. Na začátku začíná CNN s náhodnými váhami. Během tréninku vývojáři poskytují neuronové síti soubor dat trénovacích obrázků s poznámkami s odpovídajícími třídami (kočka, pes, židle atd.). Síť zpracuje každý obrázek s jeho náhodnými hodnotami a poté porovná jeho výstup se správným štítkem obrázku. Pokud výstup sítě neodpovídá označení - což je pravděpodobný případ na začátku tréninkového procesu - provede malou úpravu vah svých neuronů, aby příště až uvidí stejný obraz, její výstup byl trochu blíž ke správné odpovědi. [7]

V praxi se pro trénování nejčastěji používá metoda zpětného šíření chyby (*backpropagation*) v kombinaci s optimalizátory typu SGD nebo Adam. K omezení přeučení (*overfitting*) se používají techniky jako augmentace dat (otáčení, ořez, změna jasu), regularizace a *dropout*. Z hlediska výpočetní náročnosti je konvoluce dominantní operací, proto se CNN typicky trénují i provozují na GPU nebo specializovaných akcelérátorech. [8]

3.2.3 Detekce objektů v obraze

Detekce objektů v obraze je klíčovou úlohou počítačového vidění, která spočívá v lokalizaci (nalezení polohy) a klasifikaci (určení třídy) objektů. Výsledkem algoritmu bývá sada ohraničujících rámečků (*bounding boxes*) a k nim příslušné třídy a skóre (pravděpodobnosti). Moderní metody se často dělí podle architektury na jednoúrovňové (*one-stage*) a dvouúrovňové (*two-stage*) detektory. [6]

Dvouúrovňové metody se zaměřují na vysokou přesnost, často za cenu vyšší výpočetní náročnosti. Proces probíhá ve dvou krocích:

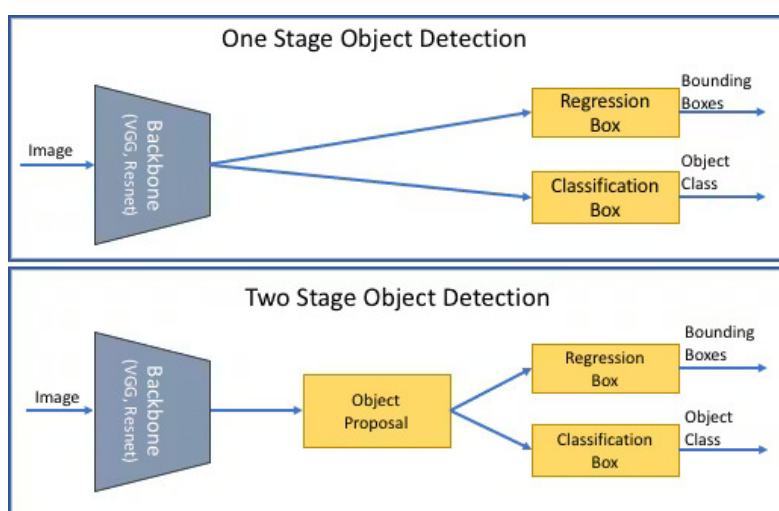
- Region Proposal (Návrh oblasti): Nejprve se vygenerují oblasti zájmu (ROI), kde by se mohl nacházet objekt.

- Klasifikace a lokalizace: V druhé fázi se tyto oblasti klasifikují a zpřesňují se ohraničující rámečky (bounding boxes). [6]

Příkladem jsou metody R-CNN, Fast R-CNN, Faster R-CNN (využívá *Region Proposal Network* – RPN) nebo Mask R-CNN (rozšíření o instanční segmentaci). Tyto metody jsou využívány zejména tehdy, když aplikace vyžaduje vysokou přesnost a nevadí pomalejší inference (např. analýza lékařských snímků). [6]

Jednoúrovňové metody analyzují celý obraz v jediném průchodu (end-to-end), což umožňuje velmi rychlou detekci. Detekci definují jako regresní problém, kdy přímo z pixelů obrazu předpovídají souřadnice ohraničujících rámečků a pravděpodobnosti tříd. [6]

Příkladem jsou metody YOLO (*You Only Look Once*), SSD (*Single Shot MultiBox Detector*) nebo RetinaNet. Tyto přístupy se často používají v reálných aplikacích, ve zpracování videa, na mobilních zařízeních a obecně v *edge AI*. [6]



■ Obrázek 3.1 Porovnání jednoúrovňové a dvouúrovňové metody [6]

3.2.4 Porovnání kvality

Pro srovnání kvality detektorů se typicky používají metriky jako *IoU* (*Intersection over Union*), *precision/recall* a zejména *mAP* (*mean Average Precision*) pro zvolený práh *IoU*. Při zpracování výstupů sítě se dále používá potlačení překryvů (*Non-Maximum Suppression*, NMS), které odstraní duplicitní detekce téhož objektu. V praxi je nutné vždy volit kompromis mezi přesností a rychlostí, který je dán požadavky konkrétní aplikace a dostupným výpočetním výkonem. [9]

3.3 Mikropočítače Raspberry Pi

Mikropočítače, známé také jako jednodeskové počítače (SBC: Single-Board Computers), představují revoluční vývoj v oblasti elektroniky a výpočetní techniky. Jsou to malá, ale plně funkční počítačová zařízení, která mají navzdory své skromné velikosti impozantní výpočetní potenciál. Na rozdíl od klasických stolních počítačů nebo notebooků, které zabírají mnohem více místa a jsou často mnohem dražší, jsou mikropočítače integrovány do jednoho kusu laminátu. Právě miniaturizace a jednodeskový design činí tato zařízení tak jedinečnými. Malá velikost mikropočítačů neznamena nižší výkon. Zařízení je schopno zvládnout širokou škálu aplikací, od jednoduchých kancelářských úkolů až po pokročilé projekty v oblasti elektroniky, robotiky nebo dokonce umělé inteligence. Velkou výhodou je jejich přenosnost a nízká cena, díky čemuž jsou velmi oblíbené u hobbyistů, vědců, inženýrů a učitelů. [10]

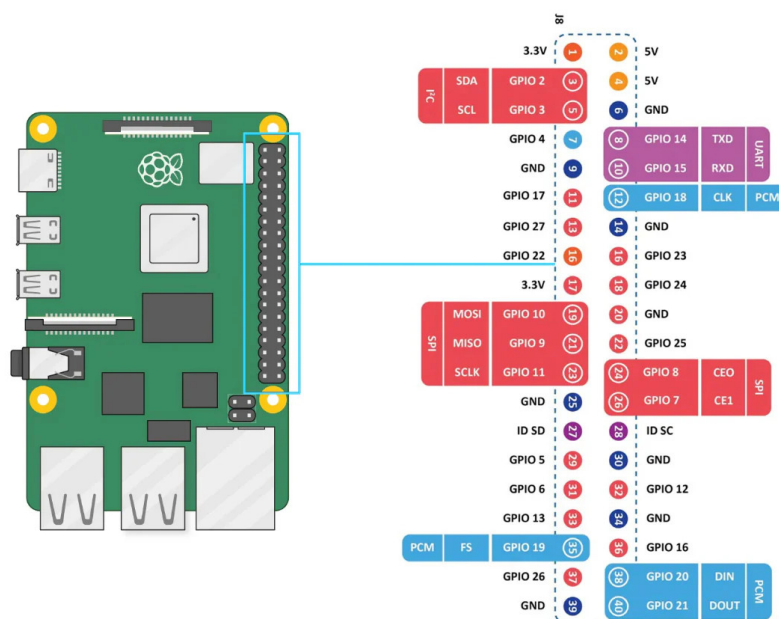
Mikropočítače RPI, celým názvem Raspberry Pi, jsou malá, kompaktní zařízení, která v sobě spojují všechny prvky potřebné pro jednodeskový počítač. Zařízení velikosti dlaně je vybaveno procesorem, který je obvykle variantou jádra ARM spolu s integrovanou grafickou jednotkou, paměť RAM pro ukládání aktuálně zpracovávaných informací, vstupní/výstupní porty pro připojení externích periférií, jako je myš, klávesnice a monitor, konektor pro paměťovou kartu, na které je uložen spuštěný systém, a několik dalších užitečných prvků včetně modulu pro bezdrátovou komunikaci a univerzálního konektoru GPIO. Operační systémy provozované na mikropočítačích jsou obvykle založeny na Ubuntu, které podporuje většinu populárních aplikací. [10]

Zařízení, jako je Raspberry Pi, jsou obvykle velmi energeticky úsporná, což umožňuje jejich nepřetržitý provoz při minimálním spotřebování energie, často i při napájení akumulátory. Díky tomu jsou ideální jako platforma pro nejrůznější mobilní projekty, od domácí automatizace a hardwaru internetu věcí až po výuku programování a experimentování s elektronikou. [10]

3.3.1 Parametry Raspberry pi 5

Mikropočítač Raspberry pi 5 je aktuálně nejnovější verzí Raspberry pi.

- Procesor: Broadcom BCM2712
- Architektura: ARMv8.2-A 64-bit
- Počet jader: 4
- Frekvence: 2,4GHz
- Nabízí se ve variantách s 2GB / 4GB / 8GB / 16GB RAM
- Disponuje Wifi, bluetooth
- Sběrnice: UART, I2C, SPI, PCIe 2.0 [11]



■ **Obrázek 3.2** Popis pinů mikropočítače Raspberry pi 5 [12]

3.4 Mikrokontroléry

Mikrokontrolér je integrovaný obvod určený pro řízení vestavných systémů. Na jednom čipu obvykle obsahuje procesorové jádro, paměť programu, pracovní paměť i vstupně-výstupní periferie, například časovače, sériová rozhraní nebo analogově-digitální převodníky. Oproti běžnému osobnímu počítači nebo mikropočítači je mikrokontrolér navržen pro vykonávání úzce zaměřených úloh, typicky s nízkou spotřebou, malými rozměry a jednoduchým hardwarem. [13]

Mikrokontroléry se používají všude tam, kde je potřeba spolehlivě a opakovaně vykonávat konkrétní úlohu v reálném čase. V praxi mohou řídit senzory, komunikovat s dalšími obvody, generovat PWM signály, číst stav vstupů nebo obsluhovat komunikační sběrnice. Jejich výhodou je jednoduché nasazení, nízká cena a proto jsou vhodné pro úlohy, kde není potřeba vysoký výpočetní výkon ani plnohodnotný operační systém. [14]

V kontextu této práce je vhodné odlišovat mikrokontrolér od mikropočítače. Mikropočítač Raspberry Pi 5 slouží jako hlavní výpočetní jednotka systému a zajišťuje náročnější úlohy, například zpracování obrazu, běh webové aplikace a rozhodovací logiku. Naproti tomu mikrokontroléry jsou využity jako pomocné řídicí prvky pro dílčí úlohy, u nichž je důležitá jednoduchost, nízká cena a snadná komunikace s periferiemi.

3.4.1 Desky Arduino

Arduino je platforma vývojových desek založených na mikrokontrolérech, která byla navržena s důrazem na jednoduché prototypování a výuku. Typická deska Arduino kombinuje samotný mikrokontrolér, napájecí část, programovací rozhraní a vyvedené vstupně-výstupní piny do jednoho snadno použitelného celku. Díky tomu lze mikrokontrolér rychle nasadit bez nutnosti navrhovat vlastní programátor nebo složitou podpůrnou elektroniku. [15]

Výhodou desek Arduino je rozsáhlý ekosystém knihoven, přehledné a jednoduché vývojové prostředí a široká dostupnost modulů a příkladů zapojení. To je důležité zejména při rychlém ověřování návrhu a při vývoji specializovaných pomocných částí systému. Desky Arduino se proto často používají tam, kde je potřeba rychle obsloužit senzory, komunikační moduly nebo jednoduché akční členy. [15]

3.5 Satelitní navigace

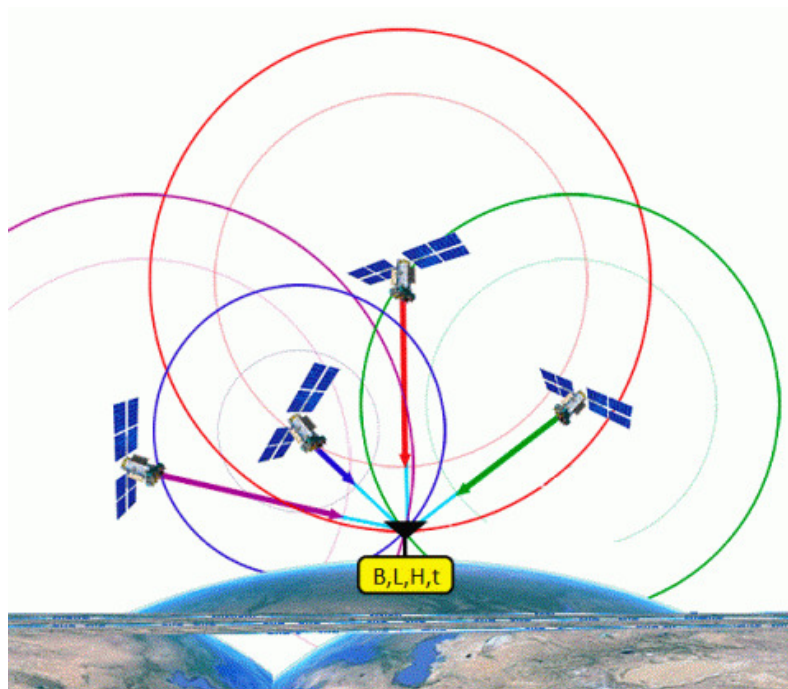
Satelitní navigace či globální družicový polohový systém (zkratka GNSS z anglického Global Navigation Satellite System) je služba umožňující za pomoci družic autonomní prostorové určování polohy s celosvětovým pokrytím. Uživatelé této služby používají malé elektronické rádiové přijímače, které na základě odeslaných signálů z družic umožňují vypočítat jejich polohu s přesností na desítky až jednotky metrů, geodetické a zemědělské aplikace používají korekční data (např. ze sítě CZEPOS), čímž dosahují centimetrové přesnosti pro navádění zemědělských strojů, staveb a mapování. [16]

Jedná se o dálkoměrný systém, tj. družice vysílají navigační zprávu, kde uvádějí (kromě jiného) své označení, polohu a čas vyslání. Přijímač, jehož poloha je určována, musí přijmout tyto signály alespoň od čtyř různých družic. Pro každou z družic lze z rozdílu času vyslání signálu družicí a přijetí signálu přijímačem vypočítat jejich vzájemnou vzdálenost, což ve spojení se znalostí polohy družice tvoří kulovou plochu. V průsečíku kulových ploch se nachází přijímač, resp. lze takto určit jeho souřadnice X, Y, Z. V principu by stačily tři družice, ale přijímač nemá ani zdaleka tak přesné hodiny, jak by bylo třeba a proto je nutné počítat čtvrtou neznámou – opravu hodin přijímače. Čím více signálů družic je zachyceno, tím je výsledek přesnější. Důležitou roli hraje z hlediska přesnosti konfigurace družic, např. pokud jsou viditelné družice seřazeny v jedné přímce (v dlouhé úzké ulici s vysokými domy po obou stranách), v podélném směru bude přesnost odpovídající, v příčném velmi špatná. [16]

Jako globální systém s celosvětovým pokrytím funguje:

- Americký GPS (Global Positioning System): Přesnost určování polohy se pohybuje od 500 do 30 centimetrů.
- Ruský GLONASS: Deklarovaná horizontální přesnost se pohybuje mezi 2,8 a 10 metry.

- Evropský Galileo: Deklarovaná přesnost je menší než jeden metr.
- čínský BeiDou: Přesnost se pohybuje kolem 2,6 až 3,6 metrů pro veřejné použití. [16]



■ Obrázek 3.3 Výpočet souřadnic pomocí GNSS [17]

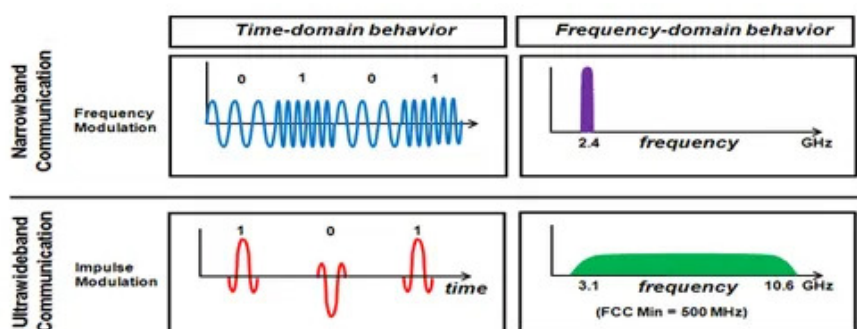
3.6 Technologie UWB

Technologie UWB (Ultra-Wideband) je moderní bezdrátová metoda komunikace, která vyniká přenosem dat v širokém frekvenčním pásmu – většinou až 500 MHz. Takto rozsáhlé pásmo je ve srovnání s klasickými technologiemi (například WiFi nebo Bluetooth, které využívají pásma pouze 20–80 MHz) výrazně širší. Díky tomu může UWB přenášet velmi vysoké objemy dat, dosahovat větších rychlostí a lépe fungovat v prostředí s rušením. Zásadní je ale i další přednost: díky krátkým a širokopásmovým pulzům lze UWB využít i k přesnému měření vzdálenosti a směru mezi zařízeními. [18]

Jednou z hlavních výhod UWB je jeho schopnost spolehlivě fungovat nejen v otevřeném prostoru, ale i v budovách – signál dokáže poměrně snadno procházet překážkami, například zdmi. Dosah bývá řádově jednotky až desítky metrů, přičemž technologie zajišťuje odolnost vůči rušení jiných zařízení v okolí. Díky velmi nízkému vyzařovanému výkonu UWB signálu není prakticky

možné způsobit rušení běžných mobilních a WiFi sítí; jiná zařízení vnímají tento signál jen jako šum. [19]

UWB technologie vysílá krátké pulzy v řádu nanosekund, na rozdíl od klasických rádiových sítí, které spoléhají na frekvenční modulaci sinusových vln. Právě díky tomuto způsobu lze docílit extrémní přesnosti v určování pozice a vzdálenosti mezi dvěma zařízeními. Výrobci uvádí, že přesnost může být i jen několik centimetrů. To je jeden z hlavních důvodů, proč je UWB vhodná například pro lokalizaci předmětů, určování polohy v rámci chytrých domácností nebo bezpečné odemýkání automobilu chytrým telefonem. [19]



Obrázek 3.4 Porovnání vysílání pomocí technologie UWB a klasických rádiových sítí [20]

Vysoká přesnost lokalizace není jedinou výhodou. UWB je také energeticky velmi úsporná a lze ji implementovat do miniaturních čipů i s integrovanou anténou. Taková zařízení mohou fungovat dlouhé měsíce či roky na malou baterii, což rozšiřuje možnosti použití v internetu věcí (IoT), sledovacích systémech nebo průmyslové automatizaci. [19]

Uplatnění technologie UWB se rychle rozrůstá. Už nyní se tato technologie nachází v některých chytrých telefonech (například Apple a Samsung) a umožňuje přesné sdílení souborů nebo hledání ztracených zařízení. Dále lze technologii využívat pro automatické otevírání aut, zabezpečené přihlašování či ještě bezpečnější bezkontaktní platby. Velké automobilky (například Audi, BMW, Ford, Kia nebo Hyundai) už pracují na implementaci UWB do svých produktů. Vedle toho může mít technologie UWB klíčový význam v oblasti chytré domácnosti a rychlého, zabezpečeného spojování zařízení. [19]

Ačkoli je tato technologie stále v počátcích svého rozšíření, její potenciál je obrovský. Postupně může změnit způsob, jakým lokalizujeme, komunikujeme i zabezpečujeme elektronická zařízení v běžném životě – od sledování předmětů přes průmyslové aplikace až po zabezpečení plateb a osobních údajů. [19]

3.6.1 Metody pro určení polohy

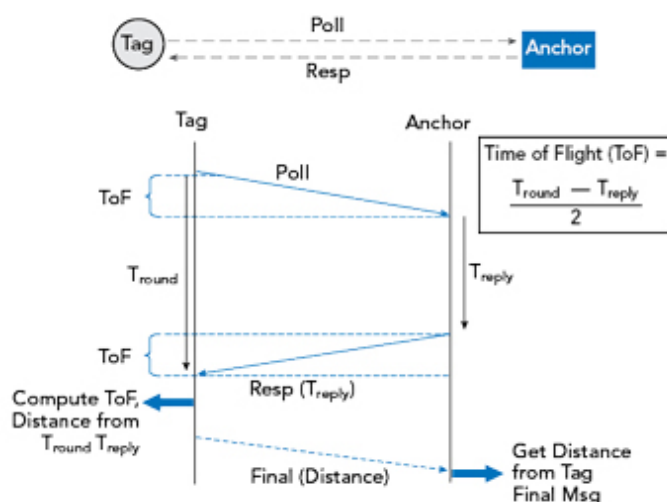
Existuje několik základních metod, jak pomocí technologie UWB určit přesnou polohu objektu. Všechny pracují s principem měření času, za který signál dorazí od vysílače k přijímači. Nejčastěji se používají tyto tři způsoby:

1. Time of Arrival (ToA, známá rovněž pod názvem Time of Flight – ToF): Tento princip spočívá v přímém sledování časového intervalu, jenž uplyne od okamžiku vyslání signálu až do momentu jeho detekce na straně příjemce. Na základě obecně známé rychlosti šíření signálu v daném prostředí (v tomto případě rychlosti světla ve vakuu či vzduchu) se vypočítá separace mezi vysílačem a přijímačem dle vzorce vzdálenost = rychlost × čas. Klíčovou výzvou této metody zůstává požadavek na extrémně přesnou synchronizaci vnitřních hodin obou prvků systému, aby nedocházelo ke zkreslení vypočtené polohy. [21]

2. Time Difference of Arrival (TDoA): Pro přesné určení polohy ve 2D prostoru je potřeba alespoň tří přijímacích stanic (tzv. kotev), ve 3D dokonce čtyř. Tento systém porovnává rozdíly v čase přijetí téhož signálu více přijímači (kotvami) umístěnými v prostoru. Z těchto vzdáleností je pak možné pomocí trilaterace určit přesnou pozici sledovaného objektu. [21]

3. Round Trip Time (RTT): V tomto případě signál putuje od vysílače k přijímači, který jej ihned odešle zpět. Měří se doba celé cesty (tam a zpět), čímž lze vykompenzovat potřebu přesné synchronizace hodin mezi zařízeními. Z vypočtené doby a rychlosti šíření signálu lze určit vzdálenost. [21]

Každá metoda má své výhody i nevýhody – ToA a TDoA více závisí na časové synchronizaci, RTT je zase náročnější na výpočetní výkon. Ve výsledku ale technologie UWB umožňuje dosahovat velmi vysoké přesnosti lokalizace, často v jednotkách až desítkách centimetrů. [21]



■ Obrázek 3.5 Výpočet vzdálenosti pomocí metody time of flight [22]

3.7 Sběrnice SPI

SPI (Serial Peripheral Interface) je jednoduchá synchronní sériová sběrnice, která umožňuje rychlou komunikaci mezi mikropočítačem (řadičem, tzv. master) a připojenými periferními zařízeními (tzv. slave). Nejčastěji se používá pro přenos dat mezi procesorem a dalšími čipy, například paměťmi, ADC/DAC převodníky, senzory, displeji nebo moduly pro komunikaci.

Základní architektura SPI zahrnuje čtyři signální vodiče:

- **SCK (Serial Clock):** Hodinový signál generovaný masterem, určuje synchronizaci přenosu.
- **MOSI (Master Out, Slave In):** Linka, po které master posílá data do slave zařízení.
- **MISO (Master In, Slave Out):** Linka, po které slave zařízení posílá data zpět do mastera.
- **SS (Slave Select, případně CS – Chip Select):** Signál pro výběr konkrétního slave. Každé zařízení typu slave má vlastní drát SS; aktivací vybraného se tento slave připojí na sběrnici, ostatní zůstávají neaktivní.

Komunikace v rámci SPI je plně duplexní, což znamená, že data mohou proudit současně v obou směrech (po MOSI i MISO). Master vždy řídí přenos — generuje hodinový signál i aktivuje konkrétní slave pomocí jeho SS linky.

Výhody standardu SPI:

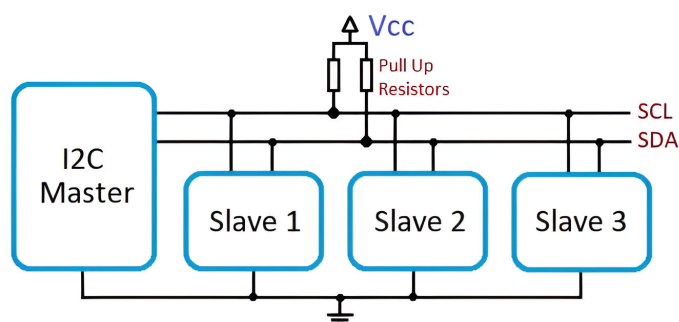
- vysoká rychlost přenosu (řádově desítky Mbit/s),
- jednoduchá implementace v hardware i software,
- plná duplexnost přenosu,
- možnost adresovat více zařízení na jedné sběrnici (každé má vlastní SS/CS vstup).

Nevýhodou je vyšší počet potřebných vodičů proti například I2C sběrnici a absence adresace přímo v protokolu (z tohoto důvodu je nutné každé periférii vyhradit vlastní signál SS/CS). SPI je díky své jednoduchosti a rychlosti velmi oblíbený pro připojení zařízení, kde je vyžadována rychlá a spolehlivá výměna dat. [23]

3.8 Sběrnice I2C

I2C je interní datová sběrnice sloužící pro přenos dat mezi jednotlivými integrovanými obvody. Na rozdíl od SPI sběrnice jsou potřeba pouze 2 vodiče na místo 3. Komunikace je ale díky tomu pouze poloduplexní (v jeden okamžik mohou

data téct pouze jedním směrem). Zároveň nejsou potřeba žádné další vodiče pro výběr obvodu. Využívá vodiče SDA (serial data) a SCL (serial clock). Jednotlivé obvody jsou synchronizovány pomocí vodiče SCL. Přenosová rychlost je standardně 100 kHz. Pokud jsou vyšší nároky na přenosovou rychlost, může být zvýšena na 400 kHz nebo 1 MHz. Ne všechny obvody ale tuto vyšší rychlost podporují. Oba sběrnice vodiče musí být při nečinnosti v logické 1. Toho lze docílit přidáním pull-up rezistorů. [24], [25]



■ Obrázek 3.6 Možné schéma zapojení [25]

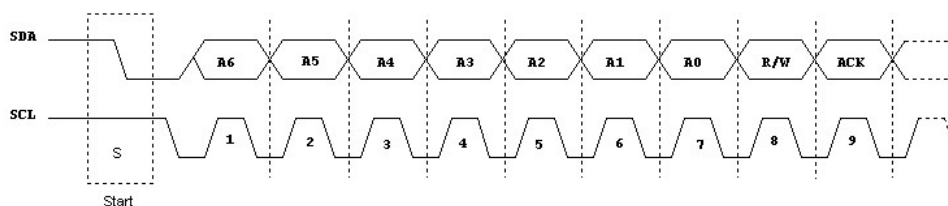
3.8.1 Adresace

Pro volbu zařízení komunikujícího pomocí sběrnice I2C používáme adresy. Díky tomu nemusíme vést selektující vodič ke každému obvodu. To především u mikrokontrolérů výrazně optimalizuje nároky na počet vstupně-výstupních pinů a celkově zjednodušuje výsledné zapojení. Na jednu sběrnici může být připojeno více integrovaných obvodů. V základní verzi jsou obvody adresovány 7bitově a v rozšířené verzi 10bitově. To umožňuje připojení 128 respektive 1024 čipů s různou adresou na jednu společnou sběrnici. V praxi jsou však tato čísla podstatně nižší, protože adresu čipu většinou nelze měnit plnými 7 (10) bity ale třeba jen třemi. U některých čipů jí nelze měnit vůbec. Takový čip může pak být na sběrnici přítomen pouze jeden. [24], [25]

3.8.2 Princip přenosu

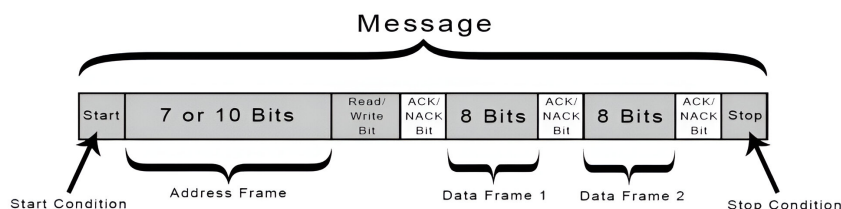
Jeden z integrovaných obvodů je nastaven do režimu master a ostatní do režimu slave. Veškeré řízení sběrnice má na starosti zařízení nakonfigurované do režimu master. V jednu chvíli může jako master pracovat pouze jediné zařízení, čímž je zaručeno, že na sběrnici nebude docházet ke kolizím. Ostatní uzly, které pracují v režimu slave, nemohou sběrnici řídit a dokonce ani nemají možnost samy žádat o vysílání či příjem dat. Master při jakémkoli přenosu generuje hodinový signál na vodiči SCL. Když jeden čip vysílá, přijímají všechny ostatní a pouze podle adresy určují, zda jsou data určena jim. [24], [25]

Komunikaci vždy zahajuje master a to tak, že sníží úroveň na datovém vodiči SDA na logickou nulu, zatímco SCL si po určitou dobu udržuje stav logické jedničky (tato doba je závislá na zvolené přenosové rychlosti). Tento stav, který se označuje termínem start bit, rozpoznávají všechny uzly připojené na sběrnici. Ihned po vyslání start bitu začne master vysílat adresu uzlu, se kterým si přeje komunikovat. Musí také určit, jestli chce do zařízení vysílat, anebo z něj přijímat. To určí R/W bitem který je součástí adresy. Standardně se k adresaci používá 7 bitů a poslední bit je R/W bit. Plná adresa tedy tvoří jeden byte. [24], [25]



Obrázek 3.7 Počátek I2C komunikace [24]

Každý bit musí mít ustálenou logickou hodnotu ve chvíli, kdy přijde náběžná hrana hodin. Po přenosu všech osmi bitů (sedm bitů adresy + osmý bit s určením směru přenosu dat) provede každý uzel slave porovnání přijaté adresy se svoji vlastní adresou. Vybraný uzel (jen on) totiž musí potvrdit, že se na sběrnici skutečně nachází. Potvrzení (bit ACK – acknowledge) se v devátém cyklu hodin pošle zpět na zařízení typu master. Příjem logické nuly značí, že adresovaný uzel skutečně existuje a je možné s ním zahájit komunikaci, logická jednička naopak může znamenat, že uzel s danou adresou buď neexistuje, nebo je z nějakého důvodu odpojen či prostě není z různých příčin připraven komunikovat. Samotná data se standardně posílají po jednom byte, za kterým následuje také potvrzení. Pokud chceme změnit datový tok mezi master obvodem a slave obvodem, musíme ukončit komunikaci a vybrat zařízení znovu se správně nastaveným READ/WRITE bitem. [24], [25]



Obrázek 3.8 Průběh I2C komunikace [25]

3.9 Standard RS-485

EIA-485, běžně označovaný jako RS-485, je standardem sériové komunikace určený především pro odolný přenos dat v průmyslovém prostředí. Oproti jednodušším sériovým rozhraním je navržen tak, aby umožňoval komunikaci na delší vzdálenosti a současně byl méně citlivý na elektromagnetické rušení. Právě z tohoto důvodu se RS-485 často používá v distribuovaných řídicích systémech, průmyslové automatizaci nebo při propojení více zařízení v jednom komunikačním segmentu. [26]

Základní vlastností RS-485 je diferenciální přenos. Data nejsou reprezentována napětím jednoho vodiče vůči zemi, ale rozdílem napětí mezi dvojicí vodičů, obvykle označovaných A a B. Toto řešení je výhodné zejména proto, že rušení indukované do obou vodičů se projeví přibližně stejně a přijímač jej díky rozdílovému vyhodnocení do značné míry potlačí. V porovnání s rozhraním RS-232 tak RS-485 poskytuje vyšší odolnost proti rušení a lepší použitelnost v elektricky náročnějším prostředí. [26]

V praxi se nejčastěji používá dvouvodičová varianta RS-485, která pracuje v poloduplexním režimu. To znamená, že přenos probíhá po jedné dvojici vodičů oběma směry, nikoli však současně. Zařízení proto musí řídit směr komunikace a zajistit, aby v jednom okamžiku vysílal pouze jeden uzel. Samotný standard RS-485 toto řízení neřeší; určuje pouze fyzickou vrstvu přenosu. Způsob střídání vysílačů tedy musí zajistit nadřazený komunikační protokol. [26]

Přenos znaků na logické úrovni odpovídá běžné asynchronní sériové komunikaci. Používají se 7- nebo 8bitové rámce doplněné start bitem, jedním nebo více stop bity a případně paritou. Z tohoto pohledu je formát přenášených dat podobný jako u RS-232; zásadní rozdíl spočívá zejména ve fyzické vrstvě a způsobu reprezentace logických úrovní. [26]

Další výhodou RS-485 je možnost vytvoření vícebodové sběrnice. Na jednu linku lze připojit více zařízení, která sdílejí stejnou komunikační dvojici vodičů. Díky tomu je možné relativně jednoduše budovat distribuované systémy s několika uzly bez nutnosti samostatného kabelového propojení mezi každou dvojicí zařízení. V kontextu této práce je důležitá především kombinace delšího dosažitelného vedení, odolnosti vůči rušení a jednoduché implementace pomocí běžně dostupných převodníků, například MAX485. [26]

3.10 Standard RS-232

RS-232 je starší standard asynchronní sériové komunikace, který byl dlouhou dobu běžnou součástí osobních počítačů, měřicí techniky i průmyslových zařízení. Na rozdíl od RS-485 používá nesymetrický přenos, tedy vyhodnocování napětí jednotlivého signálu vůči společné zemi. To zjednodušuje zapojení, ale současně snižuje odolnost proti rušení a omezuje praktickou přenosovou vzdálenost. [27]

V rámci předkládané práce hraje standard RS-232 (respektive jeho implementace přes sériovou linku) klíčovou roli zejména při zajištění datové výměny mezi mikrokontrolérem Arduino Nano a nadřazeným výpočetním systémem. Samotné Arduino využívá pro vnitřní komunikaci hardwarové rozhraní UART, které je na desce integrováno společně s převodníkem na sběrnici USB. Po fyzickém propojení Arduina s počítačem se v operačním systému vytvoří virtuální sériový port (COM). To umožňuje softwarovým aplikacím přistupovat k mikrokontroléru standardizovaným způsobem a využívat běžné knihovny pro sériovou komunikaci.” [28]

Je vhodné poznamenat, že na fyzické úrovni se v tomto případě nejedná o plnohodnotnou linku RS-232 s tradičními napěťovými úrovněmi, ale spíše o USB rozhraní emulující sériový port. Z hlediska programátora nebo uživatele je však způsob práce velmi podobný: nastavuje se přenosová rychlost, čtou se a zapisují bajty a komunikace probíhá ve formě sériového datového toku. [28]

3.11 Pulzně Šířková Modulace

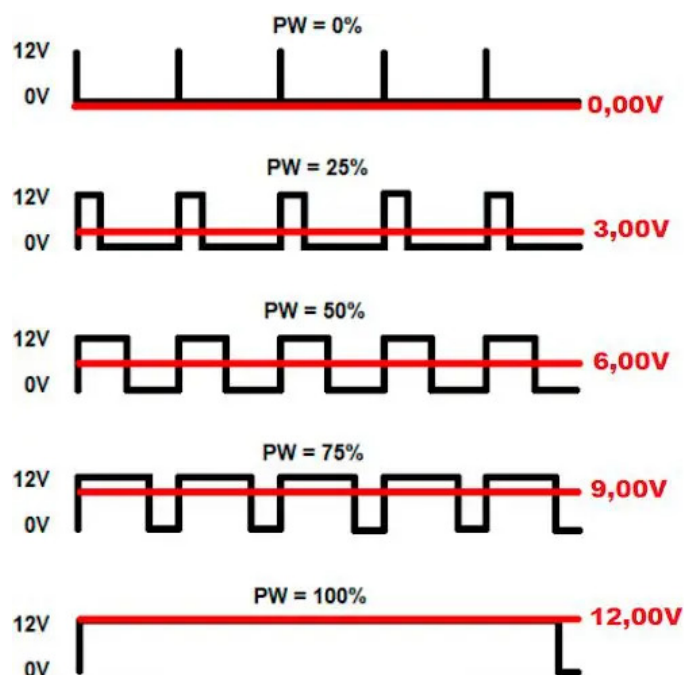
PWM neboli ”Pulse Width Modulation” znamená v češtině ”Pulzně Šířková Modulace”. Jedná se o techniku, která se používá pro ovládání výstupního výkonu elektronických zařízení, jako například pro regulaci jasu LED diod nebo pro řízení rychlosti motorů. [29]

Přenosový signál, který nese informaci o přenášené hodnotě může nabývat hodnot zapnuto/vypnuto tj. logické hodnoty 1 a 0. Hodnota přenášeného signálu je v přenosu ”zakódována” jako poměr mezi stavy zapnuto/vypnuto. Tomuto poměru se říká střída. Cyklus, kdy dojde k přenosu jedné střídy se říká perioda. Omezením pro PWM je to, že přenos informace je vždy omezen na relativní vyjádření a to 0 – 100 %, to znamená, že musí být znám poměr mezi skutečnou hodnotou a procentuálním vyjádřením. Časové hodnoty střídy se pohybují v sekundách, v milisekundách pro přesnější řízení. Perioda je vždy součtem doby zapnuto a vypnuto. [30]

3.11.1 Řízení pomocí PWM

Otáčky hřídele stejnosměrného motoru jsou závislé na hodnotě napájecího napětí. Jedním z nejběžnějších způsobů řízení otáček je řízení pomocí PWM (pulzně šířková modulace). Tato metoda zahrnuje cyklické zapínání a vypínání napájecího napětí ve velmi krátkých časových intervalech. [31]

Na základě střídy a maximální hodnoty napájecího napětí lze vypočítat průměrnou hodnotu napájecího napětí v průběhu alespoň jednoho cyklu PWM, jak je znázorněno na oscilogramech na obrázku: 3.9



■ **Obrázek 3.9** Výpočet průměrného napětí [31]

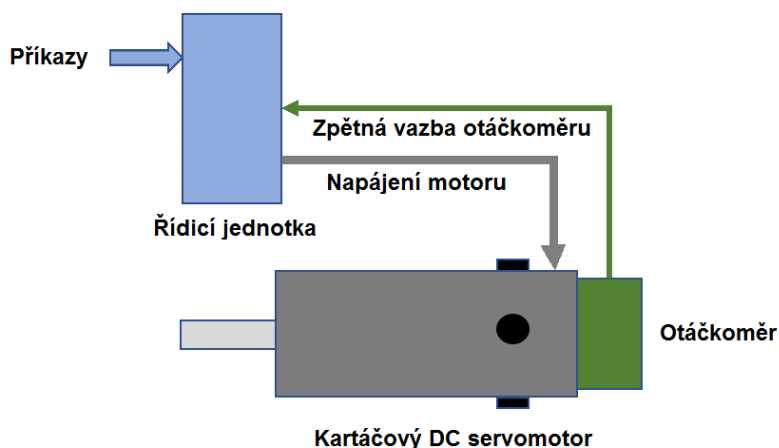
Z výše uvedených oscilogramů je patrné, že čím vyšší je hodnota střídý, tím vyšší je průměrná hodnota napájecího napětí. Praktický důsledek je ten, že čím vyšší je hodnota střídý, tím vyšší budou otáčky motoru a pokles této hodnoty způsobí odpovídající zpomalení hřídele motoru.

4.1 Natáčení kamery

Kameru určenou pro sběr obrazu pro zpracování pomocí umělé inteligence a pro případné navádění je potřeba natáčet s přesností na jednotky stupňů. Toho lze dosáhnout pomocí servomotoru.

4.1.1 Servomotory

Servomotory jsou pohonné jednotky určené k řízení polohy (případně i rychlosti nebo točivého momentu). Na rozdíl od běžných motorů typicky pracují v uzavřené regulační smyčce (*closed-loop*), což znamená, že průběžně porovnávají skutečnou polohu s polohou požadovanou, jak je vidět na obrázku 4.1.



■ Obrázek 4.1 Blokové schéma servo motoru [32]

Servo se skládá z několika klíčových součástí, které dohromady umožňují jeho přesný provoz; typicky obsahuje:

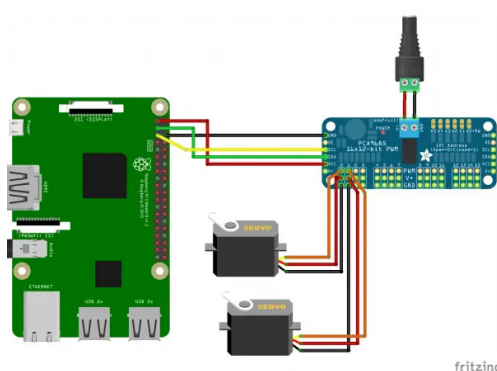
- **Motor** – v závislosti na typu serva může jít o stejnosměrný (DC) nebo bezkartáčový motor; zajišťuje vlastní pohyb.
- **Převodovka** – snižuje otáčky motoru na otáčky vhodné pro výstupní hřídel serva a současně zvyšuje točivý moment.
- **Snímač polohy** – typicky potenciometr nebo enkodér, který měří úhel natočení výstupní hřídele.
- **Řídicí elektronika** – přijímá řídicí signál, vyhodnocuje odchylku mezi požadovanou a skutečnou polohou a podle ní řídí motor.
- **Kryt** – mechanicky chrání komponenty a zajišťuje uchycení. [33]

Absolutně nejrozšířenější servem pro malé projekty, školní robotiku a lehké robotické ruce je servo SG90 Micro Servo (9g). Je velmi levné, malé, ale má plastové převody a nízký krouticí moment.

Z důvodu plastových převodů jsem se nakonec rozhodl pro vylepšené servo MG90S. Jde o odolnější alternativu k SG90 s kovovými převody, která nabízí vyšší spolehlivost. Dále byla zvolena varianta s rozsahem 270 stupňů.

4.1.2 Ovládání servomotorů

Při připojení serva přímo k Raspberry Pi a ovládání pomocí PWM může docházet k záskokům. Tyto projevy mohou souviset s méně stabilním PWM signálem generovaným na GPIO (vliv zátěže systému a časování). Nestabilitu lze omezit použitím externího PWM driveru, konkrétně PCA9685. Tento driver je připojen k Raspberry Pi přes sběrnici I²C a generuje stabilní PWM signál s rozlišením 12 bitů; zároveň umožňuje řídit až 16 servomotorů. Zapojení se dvěma servomotory je možné shlédnout na obrázku 4.2. [34] [35]



■ **Obrázek 4.2** Řízení dvou servomotorů pomocí driveru PCA9685 [35]

4.2 Motory

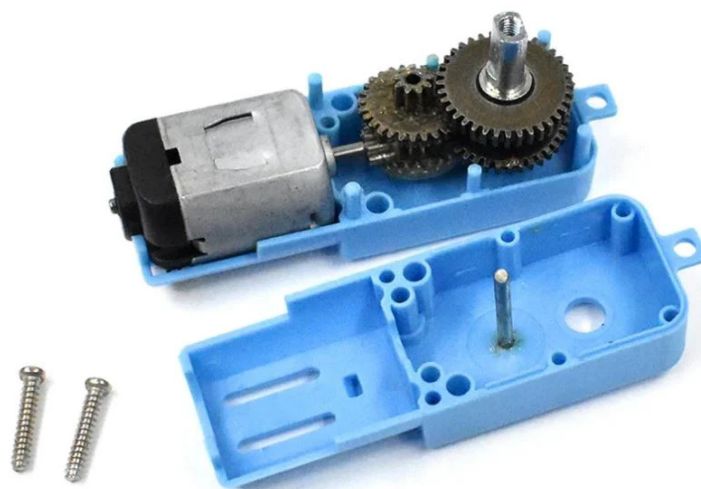
Motory pro pohyb celého podvozku jsem vybíral podle následujících parametrů v tomto pořadí:

1. robustnost
2. cena
3. maximální odběr proudu
4. maximální napětí
5. točivý moment
6. rychlost otáčení

Nejlépe podle těchto parametrů vycházejí TT motory. Tyto motory již obsahují plastové pouzdro s uložením motoru a převodovkou. K tomuto pouzdru lze navíc dohledat plnou dokumentaci s jednotlivými rozměry, tudíž k nim lze jednoduše navrhnout protikus pro uchycení do podvozku. Jejich cena se pohybuje do 100 Kč, což je oproti ostatním motorům v nabídce velmi výhodné. Jedinou nevýhodou je nižší točivý moment a vyšší hlučnost; motor však lze z pouzdra vyjmout a vyměnit za silnější, takže je do budoucna možné systém vylepšit bez nutnosti změny úchytů na podvozku. Na výběr jsou dva typy:

- S plastovou převodovkou: obsahuje převod 1:48, což má za následek vyšší rychlost (až 240 ot/min), ale nižší točivý moment.
- S kovovou převodovkou: obsahuje převod 1:90, což má za následek nižší rychlost (až 130 ot/min), ale vyšší točivý moment. [36]

Nakonec jsem se rozhodl pro motor s kovovou převodovkou kvůli vyšší robustnosti a točivému momentu, ačkoli je přibližně dvakrát dražší než motor s plastovou převodovkou a je hlučnější. Uložení motoru s kovovou převodovkou je zobrazeno na obrázku 4.3.



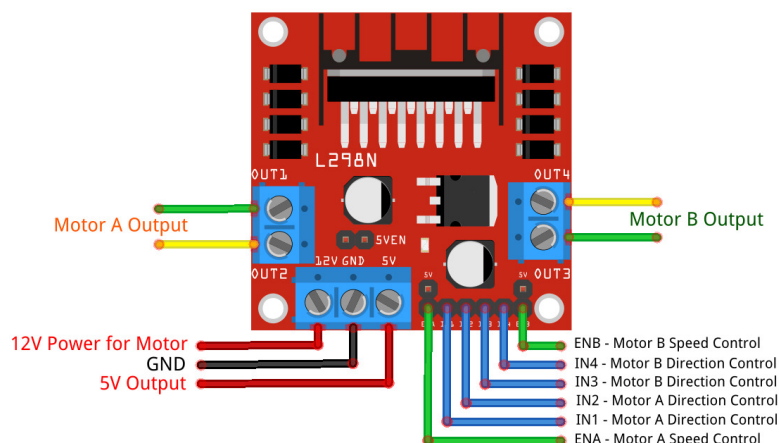
■ **Obrázek 4.3** Motor typu n20 uložený s hřídelí v pouzdře [36]

4.2.1 Ovládání motorů

Vybrané motory mají pouze dva kontakty. Směr otáčení závisí na tom, na který kontakt je přiveden kladný pól napětí; rychlost pak závisí na velikosti aplikovaného napětí. Řídit tyto motory přímo z Raspberry Pi by bylo poměrně náročné, proto jsem se rozhodl použít tzv. H-můstek, konkrétně modul dvojitého H-můstku L298N. Tento modul je cenově přijatelný a umožňuje ovládat dva motory současně. Jeho základní parametry jsou:

- Napětí motorů: 5–35 V
- Maximální proud na jeden motor: 2 A (špičkově 3 A)
- Logická úroveň: 5 V (vhodné pro řízení pomocí Raspberry Pi)
- Maximální výkon: 25 W [37]

Každý motor je ovládán pomocí tří pinů. Směr otáčení se řídí dvěma piny: na jeden je nutné přivést logickou 1 a na druhý logickou 0 (nebo naopak). Na třetí pin lze přivést PWM signál pro nastavení rychlosti. Zapojení je graficky znázorněno na obrázku 4.4.



■ **Obrázek 4.4** Popis pinů na vstupech a výstupech H můstku [38]

4.3 Akcelerátor AI

Výpočetně náročné úlohy konvolučních neuronových sítí by při běhu pouze na Raspberry Pi výrazně zatěžovaly procesor. V důsledku toho by podvozek mohl pomaleji reagovat na řídicí povely a současně by klesla snímková frekvence zpracování obrazu z kamery. Z tohoto důvodu jsem do návrhu zařadil externí AI akcelerátor.

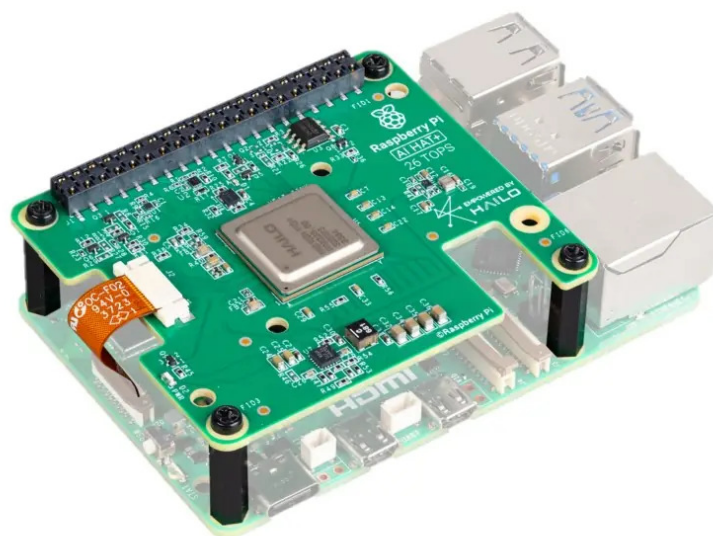
Na trhu je dostupná řada řešení. U Raspberry Pi 4 a starších modelů se nejčastěji používají USB akcelerátory, například Google Coral USB Accelerator nebo Intel Neural Compute Stick 2. Tyto varianty jsou však omezené propustností rozhraní USB a obvykle nabízejí nižší výkon. Raspberry Pi 5 již disponuje sběrnici PCIe, takže umožňuje připojení AI HAT modulu s výrazně vyšším výpočetním výkonem. [39] [40]

Existují zatím dvě dostupné varianty AI HAT pro Raspberry Pi 5 a to s výpočetním výkonem 13 TOPS a 26 TOPS (trilionů operací za sekundu). Ve srovnání s USB akcelerátory s výkonem kolem 4 TOPS jde o výrazný posun. Pro tuto práci jsem zvolil slabší variantu, protože je pro zamýšlené použití dostatečná a současně ekonomicky výhodnější. [41]

Výhodou HAT provedení je také mechanická integrace: desku lze připevnit pomocí distančních sloupků přímo k Raspberry Pi, takže odpadá potřeba samostatného držáku USB akcelerátoru. Praktickým přínosem je kompaktnější sestava, lepší kabelový management a nižší riziko uvolnění konektoru při jízdě. Vybraný akcelerátor již připojený k Raspberry Pi 5 je vyobrazen na obrázku 4.5.

Pro nasazení v mobilní robotice je důležitá nejen hodnota TOPS, ale i latence zpracování, spotřeba a kompatibilita softwarového stacku. Při volbě akcelerátoru je proto vhodné ověřit podporu používaného frameworku a možnost

optimalizace modelu (např. kvantizací), aby byl dosažen stabilní běh v reálném čase. V tomto případě vestavěné aplikace `rpical-apps` v Raspberry Pi OS již nativně podporují AI HAT a automaticky využívají NPU pro kompatibilní post-processingové úlohy. [41]

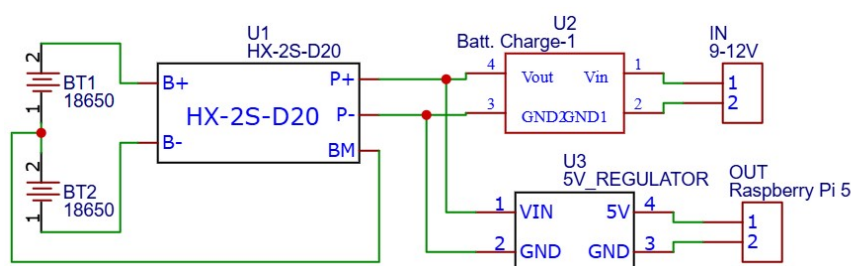


■ **Obrázek 4.5** Akcelerátor umělé inteligence [41]

4.4 Napájení

Aby byl podvozek skutečně mobilní, musí být napájen z baterií, nikoli ze síťového adaptéru. Samotné Raspberry Pi 5 vyžaduje pro stabilní provoz 5 V při proudu až 5 A, což klade poměrně vysoké nároky na napájecí část. [11]

Prvotní návrh vycházel z ukázek zapojení dvojice Li-ion článků 18650 sériově (2S). Jeden článek má jmenovité napětí 3,7 V, dvojice tedy poskytuje vyšší napěťovou úroveň vhodnou pro následnou regulaci. Při sériovém zapojení je potřeba použít ochranný a zároveň balanční obvod HX-2S-D20 a nabíjecí modul TP5100. Po doplnění step-down měniče na 5 V/5 A by tato sestava mohla napájet i Raspberry Pi. V praxi by však bylo nutné doplnit také spolehlivý dohled stavu baterie a mechanismus bezpečného vypnutí systému při vybití. Z tohoto důvodu jsem uvedené 2S napájení jehož schéma je možné schlédnout na obrázku 4.6 ponechal pro servomotory a trakční motory. [42]

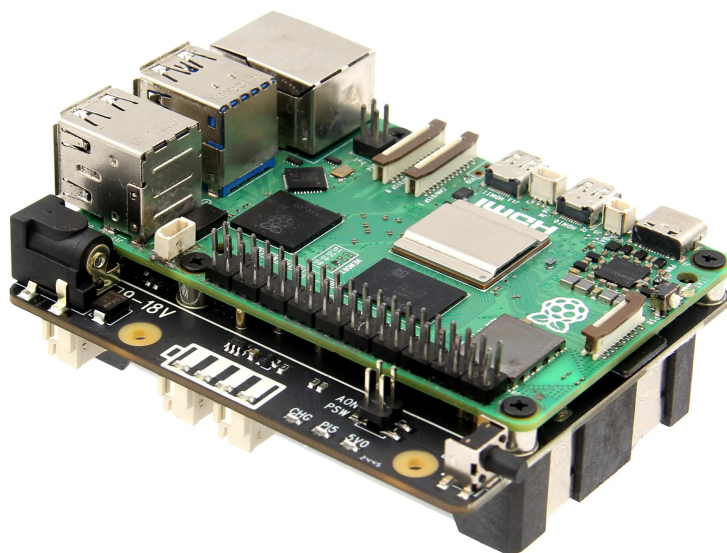


■ **Obrázek 4.6** Schéma zapojení napájení 2S [42]

Pro Raspberry Pi 5 jsem zvolil samostatné řešení v podobě UPS modulu Suptronics X1205, který je navržen přímo pro Raspberry Pi 5. Deska se montuje přímo pod Raspberry Pi, takže není nutné složité externí vedení kabelů. Hlavní vlastnosti modulu jsou:

- Nepřetržité napájení Raspberry Pi (doba provozu závisí na kapacitě baterií)
- Výstup 5,1 V / až 6000 mA s nízkými ztrátami
- Nabíjení baterií až 3000 mA, řízené přes GPIO
- Ochrana proti přepětí, přetížení a obrácené polaritě
- Automatické přepínání mezi síťovým a záložním napájením
- Signalizace stavu pomocí LED a integrovaný *fuel-gauge* přes I²C
- Podporuje zálohování dat při výpadku napájení [43]

Modul, který je zobrazen na obrázku 5.2 využívá dva Li-ion články 21700, které mají oproti 18650 zpravidla vyšší kapacitu. Je však potřeba počítat s tím, že proudový limit 6 A sdílí i další 5V výstupy. Pokud Raspberry Pi odebírá blízko 5 A, zbývá pro další zařízení přibližně 1 A, což je pro serva a pohonné motory nedostatečné. Proto je nutné výkonové větve oddělit: Raspberry Pi napájet přes UPS a motory se servy z vlastního zdroje. [43]



■ **Obrázek 4.7** UPS modul x1205 již přišroubovaný k Raspberry Pi 5 [43]

4.5 UWB Moduly

Technologie UWB není v hobby segmentu tak rozšířená jako například Wi-Fi nebo Bluetooth, proto je nabídka snadno dostupných modulů omezenější. Při výběru pro tento projekt jsem porovnával zejména cenu, dostupnost dokumentace, způsob integrace a dosažitelnou přesnost měření vzdálenosti.

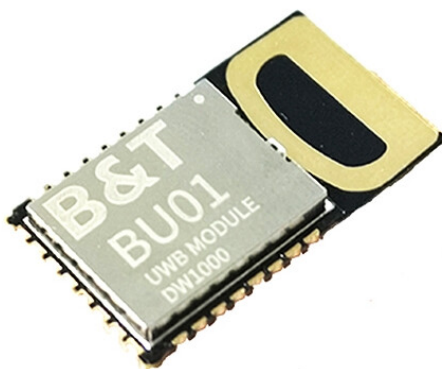
Jako první varianta se nabízel modul EWM550. Je cenově dostupný, ale má slabší ekosystém dokumentace a typicky menší flexibilitu při hlubší integraci do vlastního firmware. Komunikace bývá řešena přes sériové rozhraní, což sice usnadňuje základní zprovoznění, ale zároveň omezuje nízkouúrovňovou kontrolu nad měřením. [44]

Dále jsem uvažoval nad čipy DW1000 a DW3110 (původně Decawave, dnes Qorvo). Obě řešení používají SPI a jsou dostupná ve formě samostatného SMD čipu pro vlastní návrh desky. Varianta DW3110 je novější a obecně nabízí lepší parametry pro přesné určování polohy, ale za vyšší cenu. DW1000 je naopak dlouhodobě prověřené a cenově výhodnější řešení s velmi dobrou dostupností komunitních ukázek a knihoven. [45] [46]

Jako další možnost, která se v praxi často používá, lze zmínit integrované moduly DWM1001 nebo novější DWM3001. Tyto moduly kombinují UWB část

s další elektronikou (u některých variant i s mikrokontrolérem), takže urychlují prototypování a zjednodušují RF návrh. Nevýhodou bývá vyšší cena a menší svoboda při hardwarových úpravách oproti čistému SMD čipu. [47] [48]

Pro tento projekt jsem nakonec zvolil modul BU01 od společnosti Ai-Thinker. Jedná se o vysoce integrovaný UWB modul založený právě na čipu DW1000, který je zobrazen na obrázku 4.8. Hlavním důvodem výběru byl poměr cena/výkon, dostupnost technických podkladů a dostatečná přesnost pro zamýšlený způsob navigace. Volba zároveň ponechává možnost budoucího přechodu na novější generaci (např. DW3000), pokud by se v dalších iteracích ukázala potřeba vyšší přesnosti nebo robustnějších funkcí. [49]



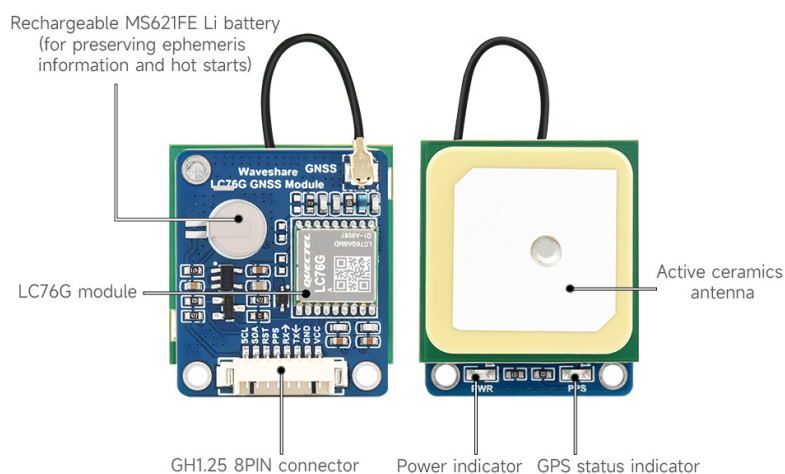
■ **Obrázek 4.8** Modul BU01 v SMD provedení [49]

4.6 GPS

V současnosti je dostupné velké množství GPS/GNSS modulů vhodných pro Raspberry Pi. Při výběru jsem zohledňoval zejména cenu, podporu více satelitních systémů, způsob komunikace a dostupnost dokumentace. Na trhu jsou dostupné jak základní moduly (např. L76K/L76G), tak i pokročilé průmyslové varianty (např. u-blox ZED-F9P), které při použití RTK korekcí umožňují přesnost na úrovni centimetrů. [50]

Pro tento projekt jsem zvolil modul L76G, který představuje rozumný kompromis mezi cenou, spotřebou a parametry. Oproti jednodušším variantám nabízí lepší použitelnost v praxi a podporuje komunikaci jak přes sériové rozhraní (UART), tak přes I²C. Hlavní vlastnosti modulu jsou:

- Podpora více GNSS systémů (Multi-GNSS)
- Vestavěný nízkošumový zesilovač (LNA) pro zlepšení citlivosti příjmu
- Vestavěná dobíjecí baterie MS621FE pro uchování efemerid a rychlejší start
- Typická přesnost polohy přibližně 1,5 m CEP [50]



■ **Obrázek 4.9** GPS modul L75G [50]

Při implementaci modulu LC76G je nezbytné zohlednit limity technologie GNSS, zejména kolísání přesnosti vlivem městské zástavby, vegetace či jiných překážek způsobujících útlum signálu. Pro dosažení spolehlivé navigace je proto nutné surová data filtrovat (např. pomocí Kalmanova filtru či klouzavého průměru). Kritickým faktorem je absence integrovaného magnetometru v modulu LC76G. Pro autonomní navádění totiž znalost pouhých souřadnic (x, y) nepostačuje; systém musí znát také aktuální orientaci (kurz) stroje vzhledem ke světovým stranám. K určení azimutu byl proto zvolen externí modul GY-511, který integruje magnetometr a tříosý akcelerometr. Tento snímač slouží jako digitální kompas umožňující výpočet vektoru k cíli na základě detekce magnetického severu. Modul je do systému integrován prostřednictvím sběrnice I2C, což zajišťuje efektivní komunikaci s nadřazeným mikrokontrolérem. [51]

Pokud by bylo v budoucnu požadováno přesnější navádění na konkrétní bod, lze systém rozšířit o RTK řešení s dvoufrekvenčním GNSS modulem (např. ZED-F9P). To však zvyšuje cenu i složitost celé sestavy.

4.7 Podvozek

Mým prvotním záměrem bylo použít hotový podvozek a tím urychlit vývoj. Na trhu je však široké spektrum variant pro TT motory, které se výrazně liší cenou, rozměry i mechanickou odolností. Při analýze dostupných řešení se ukázalo, že levnější podvozky (řádově stovky korun) často nevyhovují požadované robustnosti, zatímco výrazně dražší platformy by byly v rozporu s cílem navrhnout cenově dostupné řešení.

Pro rozhodnutí byly klíčové zejména tyto požadavky:

- dostatečná mechanická pevnost vzhledem k hmotnosti baterií, řídicí elektroniky a senzorů,
- vhodné rozměry pro umístění všech modulů bez kolizí a s možností servisního přístupu,
- kompatibilita s vybranými motory a jednoduchá montáž,
- nízká cena při zachování provozní spolehlivosti.

Vzhledem k těmto požadavkům jsem se rozhodl navrhnout vlastní podvozek. Vlastní konstrukce umožňuje optimalizovat rozmístění komponent, snížit riziko mechanických kompromisů a současně zachovat cílový poměr cena/výkon. Tento přístup je v souladu s analytickou částí práce, která upřednostňuje prakticky realizovatelné a ekonomicky dostupné řešení bez zbytečného nárůstu složitosti.

Kapitola 5

Návrh řešení

Na základě analytické části jsem nejprve navrhl základní schéma zapojení všech aktuátorů podvozku. Cílem bylo ověřit propojení hlavních výkonových a řídicích prvků, tedy Raspberry Pi, PWM driveru, servomotorů a pohonných motorů. Toto schéma sloužilo jako výchozí podklad pro další návrh, protože umožnilo včas identifikovat potřebná rozhraní, napájecí vazby i způsob oddělení řídicí a výkonové části systému. Zároveň se tímto návrhem upřesnilo, které části bude vhodné dále rozvíjet samostatně a které bude nutné řešit jako vzájemně provázané subsystémy.

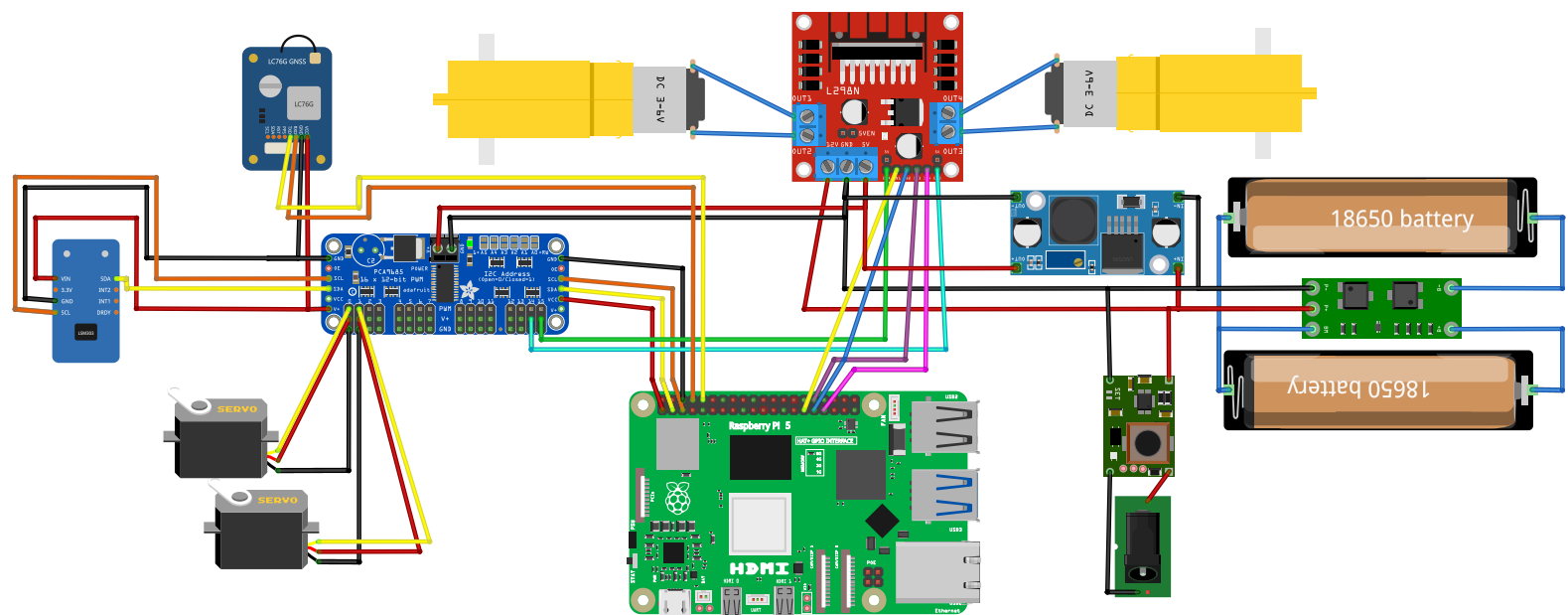
Schéma je znázorněno na obrázku 5.1. Zachycuje zejména vztah mezi hlavní řídicí jednotkou, budiči a akčními členy podvozku. Nejde o finální výrobní dokumentaci, ale o koncepční návrh, který sloužil jako podklad pro následné rozdělení systému do dalších částí.

Zbývající části systému jsem rozdělil do čtyř hlavních celků, které bylo možné vyvíjet částečně nezávisle:

- konstrukční návrh a výroba podvozku,
- návrh desek plošných spojů s moduly BU01 pro navádění pomocí UWB a vývoj programu pro jejich obsluhu,
- návrh webové aplikace pro vzdálené ovládání podvozku,
- vývoj řídicí aplikace na Raspberry Pi pro integraci všech subsystémů.

Toto členění umožnilo postupovat iterativně, průběžně testovat jednotlivé části a následně je spojit do jednoho funkčního celku. Současně se tím zjednodušilo ladění chyb, protože bylo možné jasně oddělit mechanické, hardwarové a softwarové problémy.

Na následujících stranách jsou jednotlivé části popsány podrobněji, včetně klíčových konstrukčních rozhodnutí a důvodů, které vedly k finálnímu řešení.



fritzing

■ Obrázek 5.1 Základní schéma zapojení

5.1 Řešení podvozku

Po delší analýze jsem se rozhodl navrhnout podvozek s diferenciálním řízením, tedy s jedním motorem na každé straně. Oproti řízení pomocí natáčení kol je toto řešení konstrukčně jednodušší a zároveň robustnější. Není zde potřeba přesné řízení natočení kol servem, jehož nepřesnost by mohla u klasického uspořádání způsobovat nechtěné vychylování směru jízdy. Další výhodou je vysoká manévrovatelnost, zejména možnost otáčení na místě pomocí protiběžného otáčení levé a pravé strany.

Zvolil jsem konfiguraci se dvěma koly na každé straně namísto pásového podvozku, především kvůli nižšímu tření při zatáčení, což je důležité s ohledem na omezený výkon použitých motorů. Současně jsem implementoval pohon 4×4, který zlepšuje průchodnost terénem a usnadňuje překonávání menších překážek (například prahů). Při návrhu jsem navíc zohlednil: polohu těžiště, rozložení hmotnosti mezi nápravy, servisní přístup ke komponentám a odolnost konstrukce vůči vibracím od motorů.

5.1.1 Prvotní návrh

V první fázi jsem uvažoval o konstrukci podvozku z ocelového plechu, především kvůli jeho vysoké pevnosti. Cílem bylo vytvořit co nejjednodušší jednolitý celek s minimem spojů.

Postup výroby spočíval ve vyvrtání montážních otvorů a následném ohýbání plechu. Tento přístup však přinesl několik praktických nevýhod: náročnější výrobu (nutnost přesného vrtání a ohýbání), vyšší požadavky na dílenské vybavení a menší dostupnost pro běžné domácí nebo školní podmínky. Pro kvalitní opakovatelnou výrobu je navíc vhodný přístup k CNC fréze a ohýbačce, což není vždy k dispozici.

V průběhu návrhu se ukázaly i další aspekty, které jsem zpočátku podcenil. Jednalo se zejména o výrobní tolerance po ohybu (a tím i riziko nepřesného usazení motorů), složitější servisní zásahy do jednolité konstrukce, přenos vibrací z pohonu do elektroniky a nutnost dodatečné povrchové úpravy proti korozi. Nevýhodou byla také vyšší hmotnost ocelového řešení, která by zhoršila akceleraci, zvýšila odběr proudu a zkrátila dobu provozu na baterii.

V kontextu požadavků projektu se tedy ukázalo, že ocelová varianta by negativně ovlivnila dynamiku vozidla, energetickou efektivitu i reprodukovatelnost návrhu. Z těchto důvodů jsem původní variantu z ocelového plechu opustil.

5.1.2 Návrh pomoci 3D tisku

Jako vhodnější alternativu jsem zvolil návrh podvozku pro 3D tisk. Hlavní výhodou je dostupnost technologie, nízké výrobní náklady a rychlá možnost iterací při úpravách modelu. Nevýhodou je nižší tuhost běžných tiskových materiálů oproti kovu a omezená velikost tiskové plochy, což vyžaduje rozdělení konstrukce na více dílů.

Z hlediska volby materiálu připadají v úvahu zejména PLA, PETG a ABS. Pro tento projekt jsem zvolil PLA, protože se z něho tiskne nejsnáze, má malé nároky na nastavení tiskárny a pro dané mechanické zatížení podvozku poskytuje dostatečnou pevnost. Praktickou výhodou je také to, že při běžném tisku neprodukuje tak výrazné výpary jako například ABS, což je vhodné při práci v domácím nebo školním prostředí.

Pro návrh jednotlivých částí jsem použil CAD software Fusion 360, který umožňuje přesné parametrické modelování a rychlé úpravy podle aktuálních požadavků. Tím bylo možné průběžně optimalizovat geometrii dílů, umístění montážních otvorů i návaznost na ostatní komponenty (motory, elektroniku a napájení).

Výsledný podvozek jsem rozdělil do pěti hlavních konstrukčních částí:

- stranová deska,
- držák strany,
- spodní deska,
- držák pro kameru,
- držák pro GPS a magnetometr

5.1.3 Stranová deska

Stranová deska tvoří boční nosný prvek podvozku. V této fázi návrhu jsem se rozhodl, že na každé straně bude pouze jeden motor. Toto řešení snižuje výrobní náklady i spotřebu energie, současně však vyžaduje přenos výkonu na obě kola dané strany. Uvažoval jsem dvě základní varianty přenosu, které jsou zobrazeny na obrázku 5.2:

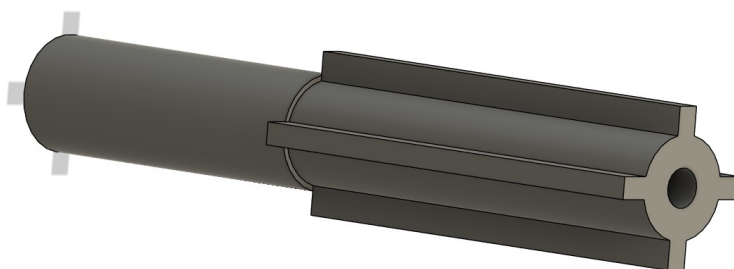
- **Přenos pomocí ozubených kol** – konstrukčně jednoduché řešení, které vyžaduje přesné rozteče os a vhodně zvolená kola. Pro dlouhodobý provoz by bylo vhodné použít odolnější kovová kola, což však zvyšuje cenu i hmotnost sestavy. Nevýhodou je také vyšší tření a citlivost na nepřesnosti v osazení.
- **Přenos pomocí řemenu** – návrhově složitější varianta kvůli potřebě spolehlivého napnutí řemenu, ale s nižší hlučností, menšími ztrátami a lepšími tolerancí drobných nepřesností.



■ **Obrázek 5.2** Přenos pomocí ozubených kol v horní části a pomocí řemenu v dolní části

Po návrhu a praktickém ověření obou variant jsem zvolil řemenový převod s ozubeným řemenem GT2, který je běžně dostupný díky rozšíření v 3D tiskárnách. Vhodnější je použít uzavřený řemen, protože u otevřené varianty je v této velikosti komplikované zajistit spolehlivé spojení konců bez rizika prokluzu nebo seskočení. Výhodou je také to, že je tento řemen dostupný v mnoha rozměrech, takže je možné podvozek v budoucnu například prodloužit.

Pro řemen GT2 jsou dostupné kladky pro různé průměry hřídele. V mém případě jsem zvolil variantu pro osu 8 mm s ohledem na geometrii použitého TT motoru. Přední kolo je uchyceno přímo na výstupní hřídel motoru, zadní kolo je uloženo v ložisku. Pro přední část byla navržena osa s průchozím otvorem pro závitovou tyč, aby bylo možné přesné a pevné mechanické uchycení do hřídele motoru. Na obou hřídelích jsou umístěny kladky, které jsou zajištěny stavěcími šrouby. Zadní osa prochází ložiskem a je v něm rovněž zajištěna stavěcím šroubem. Hřídel jsem také opatřil výběžky po stranách pro zamezení prokluzu v náboji kola. Návrh hřídele je zobrazen na obrázku 5.3.



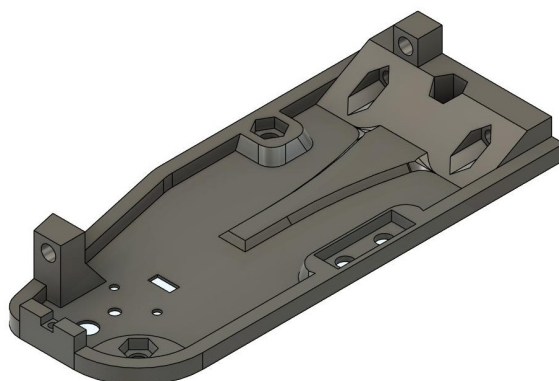
■ **Obrázek 5.3** Finální návrh hřídele

Původně jsem napínání řemenu řešil samostatnou napínací kladkou. Ta byla uložena ve vytištěném držáku se samostatnou osou. Tento držák bylo následně možné posouvat od středové osy pomocí připevněného šroubu. V praxi se však ukázalo, že toto řešení má nízkou tuhost: osa kladky se ve vytištěném držáku natáčela a mechanismus zabíral příliš mnoho prostoru.

Ve finální variantě je ložisko zadního kola uloženo v samostatném posuvném dílu, který je veden dvěma závitovými tyčemi upevněnými do stranové desky. Napnutí řemenu se nastavuje symetrickým posunem tohoto dílu pomocí dvojic matic. Oproti původnímu konceptu je řešení kompaktnější, tužší a lépe servisovatelné. Výhodou je také možnost jednoduše upravit rozvor podvozku použitím delšího řemenu.

Při návrhu jsem dále zohlednil: souosost motoru a zadní osy (kvůli rovnoměrnému opotřebení řemenu), dostatečnou rezervu kolem rotujících částí, přístup k maticím pro opakované seřízení a lokální vyztužení v okolí montážních otvorů. Důležitá je také orientace dílu při tisku a počet perimetrů, protože právě v okolí šroubových spojů a ložisek vzniká nejvyšší mechanické namáhání. Po otestování prvotního návrhu jsem dále navrhl uchycení pro ložisko 608ZZ, které je přišroubováno k motoru z druhé strany desky a pomáhá vést hřídel spojenou s motorem. Toto řešení snižuje vůli hřídele ve spoji s motorem.

Stranová deska tedy neplní pouze funkci nosného prvku pro uchycení motoru, ale současně určuje geometrii celé boční části podvozku. Pro zvýšení tuhosti konstrukce obsahuje také krajní body pro vložení závitových tyčí, které omezují nežádoucí deformace při jízdě v terénu. Výsledný návrh stranové desky je zobrazen na obrázku 5.4.



■ **Obrázek 5.4** Finální návrh stranové desky

S ohledem na požadovanou pevnost v oblasti uchycení motorů bylo nutné umístit motory do střední části boční desky, kde je konstrukce nejméně náchylná k lokálním deformacím. Současně bylo potřeba navrhnout uchycení držáku strany v dostatečné vzdálenosti od motoru a posunout jej níže, aby se podařilo snížit těžiště celé sestavy. Tato kombinace konstrukčních požadavků vedla k potřebě kol o průměru větším než 9 cm, která zajistí dostatečnou světlou výšku i prostorovou návaznost na ostatní části podvozku.

Přímý 3D tisk takto velkých kol by byl časově náročný a z hlediska reprodukovatelnosti i ekonomiky výroby nevýhodný. Zároveň jsem se chtěl vyhnout nutnosti kupovat hotová kola, protože by tím vzrostla cena řešení a zmenšila by se možnost jejich snadné úpravy. Proto jsem zvolil kompromisní variantu v podobě tištěného disku, na který je po vytištění upevněna gumová obruč tvořící výsledný běhoun. Toto řešení zachovává nízké výrobní náklady, umožňuje snadnou opakovatelnost a současně poskytuje možnost měnit vlastnosti kola podle typu povrchu. Praktickou výhodou je také možnost experimentovat s šířkou, tvrdostí nebo profilem gumové vrstvy bez nutnosti přepracovat celý disk. Výsledný návrh disku kola je zobrazen na obrázku 5.5.



■ **Obrázek 5.5** Finální návrh disku kola

5.1.4 Držák strany

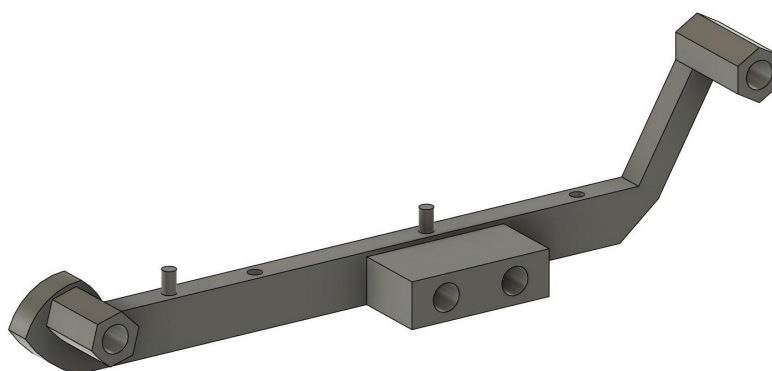
Stranové desky jsem se rozhodl propojit pomocí závitových tyčí. Tento způsob je cenově dostupný, konstrukčně jednoduchý a při vhodně zvoleném průměru poskytuje dostatečnou tuhost i pro zatížení vznikající při jízdě v terénu. Pro návrh jsem zvolil závitové tyče M6, které představují rozumný kompromis mezi hmotností, pevností a dostupností spojovacího materiálu.

V první variantě jsem uvažoval o přímém uchycení spodní desky na závitové tyče. Toto řešení se však ukázalo jako nevhodné z hlediska výroby pomocí 3D tisku, protože by vyžadovalo velké množství podpor a vedlo by ke zhoršení kvality funkčních ploch.

Proto jsem navrhl samostatný držák strany, který se připevňuje ke stranové desce a současně vytváří montážní body pro spodní desku. Držák konstrukčně propojuje oblast otvorů pro závitové tyče; po zasunutí do stranové desky se chová jako její zpevňující část. Samotné závitové tyče následně procházejí skrz celý uzel, čímž zvyšují pevnost a zároveň usnadňují přesné ustavení sestavy při montáži.

Důležitým požadavkem byla jednoduchá tisknutelnost bez složitých podpor a možnost opakované demontáže. Při návrhu jsem proto zohlednil i dostatečné vůle pro zasunutí dílu po vytištění, vyztužení v okolí otvorů proti praskání, přístup k maticím během montáže a minimalizaci koncentrace napětí v ostrých rozích.

Držák strany, zobrazený na obrázku 5.6, tak plní tři funkce současně: zajišťuje přesnou polohu dílů, zvyšuje celkovou tuhost podvozku a zjednodušuje servisní zásahy při výměně nebo úpravě komponent.



■ Obrázek 5.6 Finální návrh držáku strany

5.1.5 Spodní deska

Spodní deska tvoří hlavní montážní platformu pro elektroniku a napájecí prvky. Při návrhu bylo klíčové rozmístění otvorů a upevňovacích bodů tak, aby bylo možné bezpečně umístit Raspberry Pi, napájecí moduly i další periferní desky, a současně zachovat prostor pro vedení kabeláže a přístup k servisním bodům.

V průběhu návrhu se ukázalo, že plocha potřebná pro osazení všech komponent je již příliš velká pro efektivní tisk v jednom kuse. Tisk plného dílu by byl časově náročný, materiálově neekonomický a zvyšoval by riziko deformací při chladnutí. Proto jsem jako základ zvolil překližkovou desku, která je v této části konstrukce dostatečně pevná, lehká a snadno opracovatelná běžným nářadím.

Na překližkový základ jsem navrhl a vytiskl pouze funkční doplňky: vztužovací prvky, montážní držáky a distanční sloupky pro jednotlivé moduly. Tyto díly jsou k desce připevněny šroubovými spoji, což usnadňuje výměnu komponent i budoucí úpravy rozložení. Zároveň tím odpadla potřeba tisknout velké díly se složitými podporami.

Při realizaci jsem zohlednil i praktické detaily, jako jsou rezerva pro poloměr ohybu kabelů, oddělení silové a signálové kabeláže kvůli omezení rušení, dostatečné větrání kolem Raspberry Pi a měničů, ochrana spodní strany desky proti oděru a vibracím a použití standardizované rozteče montážních otvorů pro snadnější opakovatelnost výroby.

5.1.6 Držák pro kameru

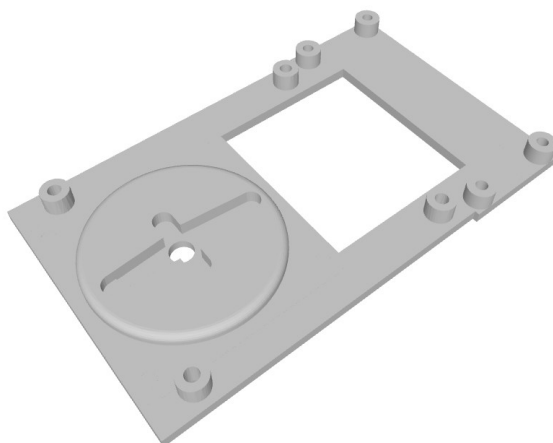
Držák pro kameru je důležitou součástí systému pro navádění pomocí strojového vidění. Jeho hlavním úkolem je umožnit natáčení kamery při skenování okolí. S ohledem na jednoduchou tisknutelnost jednotlivých komponent jsem držák rozdělil na čtyři části:

- základnu pro upevnění k podvozku,
- držák servomotoru pro zajištění horizontálního pohybu,
- držák servomotoru pro zajištění vertikálního pohybu,
- uchycení samotné kamery.

Základna

Pro dostatečný výhled musí být kamera umístěna co nejvýše, aby jí ostatní komponenty nebránily ve výhledu. Základnu jsem proto koncipoval jako samostatný nástavec, který lze připevnit na Raspberry Pi. K upevnění slouží čtveřice otvorů, pomocí nichž jsou k základně přišroubovány distanční sloupky. V mém případě jsou tyto sloupky následně zašroubovány do dalších distančních sloupků, které nesou akcelerační neuronové síť na Raspberry Pi. Součástí

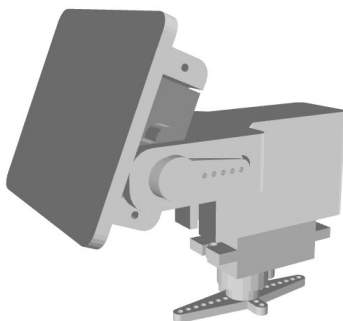
základny je také část pro přišroubování křížové páky serva a modulu PCA9685 pro jejich obsluhu. Připevněné servo se tedy nachází v poloze vzhůru nohama. Vzhledem k umístění nad akcelerátorem jsem se při návrhu snažil zachovat dostatečný prostor pro odvod tepla. Navržená základna je zobrazena na obrázku 5.7.



Obrázek 5.7 Základna pro držák na kameru

Uchycení kamery

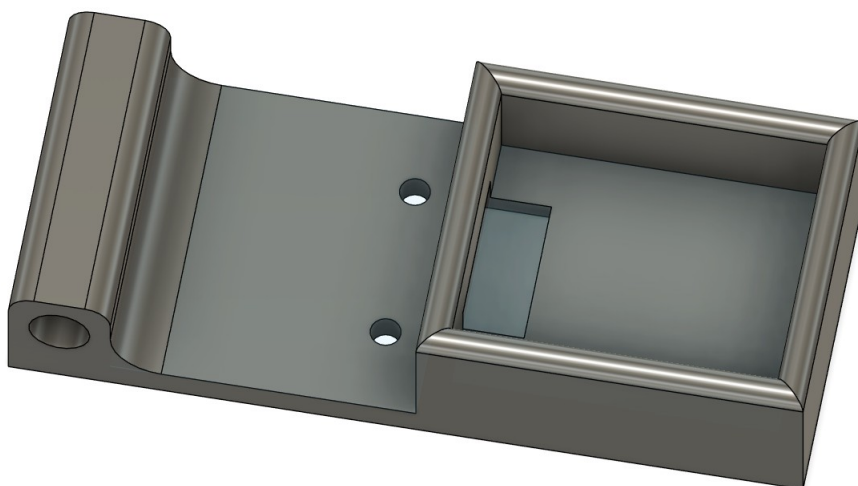
Pro uchycení kamery k základně bylo nejprve nutné navrhnout držáky serv. Držák serva přichyceného k základně jsem koncipoval jako násuvný modul se dvěma rameny pro uchycení druhého serva. V jednom rameni je vytvořen prostor pro přišroubování jednoramenné páky serva, druhé rameno slouží k vystředění serva tak, aby nebylo nakloněné. Samotný návrh byl poměrně rychlý díky dostatečné dokumentaci vybraných serv. Na servo zajišťující pohyb ve vertikální ose je následně pomocí šroubů připevněna deska pro uchycení držáku kamery. Samotný držák kamery rovněž obsahuje desku, která je k této části přišroubována. Dělení tohoto úchytu na dva díly zjednodušuje tisk bez podpor. Zároveň tento způsob umožňuje navrhovat různé úchyty pro různé kamery a následně je jednoduše vyměňovat. Na obrázku 5.8 jsou zobrazena serva uchycená v držácích a k nim připevněná deska pro uchycení držáku kamery.



■ **Obrázek 5.8** Konstrukce uchycení serv

5.1.7 Držák pro GPS a magnetometr

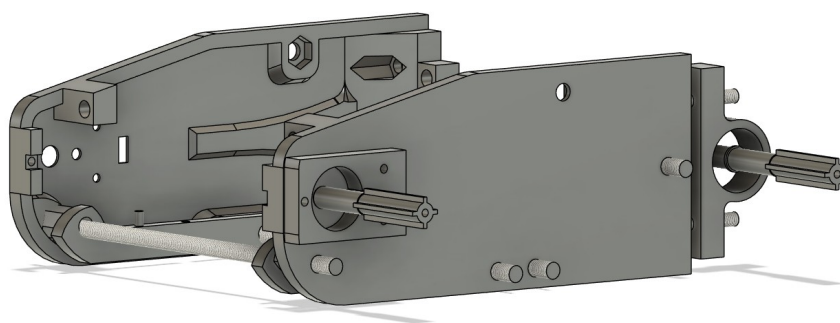
Pro zajištění co nejvyšší přesnosti měření GPS modulu a magnetometru bylo nutné oba prvky umístit v dostatečné vzdálenosti od zdrojů elektromagnetického rušení. Negativní vliv na kvalitu měření nemají pouze samotné motory, ale také vodiče s vyšším proudovým zatížením a výkonová elektronika, které ve svém okolí vytvářejí proměnné elektromagnetické pole. Z tohoto důvodu byl navržen samostatný držák pro GPS modul a magnetometr, který je upevněn na závitové tyči spojující obě strany podvozku v jeho zadní části. Toto konstrukční řešení zajišťuje větší odstup od rušivých prvků, umístění senzorů ve vyšší poloze nad ostatními komponentami a zároveň umožňuje přehledné vedení kabeláže. Přínosem je rovněž snadná montáž, demontáž a možnost dodatečné úpravy polohy držáku při finálním ladění měřicí soustavy. Na obrázku 5.9 je zobrazen finální návrh držáku.



■ **Obrázek 5.9** Výsledný návrh držáku GPS

5.1.8 Shrnutí návrhu podvozku

Zvolené rozdělení na nosnou překližkovou část a tištěné funkční prvky umožnilo pružně reagovat na změny během vývoje, zrychlilo prototypování a současně zachovalo požadovaný poměr cena/výkon. Tento postup je v souladu s cílem práce: navrhnout funkční, cenově dostupný a snadno reprodukovatelný systém. Výsledný návrh podvozku je zobrazen na obrázku 5.10.



■ Obrázek 5.10 Výsledný návrh podvozku

5.2 Řešení UWB modulů

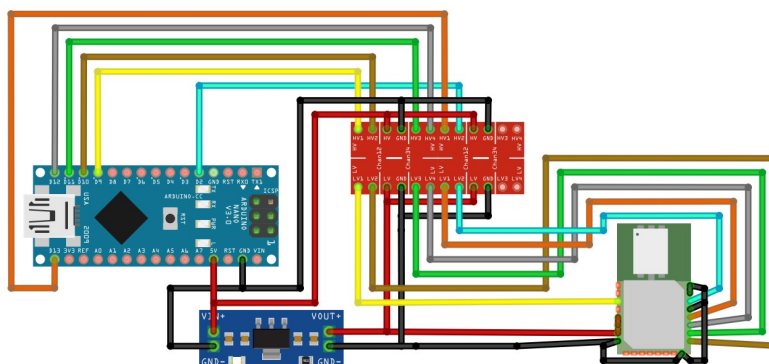
V počáteční fázi vývoje bylo mým záměrem obsluhovat UWB moduly přímo z Raspberry Pi 5. Po analýze architektury systému se však ukázalo, že by toto řešení nebylo vhodné. Modul BU01 neobsahuje samostatnou řídicí logiku pro kompletní zpracování měření vzdálenosti; je tedy nutné zajišťovat konfiguraci, časově citlivou komunikaci i výpočetní kroky v nadřazeném zařízení. Přenesení těchto úloh přímo na Raspberry Pi by vedlo ke zbytečnému zatížení hlavního řídicího systému a mohlo by negativně ovlivnit paralelně běžící části aplikace (řízení pohonu, zpracování obrazu, webové rozhraní).

Limitem je také charakter časování v běžném linuxovém prostředí. Ačkoliv Raspberry Pi nabízí vysoký výpočetní výkon, operační systém negarantuje tak deterministické časování jako mikrokontrolér. U UWB měření je přitom stabilita časování zásadní, protože i malé odchylky mohou zvyšovat chybu výsledné vzdálenosti. Z praktického hlediska hraje roli také softwarová podpora: dostupné knihovny pro čip DW1000 jsou lépe rozpracované pro mikrokontroléry (zejména Arduino) než pro přímé nasazení v Pythonu na Raspberry Pi.

Z těchto důvodů jsem zvolil oddělenou architekturu, ve které UWB moduly obsluhuje externí mikrokontrolér Arduino Nano a Raspberry Pi zajišťuje nadřazené rozhodování. Toto rozdělení přináší dvě hlavní výhody: časově kritická část měření běží na dedikovaném zařízení a hlavní počítač není zatěžován nízkourovňovou komunikací. Arduino Nano bylo zvoleno také s ohledem na cenu, dostupnost a rozsáhlou komunitní dokumentaci.

Vybraný modul BU01 využívá rozhraní SPI a doplňkové řídicí signály. Prakticky je potřeba připojit čtyři vodiče SPI (MOSI, MISO, SCK, CS), dále signál přerušení (IRQ) a reset (RST). Klíčovým omezením je napěťová úroveň: BU01 pracuje na 3,3 V, zatímco běžné Arduino Nano používá logiku 5 V. Přímé propojení by mohlo modul poškodit, proto je nutné použít převodník logických úrovní a stabilní napájení 3,3 V.

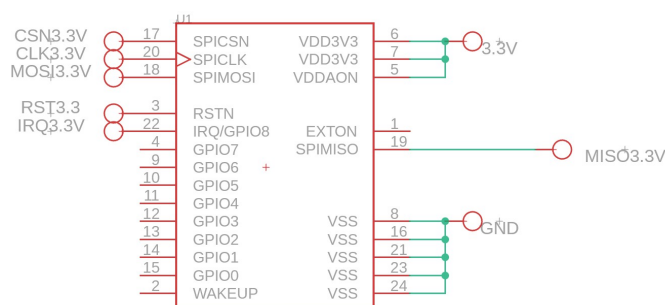
Při návrhu prototypu jsem použil osmikanálový obousměrný převodník založený na tranzistorech BSS138 (využito bylo šest kanálů) a samostatný modul lineárního regulátoru AMS1117 pro převod napětí z 5 V na 3,3 V. Protože modul BU01 je v SMD provedení, nebylo možné celý prototyp pohodlně sestavit na nepájivém poli. Většina spojů musela být pájena ručně. Na obrázku 5.11 je zobrazeno kompletní zapojení. Funkčnost se mi podařilo ověřit s využitím ukázkového programu a systém byl schopen měřit vzdálenost, nicméně opakovatelnost výsledků nebyla dostatečná pro přesné navádění. Typická odchylka se pohybovala přibližně v řádu vyšších desítek centimetrů (okolo 70 cm podle podmínek měření).



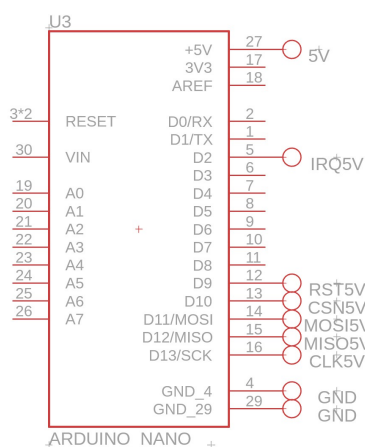
Obrázek 5.11 Prvotní zapojení modulu BU01

Při diagnostice se ukázalo, že významným zdrojem chyb je kvalita propojení prototypu: dlouhé vodiče, více přechodových odporů a citlivost na elektromagnetické rušení. Pro zvýšení stability měření jsem proto navrhl vlastní desku plošných spojů. Cílem návrhu bylo zkrátit kritické signálové cesty, sjednotit zemní referenci, omezit parazitní jevy a zlepšit mechanickou spolehlivost celé sestavy.

Desku jsem navrhoval v programu Fusion 360, který umožňuje propojit schéma, návrh PCB a následnou 3D kontrolu osazení v jednom prostředí. To mi usnadnilo ověření prostorových návazností na ostatní části robota i plánování servisního přístupu. Schéma zapojení modulu BU01 a mikrokontroléru Arduino Nano jsou zobrazeny na obrázcích 5.12 a 5.13.

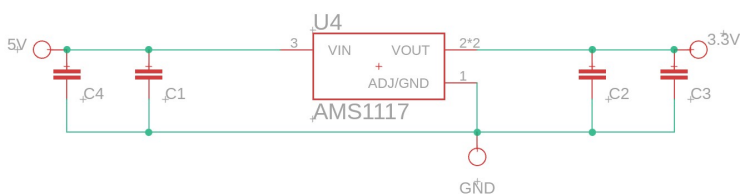


■ **Obrázek 5.12** Schéma zapojení modulu BU01



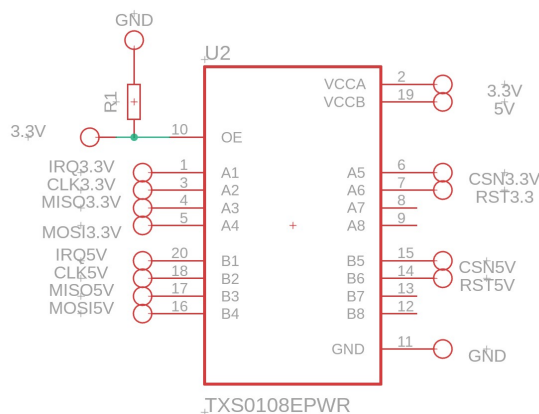
■ **Obrázek 5.13** Schéma zapojení mikrokontroléru Arduino Nano

Pro eliminaci dlouhých vodičů a přechodových spojů jsem se rozhodl nevyužít modul lineárního regulátoru, ale navrhnout regulátor přímo na desce. Rozhodl jsem se použít stejný čip ASM1117, a proto jsem při návrhu vycházel ze schématu zobrazeného na obrázku 5.14, které je dostupné přímo v datasheetu tohoto čipu.



■ **Obrázek 5.14** Schéma zapojení lineárního regulátoru ASM1117

Oproti původnímu prototypu jsem také upravit koncepci převodu logických úrovní: místo diskretního řešení s více tranzistory jsem použil integrovaný převodník TXS0108E. Tím se snížil počet součástek, zjednodušilo routování a zlepšila reprodukovatelnost při osazování dalších kusů. Na obrázku 5.15 níže je vyobrazeno schéma zapojení převodníku.

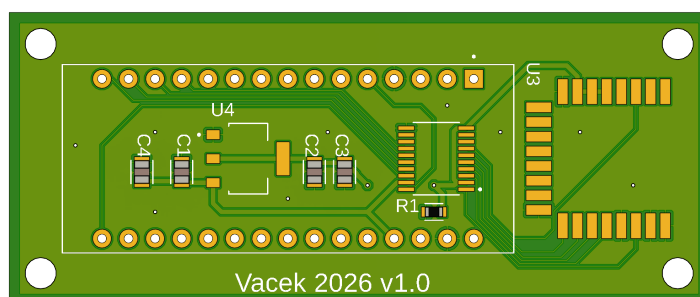


Obrázek 5.15 Schéma zapojení integrovaného převodníku TXS0108E

Při návrhu PCB jsem zohlednil i detaily, které mají v praxi zásadní vliv na kvalitu měření:

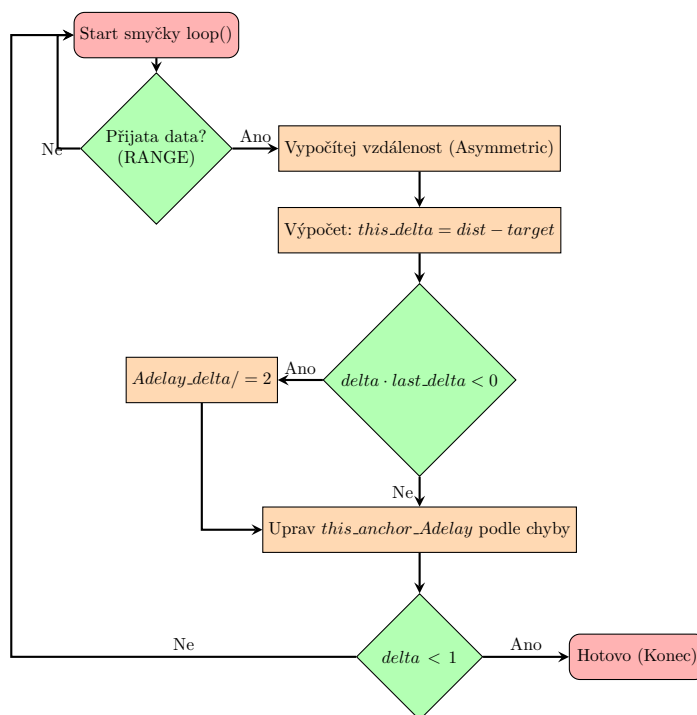
- krátké a přehledné vedení signálů SPI,
- dostatečně dimenzované napájecí cesty a lokální blokovací kondenzátory u citlivých obvodů,
- souvislou zemní plochu pro omezení rušení,
- mechanicky pevné umístění konektorů kvůli vibračním podvozku,
- testovací body pro rychlé oživení a diagnostiku.

Po přechodu na vlastní PCB zobrazené na obrázku 5.16, se výrazně zlepšila stabilita komunikace i opakovatelnost měření. I když výsledná přesnost stále závisí na aktuálních podmínkách prostředí (odrazy signálu, překážky, orientace antén), deska poskytla podstatně robustnější základ pro další ladění algoritmů navádění.



■ **Obrázek 5.16** Navržená deska plošných spojů pro obsluhu BU01

Dalším problémem, na který jsem narazil, byla nutnost kalibrace anténního zpoždění. Výrobce není schopen zajistit stejnou hodnotu anténního zpoždění pro každý vyrobený modul, proto je nutné jednotlivé moduly při prvním použití kalibrovat. Po vytvoření jednoduchého kalibračního programu, který vychází z ukázek knihovny Thomase Trojera [52], se mi podařilo moduly zkalibrovat na referenční vzdálenost. Celý proces kalibrace spočívá v postupném upravování hodnoty anténního zpoždění pro dosažení nejlepšího výsledku. Před spuštěním se nastaví kalibrační vzdálenost. Poté se postupně zvětšuje hodnota anténního zpoždění o konstantu. Jakmile je výsledkem měření hodnota menší než skutečná vzdálenost, konstanta se odečte a vydělí dvěma. Následně se hodnota zvětšuje pomaleji, dokud nebude naměřená vzdálenost opět větší. Tento proces se opakuje, dokud čip neměří pouze s malou odchylkou. Vývojový diagram kalibrační smyčky je vyobrazen na obrázku 5.17. Kalibrace byla provedena na vzdálenosti 3 m a následná odchylka měření se snížila přibližně na 10 cm. K samotnému měření je využívána funkce `computeRangeAsymmetric`. Tato funkce provádí měření metodou Round Trip Time, a tudíž eliminuje chybu způsobenou rozdílným časováním (*clock drift*) u tagu a anchoru, což je u měření technologií UWB klíčové pro dosažení centimetrové přesnosti.



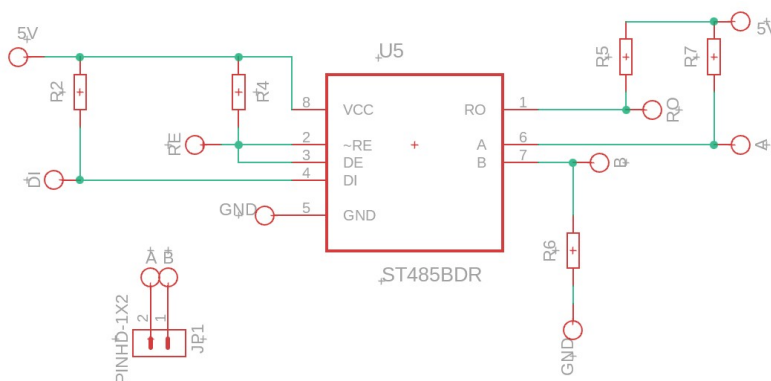
■ **Obrázek 5.17** Vývojový diagram kalibračního algoritmu Antenna Delay.

Celkově se tedy návrh vlastní UWB desky ukázal jako klíčový krok: snížil hardwarové kompromisy prototypového zapojení, zvýšil kvalitu dat pro navigaci a současně zachoval důležitý požadavek práce na cenově dostupné a reprodukovatelné řešení. Samotné desky byly vyrobeny firmou JLCPCB, se kterou mám již dlouhodobě dobrou zkušenost.

Ve chvíli, kdy již měření vzdálenosti fungovalo spolehlivě, bylo zapotřebí doplnit třetí modul pro určení orientace sledovaného objektu. Dva moduly umístěné na podvozku musí mezi sebou sdílet data a porovnávat vzdálenosti ke třetímu modulu, který nese sledovaná osoba. V první variantě jsem uvažoval o řízení obou modulů jedním Arduinem. Ukázalo se však, že kvůli požadované přesnosti je nutné umístit moduly dále od sebe (přibližně 50 cm), a na tuto vzdálenost již není SPI sběrnice dostatečně spolehlivá. Teoreticky by bylo možné snížit frekvenci SPI, to by však vedlo ke zpomalení měření.

Proto jsem zvolil architekturu s jedním Arduinem Nano pro každý UWB modul. První mikrokontrolér funguje jako hlavní uzel, který počítá výsledný úhel a komunikuje s nadřazeným Raspberry Pi přes USB. Druhý mikrokontrolér provádí měření vzdálenosti na své straně a předává výsledky hlavnímu uzlu. Pro spojení mezi oběma mikrokontroléry jsem zvolil sběrnici RS-485, protože je diferenciální a výrazně odolnější vůči rušení v porovnání s přímým TTL propojením na delší vzdálenost.

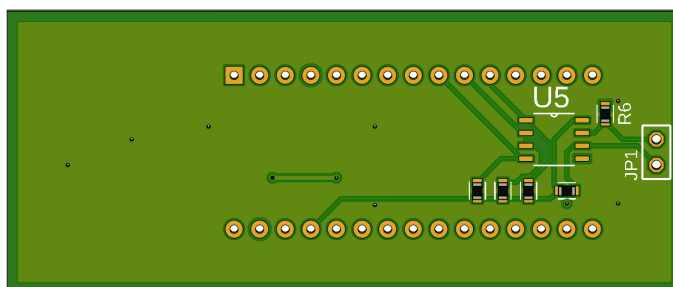
Pro komunikaci po RS-485 je na každé straně použit převodník MAX485. Převodník, jak je znázorněno na obrázku 5.18, je připojen k mikrokontroléru třemi řídicími a signálovými vodiči (RE, DI, RO), zatímco samotná komunikace mezi uzly probíhá poloduplexně po kroucené dvojlince (A, B). Toto řešení se v praxi osvědčilo jako stabilní i při provozu v blízkosti motorů a výkonové elektroniky.



Obrázek 5.18 Schéma zapojení převodníku MAX485

Po doplnění převodníku na zadní stranu desky, jak je vidět na obrázku 5.19, bylo možné spolehlivě realizovat i měření orientace. Pro dosažení přesného měření je potřeba, aby v jeden okamžik měřil vždy pouze jeden modul. Samotné programy jsem tedy koncipoval tak, že nejdříve měří hlavní uzel a druhý mikrokontrolér čeká na pokyn k měření. Po přijetí pokynu změří hodnoty a mezitím hlavní uzel čeká na odpověď. Po obdržení odpovědi hlavní uzel provede výpočet vzdálenosti a úhlu. Vyšší přesnosti a omezení chyb jsem dosáhl tím, že každý modul provádí měření dvacetkrát a následně pracuje s průměrem těchto hodnot. Současně se od začátku nezapočítávají záporné hodnoty ani hodnoty větší než 100 m. Počet dvaceti měření byl zvolen s ohledem na rychlost měření tak, aby systém zůstal být schopen dostatečně rychle reagovat na změny. Naměřené hodnoty lze dále vyhlazovat pomocí klouzavého průměru, tuto úlohu jsem se však rozhodl ponechat až řídicí aplikaci. Do budoucna je možné desku dále integrovat osazením mikrokontroléru ATmega328P přímo na PCB, čímž by odpadlo propojení s externím Arduino modulem. V této fázi jsem však tuto variantu záměrně nerealizoval, protože oddělené Arduino výrazně usnadňuje testování, ladění firmwaru i případnou rychlou výměnu řídicí jednotky.

Pro hotovou desku jsem následně navrhl uchycení k závitové tyči, která je přišroubována k horní části stranové desky podvozku. Tento způsob umožňuje jednoduché nastavení vzdálenosti těchto dvou modulů, ale zároveň zajišťuje, aby spojnice těchto modulů byla rovnoběžná s přední částí podvozku.



■ **Obrázek 5.19** Upravená deska plošných spojů o převodník MAX485

5.3 Řešení webové aplikace

Webovou aplikaci jsem navrhl jako lehké klientské rozhraní postavené na technologiích HTML, CSS a JavaScript. Toto řešení je pro daný projekt dostatečné, snadno udržitelné a umožňuje rychle reagovat na změny během vývoje i testování. Implementace je rozdělena do tří souborů: `index.html` (struktura stránky), `styles.css` (vizuální styl) a `control.js` (logika ovládání a komunikace se serverem).

Při návrhu jsem postupoval v tomto pořadí:

1. návrh kostry webu v HTML,
2. návrh grafického rozhraní a responzivního rozložení v CSS,
3. implementace obslužné logiky v JavaScriptu.

5.3.1 Kostra webové aplikace

Nejprve bylo nutné definovat, jaké informace má aplikace zobrazovat a jaké zásahy má uživateli umožnit. Kromě samotného řízení slouží rozhraní i jako nástroj pro ladění, protože Raspberry Pi nemá během provozu vždy k dispozici samostatný diagnostický výstup.

Do rozhraní jsem proto zahrnul:

- telemetrii (rychlost, režim, stav serv),
- GPS souřadnice,
- obraz z kamery,
- přepínače režimů navigace,
- ovládací prvky pro pohyb podvozku a naklápění kamery,
- volby pro práci s detekcí osob (zobrazení ohraničujících rámců, výběr cíle kliknutím).

Struktura stránky je rozdělena na levý panel, středovou videočást a pravý panel s doplňující telemetrií a ovládáním.

```
1 <div class="main-grid">
2   <div class="left-column">...</div>
3   <div class="panel video-panel">
4     
5   </div>
6   <div class="right-column">...</div>
7 </div>
```

■ Výpis kódu 1 Zjednodušená ukázka hlavní části HTML

Toto členění se ukázalo jako přehledné i při menších rozlišeních, protože informace nejsou překrývány přes video a uživatel má stále jasně oddělené diagnostické a řídicí prvky.

5.3.2 Grafické rozhraní

Při grafickém návrhu jsem se inspiroval rozhraními aplikací pro ovládání dronů a roverů. Cílem bylo vytvořit vizuálně výrazné, ale funkční prostředí, které umožní rychlou orientaci i během jízdy.

Návrh stylu je realizován samostatně v souboru `styles.css`. Rozhraní využívá panelové uspořádání, barevné zvýraznění důležitých hodnot a jednotný vzhled ovládacích prvků (tlačítka, joysticky, telemetrické prvky). Praktický přínos má zejména jasné oddělení videa od ovládacích prvků, což omezuje chyby při ovládání a zlepšuje čitelnost dat.

Pro telemetrii jsou navíc použity jednoduché vizuální indikátory (například barevné stavy a sloupcové ukazatele), které pomáhají rychle odhadnout stav systému bez nutnosti číst každou číselnou hodnotu.

5.3.3 Obsluha a komunikace

Aplikační logika je implementována v souboru `control.js`. Skript zajišťuje obsluhu joysticků, klávesnice, gamepadu, periodický sběr telemetrie i odesílání řídicích příkazů na server.

Odesílání řídicích dat

Všechny klíčové povely jsou odesílány ve formátu JSON na endpoint `/control`. Tím je zajištěna jednotná komunikace mezi klientem a řídicí aplikací:

```
1 function send() {  
2   fetch('/control', {  
3     method: 'POST',  
4     headers: {'Content-Type': 'application/json'},  
5     body: JSON.stringify({  
6       servo_x: Math.round(pan),  
7       servo_y: Math.round(tilt),  
8       speed: parseFloat(speed.toFixed(2)),  
9       turn: parseFloat(turn.toFixed(2))  
10    })  
11  }).catch(() => {});  
12 }
```

■ **Výpis kódu 2** Zjednodušená ukázka odesílání ve formátu JSON

Takto navržená funkce minimalizuje duplicitu v kódu, protože ji využívají všechny vstupní mechanismy (joystick, klávesnice i gamepad).

Ovládání pomocí joysticku

Aplikace používá dva virtuální joysticky:

- pravý joystick pro naklápění kamery (pan/tilt),
- levý joystick pro pohyb podvozku (speed/turn).

Pozice joysticků je zpracována pomocí **pointer** událostí, které dobře fungují jak na dotykových zařízeních, tak na běžném desktopu. U levého joysticku jsou příkazy odesílány průběžně při změně polohy, aby bylo řízení plynulé. U pravého joysticku je použita periodická smyčka, která postupně mění úhel serv a tím omezuje trhavý pohyb kamery.

Klávesnice a gamepad

Pro práci na počítači bez dotykové obrazovky je implementováno i ovládání klávesnicí: pohyb přes WASD, náklon kamery přes šipky. Dále je doplněna podpora USB ovladače (například pro Xbox), která umožňuje jemnější analogové řízení.

Tato vícevrstvá obsluha se v praxi ukázala jako důležitá při testování, protože umožňuje rychle přepínat mezi různými režimy ovládání podle aktuální situace.

Telemetrie a periodické aktualizace

Klient periodicky načítá data z endpointu `/telemetry` a aktualizuje hodnoty v rozhraní. Aktualizace probíhá v intervalu 200 ms, což představuje vhodný kompromis mezi plynulostí a zatížením systému.

Kromě telemetrie je možné z aplikace přepínat i mezi doplňkovými funkcemi, například zobrazení detekčních rámců (`/toggle_bboxes`) nebo výběr cílové osoby kliknutím do obrazu (`/select_target`).

Výběr cíle

Pro autonomní sledování osoby pomocí strojového vidění je nejdříve nutné zvolit správný objekt v obraze. Webová aplikace umožňuje zobrazení ohraničujících rámců nalezených osob ve videu. Uživatel následně musí zakliknout vybranou osobu. Kliknutí uživatele na video je převedeno na normalizované souřadnice a odesláno serveru:

```
1 const x = (event.clientX - rect.left) / rect.width;  
2 const y = (event.clientY - rect.top) / rect.height;  
3 fetch('/select_target', { ... body: JSON.stringify({x, y}) });
```

■ Výpis kódu 3 Ukázka normalizování souřadnic

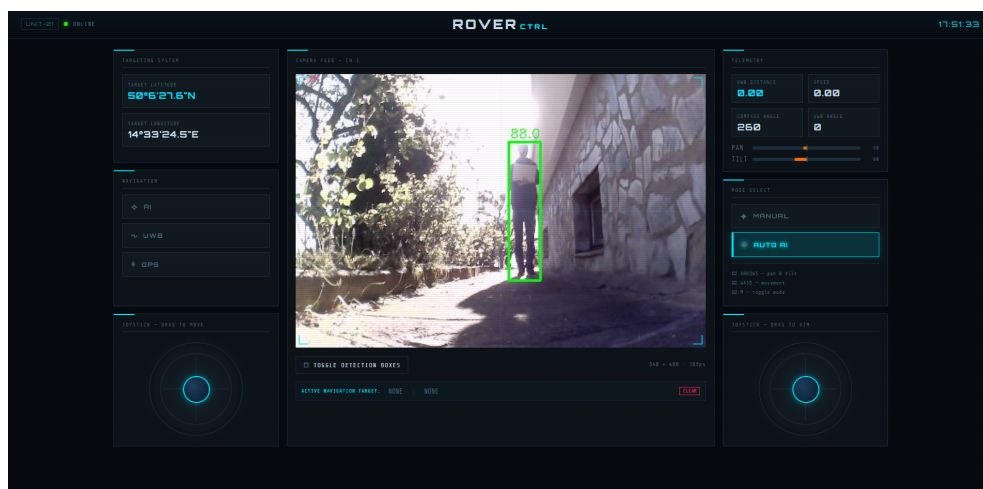
Tento přístup je nezávislý na aktuálním rozlišení zobrazení v prohlížeči a zjednodušuje následné mapování na detekované objekty. Po vybrání osoby má uživatel následně možnost zapnout samotné navádění.

Dále může uživatel pomocí webové aplikace zapnout navádění pomocí technologie UWB. Stroj následně sleduje osobu, pokud přijímá data z tagu. Tento způsob navádění lze kombinovat s naváděním pomocí strojového vidění.

Nakonec je možné zapnout také režim navádění pomocí GPS. Tento způsob navádění nelze aktivovat současně s ostatními režimy. Spouští se zadáním cílových souřadnic do polí v horní levé části stránky a následným stisknutím tlačítka GPS.

5.3.4 Shrnutí webové aplikace

Výsledná webová aplikace plní dvě hlavní role: operátorské rozhraní pro řízení robota a současně diagnostický panel pro vývoj a testování. Důraz byl kladen na jednoduchost, rychlou odezvu a snadnou rozšiřitelnost. Oddělení do tří souborů (`index.html`, `styles.css`, `control.js`) se osvědčilo z hlediska údržby i dalšího vývoje.



Obrázek 5.20 Webová aplikace

5.4 Řešení řídicí aplikace

Řídicí aplikaci jsem implementoval v jazyce Python. Hlavním důvodem byla dostupnost knihoven pro práci s Raspberry Pi, kamerovým vstupem, webovým serverem i akcelerátorem Hailo. Python současně umožňuje rychlý vývoj a dobrou čitelnost kódu, což bylo důležité zejména s ohledem na průběžné experimentování s více způsoby navádění.

Základ řídicí aplikace staví na ukázkách knihovny Hailo AI [53] určených pro Raspberry Pi AI HAT. V průběhu práce se ukázalo, že použitá verze knihovny nebyla v době vývoje plně kompatibilní s nejnovějším Raspberry Pi OS Trixie (Debian 13), ale stále fungovala na systému Bookworm (Debian 12). Tato nekompatibilita nebyla v dokumentaci výrazně zdůrazněna, proto mi její identifikace zabrala poměrně značné množství času. Po přechodu na podporovanou verzi systému již bylo zprovoznění knihovny bez větších obtíží. Knihovnu lze instalovat pomocí instalačního skriptu dostupného v jejím repozitáři, kde je zároveň popsán správný postup instalace. Před spuštěním Python skriptů je následně nutné aktivovat správné virtuální prostředí, do kterého instalační skript nainstaluje všechny potřebné moduly.

Knihovna Hailo poskytuje několik demonstračních aplikací, například detekci objektů, odhad pozice lidského těla, instanční segmentaci nebo odhad hloubky. V rámci této práce jsem vycházel především z ukázky pro detekci objektů, protože přímo odpovídá požadavku sledování osoby. Jako detekční model jsem zvolil modernější variantu YOLO, konkrétně model z řady YOLO11 [54], který poskytoval dostatečně spolehlivou detekci osob pro účely navádění. Tento model navíc dokáže detekovat i mnoho dalších objektů, například stoly, židle nebo květináče. Tohoto faktu lze následně využít i pro detekci překážek.

Dále jsem experimentoval i s vlastním modelem určeným pro detekci teni-

sových míčků. Dataset pro tento experiment byl připraven pomocí platformy Roboflow a samotné přeučení modelu proběhlo lokálně na grafické kartě GeForce RTX 2060. Tento experiment mi pomohl ověřit, že navržená architektura aplikace není pevně vázána pouze na jednu konkrétní třídu objektů a že je možné modely vyměňovat s minimálními zásahy do řídicí logiky. Toho lze využít například pro specializaci navádění v uzavřeném prostředí, kde je potřeba sledovat jiný objekt nebo rozeznávat pouze určité překážky.

5.4.1 Architektura aplikace

Řídicí aplikace je navržena jako vícevláknový systém. Raspberry Pi 5 disponuje vícejádrovým procesorem, proto bylo výhodné rozdělit jednotlivé činnosti do samostatných vláken a tím oddělit výpočetně náročné zpracování obrazu od webové komunikace a řízení hardwaru.

V aktuální verzi skriptu jsou využita zejména tato paralelně běžící vlákna:

- vlákno pro zpracování obrazu a detekci objektů založené na pipeline Hailo,
- vlákno navádění (`tracking_loop`),
- vlákno pro obsluhu serv a motorů (`servo_loop`),
- vlákno webového serveru Flask,
- vlákno kontrolující připojení ovládací aplikace.

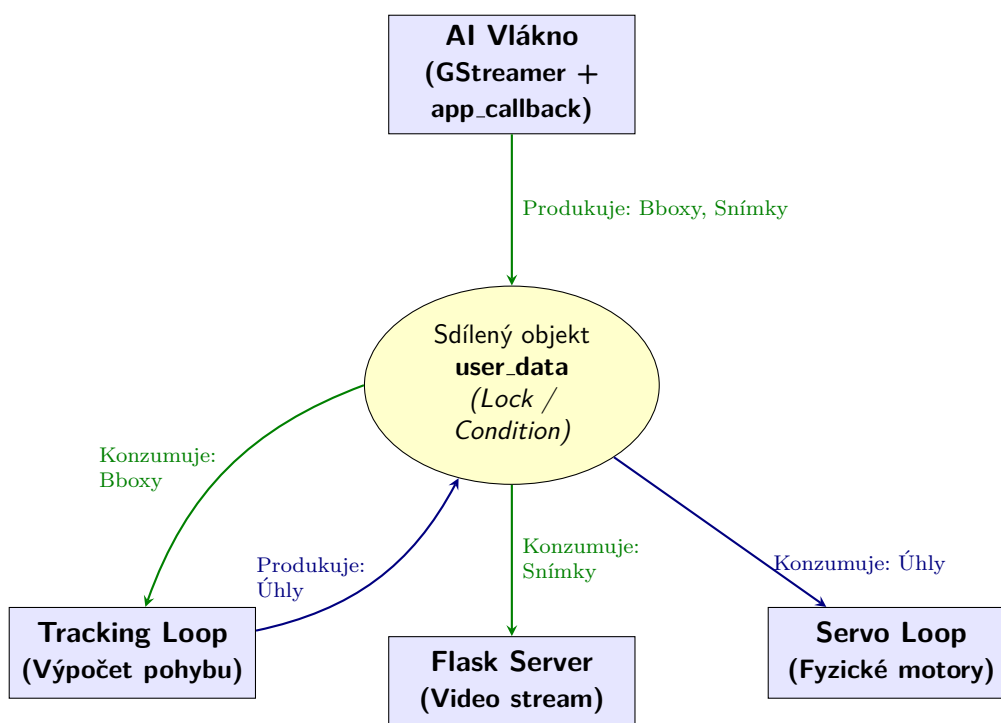
Toto rozdělení umožňuje zachovat přehlednou strukturu programu. Vlákna navíc mohou pracovat asynchronně, a podvozek tak dokáže rychleji reagovat. Jednotlivé části aplikace mezi sebou nekomunikují přímým voláním funkcí, ale prostřednictvím sdílené datové struktury. Ta je implementována třídou `user_app_callback_class`, která uchovává například poslední snímek z kamery, seznam detekovaných objektů, stav režimu řízení, hodnoty serv, rychlost a směr pohybu.

```
1 class user_app_callback_class(app_callback_class):  
2     def __init__(self):  
3         super().__init__()  
4         self.condition = Condition()  
5         self.lock = threading.Lock()  
6         self.frame = None  
7         self.bboxes = []  
8         self.servo_x = 130  
9         self.servo_y = 130  
10        self.speed = 0.0  
11        self.turn = 0.0
```

■ **Výpis kódu 4** Zjednodušená ukázka této sdílené struktury

V implementovaném systému hraje klíčovou roli asynchronní předávání dat. Hlavní vlákno aplikace (GStreamer) vystupuje v roli producenta, který v reálném čase plní sdílenou strukturu `user_app_callback_class` informacemi o detekovaných objektech.

Ostatní vlákna (`tracking_loop`, `servo_loop`, `web_app`) zde vystupují jako konzumenti. Aby nedocházelo k tzv. *race conditions* (souběhu operací s daty), je přístup ke sdílené struktuře chráněn zámky (`threading.Lock`). Pro efektivní využití procesoru je použit také mechanismus `threading.Condition`. Ten umožňuje konzumentům „spát“, dokud producent nevygeneruje nový snímek, čímž se eliminuje zbytečné vytížení CPU naprázdno běžícími cykly (*busy waiting*). Pro zajištění bezpečnosti robota je při pohybu nutné nepřetržité připojení webové aplikace. Tuto úlohu zajišťuje samostatné kontrolní vlákno. Při ztrátě spojení je aktivováno bezpečné zastavení podvozku. Kontrola je realizována periodickými dotazy mezi webovou a řídicí aplikací a při odmlce delší než půl sekundy se aktivuje bezpečnostní zastavení.



■ Obrázek 5.21 Schéma architektury Producent–Konzument v aplikaci.

5.4.2 Zpracování obrazu a detekce objektů

Jádrem aplikace je callback funkce volaná v okamžiku, kdy pipeline Hailo vrátí nový snímek s výsledky inference. Tato funkce získá obrazová data z bufferu, vyčte detekované objekty a vyfiltruje pouze osoby a možné překážky. Překážky lze nastavit jako výčet názvů tříd, pro které je model trénován. U každé relevantní detekce se ukládají souřadnice ohraničujícího rámce (*bounding boxu*), konfidence a případně i identifikátor sledovaného objektu.

```
1 roi = hailo.get_roi_from_buffer(buffer)
2 detections = roi.get_objects_typed(hailo.HAILO_DETECTION)
3 label = detection.get_label()
4
5 for detection in detections:
6     if label == "person" or label in obstacle:
7         bbox = detection.get_bbox()
8         confidence = detection.get_confidence()
9         ...
```

■ Výpis kódu 5 Zjednodušený princip callbacku

Pokud je k dispozici obrazový snímek, aplikace jej převede do formátu OpenCV, doplní základní diagnostické informace a uloží do sdílené struktury. Vedle samotného obrazu je uložen i seznam detekovaných objektů. Díky tomu může webová aplikace zobrazovat jak živý stream, tak i ohraničující rámce detekcí a zvýraznění vybrané osoby. Při navádění pomocí strojového vidění se zároveň ukládají souřadnice sledované osoby v obraze pro následné sledování. Sledovaná osoba je identifikována pomocí ID, které je součástí výsledku inference a které má každý detekovaný objekt. Při testování jsem však pozoroval občasnou změnu tohoto ID u detekovaných osob. Pipeline totiž může osobu na jednom snímku nedetekovat a na následujícím snímku ji znovu rozpoznat již s jiným ID. Bylo proto nutné implementovat algoritmus, který se tomuto stavu přizpůsobí. Aplikace si proto do sdílené datové struktury ukládá i poslední známé souřadnice cíle. Ve chvíli, kdy je sledovaná osoba ztracena, nastává fáze hledání. Během následujících tří snímků aplikace hledá osobu v místech, kde byla naposledy detekována. Pokud pipeline vrátí detekovanou osobu na podobném místě, ale s jinou hodnotou ID, systém toto ID převezme jako nové ID sledovaného objektu. Hodnota tří snímků se při testování ukázala jako vhodný kompromis: pipeline většinou během této doby osobu znovu detekuje a zároveň je malá pravděpodobnost záměny za jinou osobu.

5.4.3 Navádění

Automatické navádění je řešeno samostatnou smyčkou `tracking_loop`. Tato smyčka se vykonává pouze tehdy, pokud není aktivní manuální režim a zároveň

je spuštěn některý z režimů navádění.

Režim navigace pomocí GPS lze spustit pouze samostatně. Nejdříve je potřeba zadat cílové souřadnice ve webové aplikaci a následně spustit navádění.

V režimu sledování obrazu aplikace porovnává střed detekovaného cíle se středem snímku. Pokud je odchylka větší než definovaný práh, upraví se poloha pan a tilt serva tak, aby kamera sledovaný objekt znovu vystředila.

Dále se vyhodnocuje výška sledovaného cíle v obraze. Tato výška slouží jako přibližný údaj o vzdálenosti mezi podvozkem a cílem. Čím blíže je cíl, tím větší výšku zaujímá ve videu. V závislosti na tomto údaji je podvozek uveden do pohybu a současně se zatáčením snaží o vystředění kamery. Výsledkem je natočení podvozku směrem k cíli a jeho přibližování. Ve chvíli, kdy je cíl na snímku dostatečně vysoký, je podvozek zastaven a dále probíhá pouze sledování kamerou.

```
1 cx = user_data.frame_w // 2
2 cy = user_data.frame_h // 2
3
4 dx = user_data.target_x - cx
5 dy = user_data.target_y - cy
6
7 if abs(dx) > DEADZONE:
8     user_data.servo_x -= dx * SPEED
9 if abs(dy) > DEADZONE:
10    user_data.servo_y += dy * SPEED
```

■ Výpis kódu 6 Ukázka principu výpočtu otočení kamery

Tento přístup je záměrně jednoduchý, ale pro základní experimentální sledování osoby postačuje. Výhodou je snadná laditelnost parametrů, zejména hodnoty prahů a koeficientu přepočtu odchylky na natočení serv. V budoucnu by bylo možné navádění zpřesnit například o PID regulaci nebo o kombinaci s dalšími senzory.

Režim UWB načítá data o vzdálenosti a úhlu z Arduina pomocí sériové linky. Na jejich základě upravuje natočení kamery a uvádí podvozek do pohybu, pokud je vzdálenost od cíle větší než definovaný práh. Pokud jsou současně aktivní režim sledování pomocí UWB i režim založený na strojovém vidění, systém průběžně porovnává, zda sledovaná osoba ve videu leží v podobném úhlu od přední části podvozku jako osoba nesoucí tag s UWB modulem. Pokud je odchylka mezi těmito úhly větší než definovaný práh, je to považováno za chybu navádění (například kvůli záměně osob v obraze nebo ztrátě tagu) a navádění je z bezpečnostních důvodů ukončeno.

```
1 if ser and ser.in_waiting > 0:
2     line = ser.readline().decode('utf-8', errors='ignore')
3     .strip()
4     if line.startswith("ANGLE :"):
5         parts = line.replace("ANGLE :", "")
6             .replace("DISTANCE:", "").split(",")
7         with user_data.lock:
8             user_data.uwb_angle = float(parts[0].strip())
9             user_data.uwb_distance = float(parts[1].strip())
```

■ **Výpis kódu 7** Ukázka principu načítání dat ze sériové linky

5.4.4 Řízení hardwaru

Samostatná smyčka `servo_loop` zajišťuje přenos vypočtených hodnot na fyzické akční členy. Aplikace ovládá dvě oblasti hardwaru:

- serva pro pan a tilt kamery,
- pohon levé a pravé strany podvozku.

Serva i PWM výstupy motorů jsou obsluhovány přes řadič PCA9685 pomocí knihovny `adafruit_servokit`. Směr otáčení motorů je řízen samostatně pomocí GPIO výstupů. Pro zjednodušení je řízení motorů zapouzdřeno do funkce `set_motor()`, která podle požadovaného směru nastaví logické úrovně na pinech a zároveň upraví střihu PWM.

```
1 def set_motor(motor, direction, speed):
2     if direction == 'forward':
3         ...
4     elif direction == 'backward':
5         ...
6     else:
7         ...
8     pwm_channel.duty_cycle = int(speed * 65535)
```

■ **Výpis kódu 8** Ukázka funkce `set_motor()`

V řídicí smyčce se dále porovnávají aktuální a poslední odeslané hodnoty. Tím se omezuje zbytečné opakované nastavování hardwaru při malých změnách a snižuje se mechanické namáhání serv. Současně je zde implementována základní logika převodu hodnot `speed` a `turn` na konkrétní pohyb podvozku vpřed, vzad nebo otáčení na místě.

5.4.5 Webová komunikace

Součástí řídicí aplikace je i integrovaný webový server postavený na frameworku Flask. Ten poskytuje jak samotné HTML rozhraní, tak datové endpointy pro komunikaci s frontendem.

K dispozici jsou zejména tyto endpointy:

- / pro načtení hlavní stránky,
- /video_feed pro stream videa ve formátu MJPEG,
- /control pro příjem řídicích příkazů z webové aplikace,
- /telemetry pro poskytování telemetrických údajů,
- /toggle_bboxes pro zapnutí a vypnutí vykreslování detekcí,
- /select_target pro výběr sledované osoby kliknutím do obrazu.

Stream videa je realizován generátorem, který čeká na nový snímek ve sdílené struktuře a případně do něj doplňuje vykreslení detekčních rámců, poté jej odešle jako JPEG součást MJPEG streamu. Tento přístup je výhodný tím, že nevyžaduje složitější streamovací infrastrukturu a pro potřeby ovládacího rozhraní je zcela dostačující.

```
1 @app.route("/telemetry")
2 def telemetry():
3     return jsonify(
4         speed=user_data.speed,
5         servo_x=int(user_data.servo_x),
6         servo_y=int(user_data.servo_y),
7         mode="MANUAL" if user_data.manual_mode else "AUTO"
8     )
```

■ Výpis kódu 9 Ukázka obsluhy endpointu pro telemetrii

Výhoda tohoto řešení spočívá v tom, že je webové rozhraní a řídicí logika zůstávají součástí jednoho procesu a není nutné zavádět další komunikační vrstvu mezi serverem a hlavní aplikací. Po testování komunikace s webovou aplikací jsem však zjistil, že je stream videa občas trhaný. Charakter tohoto trhání nasvědčoval zahlcení komunikace ze strany ovládání. Rozhodl jsem se proto přejít na protokol WebSocket.

Přechod z protokolu HTTP (`fetch`) na WebSockets představuje zásadní posun od pasivního dotazování k aktivní, obousměrné komunikaci v reálném čase. Zatímco `fetch` funguje na principu „dotaz–odpověď“, kdy prohlížeč musí pro každou aktualizaci otevřít nové spojení s relativně vysokou režii, WebSocket vytvoří jeden trvalý kanál, který zůstává otevřený. Hlavní výhodou

je velmi nízká latence, protože data proudí okamžitě bez nutnosti opakovaného navazování spojení. Tímto přechodem jsem eliminoval trhané ovládání a umožnil řídicí aplikaci posílat změny v telemetrii ihned v okamžiku, kdy nastanou, čímž se zároveň výrazně snížila zátěž procesoru i Wi-Fi síť.

```

1 def send_telemetry_socket():
2     data = {
3         "speed": round(user_data.speed, 2),
4         "servo_x": int(user_data.servo_x),
5         "servo_y": int(user_data.servo_y),
6         "mode": "MANUAL" if user_data.manual_mode else "AUTO",
7     }
8     socketio.emit('telemetry_update', data)

```

■ **Výpis kódu 10** Ukázka obsluhy telemetrie využívající WebSocket

Stejným způsobem byly upraveny také funkce v souboru `control.js`, který náleží webové aplikaci

```

1 function send() {
2     socket.emit('control_signal', {
3         servo_x: Math.round(pan),
4         servo_y: Math.round(tilt),
5         speed: parseFloat(speed.toFixed(2)),
6         turn: parseFloat(turn.toFixed(2))
7     });
8 }

```

■ **Výpis kódu 11** Ukázka funkce využívající WebSocket v souboru `control.js` náležícím webové aplikaci

5.4.6 Inicializace a spuštění systému

Řídicí aplikaci lze spouštět pomocí příkazové řádky. Při spuštění aplikace založené na knihovně Hailo AI se na monitoru zobrazí okno s obrazem videa a ohraničujícími rámci všech detekovaných objektů, které daný model umělé inteligence dokáže rozpoznat. V dostupné dokumentaci jsem bohužel neshledal, jak toto zobrazování vypnout. To představovalo problém, protože při absenci připojeného displeje přes HDMI nebo interní port Raspberry Pi aplikace končila chybou `segmentation fault`. Bylo tedy nutné najít způsob, jak tuto chybu obejít. Nakonec jsem zvolil softwarovou simulaci displeje pomocí programu `Xvfb`. Příkaz `xvfb-run` vytvoří v paměti virtuální grafické rozhraní. Standardně knihovna Hailo nepředává přes pipeline dále obrazová data. Pro jejich předání do vlákna pro zpracování obrazu je potřeba program spouštět

s příznakem `--use-frame`. Při použití jiného než základního YOLO modelu je navíc nutné nastavit i příznaky s cestou k souboru modelu a k souboru se seznamem tříd.

Ukázka spuštění aplikace pomocí příkazové řádky s USB kamerou:

```
xvfb-run -a python3 detection_flask.py --input usb --use-frame
```

Při startu programu následně proběhne inicializace GStreameru, nastavení serv, vytvoření sdílené datové struktury a spuštění pomocných vláken. Následně je spuštěna Hailo detekční pipeline, která tvoří hlavní zdroj obrazových dat pro zbytek aplikace.

```
1 threading.Thread(target=tracking_loop,  
2                 args=(user_data,), daemon=True).start()  
3 threading.Thread(target=servo_loop,  
4                 args=(user_data,), daemon=True).start()  
5 threading.Thread(target=lambda: web_app.run(...),  
6                 daemon=True).start()  
7 threading.Thread(target=safety,  
8                 args=(user_data,), daemon=True).start()  
9  
10 app = GStreamerDetectionApp(app_callback, user_data)  
11 app.run()
```

■ Výpis kódu 12 Zjednodušená ukázka spuštění jednotlivých vláken

Tento způsob spuštění umožňuje, aby všechny dílčí části systému běžely paralelně a reagovaly v reálném čase na vstupy od uživatele i na data z kamery.

5.4.7 Shrnutí řídicí aplikace

Výsledná řídicí aplikace spojuje několik dříve navržených částí systému do jednoho funkčního celku. Zajišťuje akvizici a zpracování obrazu, webovou komunikaci, řízení pohonu i obsluhu serv. Důležitou vlastností je modulární struktura, díky níž je možné jednotlivé části dále rozšiřovat – například doplnit přesnější regulaci pohybu nebo pokročilejší filtrování dat.

Skript v současné podobě obsahuje všechny klíčové stavební bloky potřebné pro správnou funkci systému a dostatečně dokumentuje způsob, jakým byly jednotlivé části navrženy a propojeny.

Kapitola 6

Testování

Testování jsem rozdělil do několika navazujících částí. Nejprve jsem ověřoval jednotlivé subsystémy samostatně, aby bylo možné určit jejich přesnost, stabilitu a praktická omezení. Následně jsem provedl testování kompletní sestavy, při kterém jsem hodnotil mechanické vlastnosti podvozku, chování napájecí soustavy i úspěšnost jednotlivých metod navádění v reálném provozu.

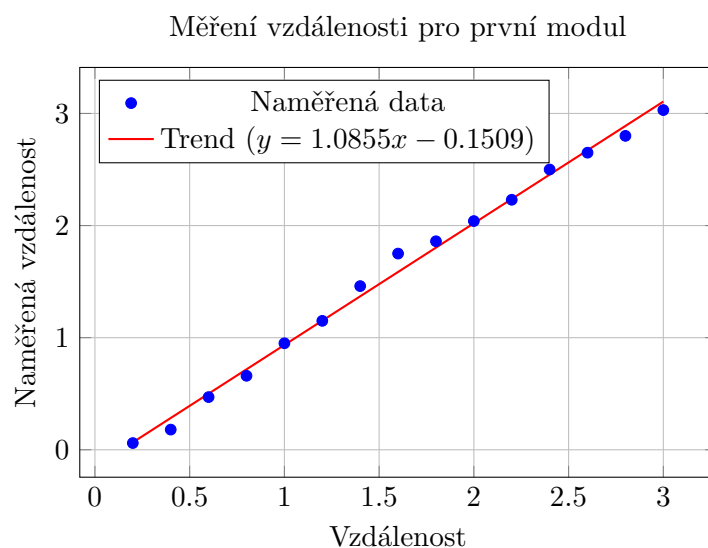
Ve všech případech bylo cílem nejen ověřit funkčnost navrženého řešení, ale také určit, zda dosažené parametry postačují pro zamýšlené použití robotické platformy.

6.1 Testování UWB modulů

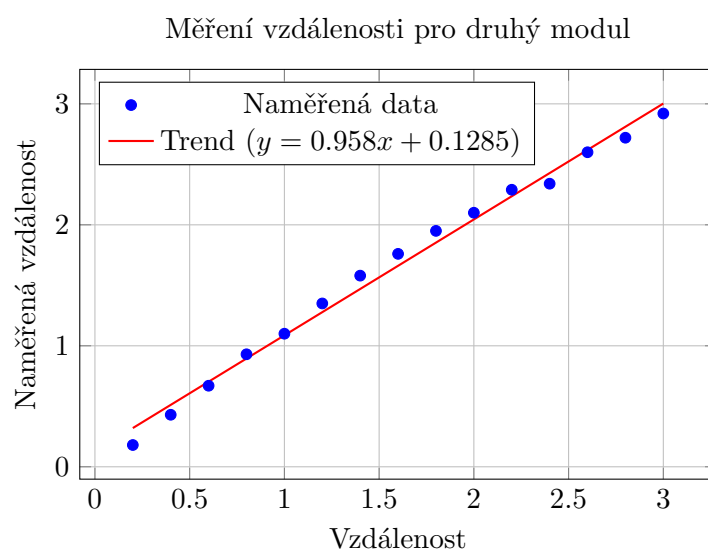
6.1.1 Měření vzdálenosti jednotlivých modulů

Na začátku jsem testoval měření vzdálenosti pomocí UWB modulů. Jelikož jsou pro výpočet úhlu natočení potřeba dva moduly umístěné na podvozku, bylo nejprve nutné každý modul samostatně zkalibrovat a následně otestovat. Každý modul jsem kalibroval na vzdálenost 3 m. Výsledné hodnoty anténního zpoždění byly 16558 pro první modul a 16545 pro druhý modul.

Následně jsem měřil vzdálenosti v krocích po 20 cm a postupně se s tagem přibližoval ke kotvě umístěné na podvozku. Měření jsem prováděl v místnosti bez překážek, aby byly co nejvíce omezeny odrazy signálu a další rušivé vlivy. V následujících grafech 6.1 a 6.2 jsou zobrazeny výsledky měření jednotlivých modulů.



■ **Obrázek 6.1** Porovnání skutečné vzdálenosti s naměřenou pro první modul

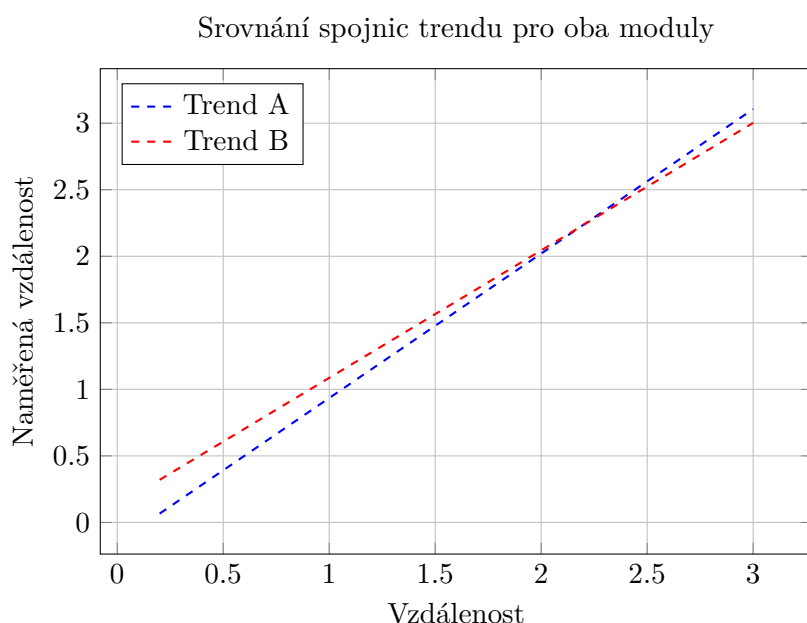


■ **Obrázek 6.2** Porovnání skutečné vzdálenosti s naměřenou pro druhý modul

6.1.2 Měření úhlu natočení

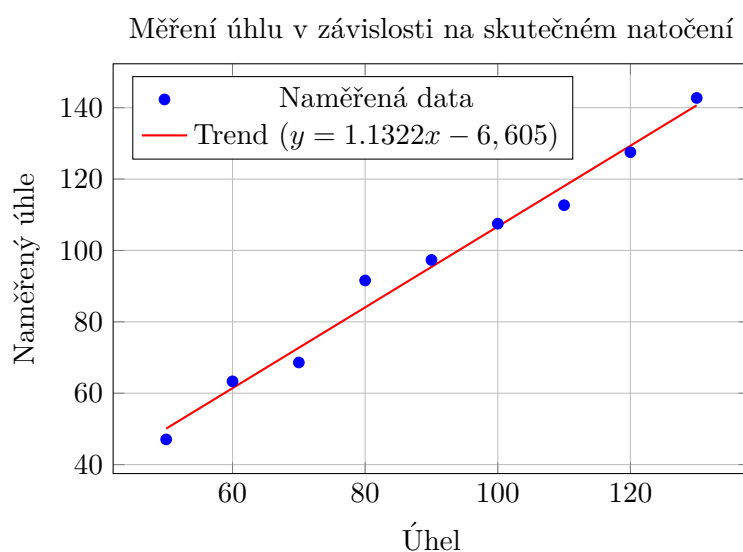
Pro dosažení co nejlepších výsledků při měření úhlů je potřeba, aby oba moduly vykazovaly podobnou odchylku. Na grafu 6.3 jsou proto znázorněny spojnice trendu pro oba moduly. Z výsledků je patrné, že použité moduly nemají shodnou odchylku v celém rozsahu vzdáleností. Se zmenšující se vzdáleností jejich vzájemná odchylka roste, podobný jev se projevuje i při větších vzdálenostech, avšak s menší intenzitou.

Průnik obou trendových přímek nastává přibližně ve vzdálenosti 2 metry od tagu. Tuto vzdálenost jsem proto zvolil jako výchozí pro měření úhlu natočení. Z praktického hlediska jde zároveň o vhodnou vzdálenost i pro samotné následování osoby robotem.

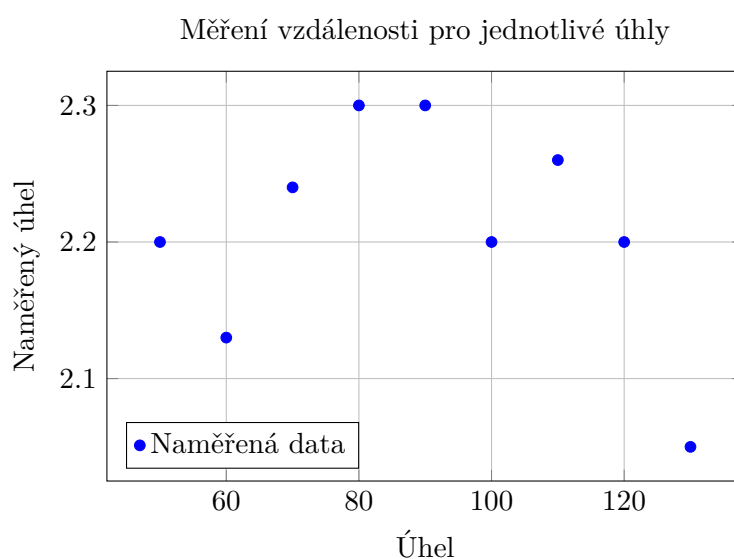


■ **Obrázek 6.3** Srovnání naměřených sad dat.

Na grafu 6.4 jsou vyobrazeny naměřené hodnoty v závislosti na reálném natočení podvozku vůči tagu. Z grafu je patrná nepřesnost měření pomocí těchto modulů, která je bohužel ještě násobena. Pro rozhodnutí jestli je sledovaný objekt více v pravo či vlevo jsou ale tyto měření dostatečně přesná. Na grafu 6.5 je již tato zkmaná nepřesnost jasně viditelná. Při měření pouze jedním modulem, byla přesnost do 10 centimetrů. V případě měření dvěma moduly je již maximální možná odchylka 30 centimetrů. Tato hodnota je ale stále ještě pro tento způsob navádění dostatečně malá.



Obrázek 6.4 Porovnání skutečného úhlu natočení s naměřeným



Obrázek 6.5 Porovnání naměřené vzdálenosti při zvolených úhlech

6.2 Testování inference strojového vidění

Následně jsem testoval rychlost a kvalitu rozpoznávání objektů pomocí strojového vidění. Do řídicí aplikace jsem přidal časové značky, na jejichž základě jsem měřil rychlost produkce nových dat z pipeline. Výsledkem bylo stabilních 25 až 30 snímků za sekundu.

Tato rychlost je dostatečná jak pro manuální navádění pomocí obrazu z kamery, tak pro samotné autonomní navádění umělou inteligencí, která je díky

tomu schopna dostatečně rychle reagovat na pohyb cíle. Prakticky se také potvrdilo, že možnost obnovení sledování při výpadku detekce na jednom až dvou snímcích výrazně zvyšuje stabilitu systému při otřesech kamery nebo při rychlém otáčení podvozku.

Během testování se ukázalo, že kvalita detekce je dostatečná především v situacích, kdy je cíl dobře osvětlený a není výrazně zakryt jinými objekty. Naopak v horších světelných podmínkách nebo při částečném zakrytí osoby docházelo k občasným krátkodobým výpadkům detekce. Tyto výpadky však navržený systém ve většině případů dokázal překlenout díky ukládání poslední známé polohy cíle a následnému znovunavázání sledování.

6.3 Testování GPS a magnetometru

Samotný GPS modul nevyžaduje kalibraci, a proto jsem jeho testování zaměřil především na ověření parametrů udávaných výrobcem. Modul byl schopen navázat spojení se satelity při použití mimo budovu nebo v blízkosti okna. Prakticky dosažená přesnost odpovídala očekávaným parametrům a byla dostatečná pro orientační navádění podvozku k zadanému cíli.

Naopak senzor magnetického pole bylo nutné kalibrovat, protože maximální hodnoty magnetické indukce se mění v závislosti na prostředí a míře rušení od okolní elektroniky. Po zkalibrování pomocí vytvořeného kalibračního programu dosahovala odchylka od skutečných hodnot maximálně přibližně 20°. Tato hodnota je pro dané použití akceptovatelná, protože navádění je ukončováno již ve vzdálenosti přibližně 2 m od cíle a magnetometr zde slouží především k orientačnímu určení směru.

6.4 Testování webové aplikace

Webovou aplikaci jsem testoval současně s řídicí aplikací v manuálním režimu a zároveň jsem sledoval telemetrická data. Ovládání se ukázalo jako funkční a dostatečně spolehlivé pro běžné použití. Aplikace správně přenášela řídicí povely, zobrazovala obraz z kamery a umožňovala přepínání jednotlivých režimů navádění bez zásadních problémů.

Jako slabší místo současného řešení se ukázal přenos obrazu pomocí MJPEG streamu. Tento způsob je implementačně poměrně jednoduchý a pro účely této práce se ukázal jako dostatečný, avšak z hlediska latence a efektivity přenosu není optimální. Do budoucna by proto bylo vhodné uvažovat o přechodu na UDP přenos videa, který by mohl nabídnout nižší zpoždění a plynulejší přenos obrazu. Takové řešení je však implementačně výrazně náročnější, a proto jsem pro tuto práci ponechal jednodušší variantu s MJPEG, která dostatečně splnila požadavky na funkčnost celého systému.

6.5 Testování celého systému

6.5.1 Mechanické a napájecí vlastnosti

Následně jsem testoval podvozek jako celek. Samotný podvozek je dostatečně tuhý a bez problémů unese veškerou elektroniku obsaženou v návrhu. Hlavním nedostatkem se ukázaly být použité motory. Na hladkém podkladu, jako je lino, plovoucí podlaha nebo beton, neměly problém s pohybem podvozku. Na površích s vyšším třením, například na koberci, však již měly problém s otočením na místě. Při dalším návrhu by proto bylo vhodné použít výkonnější motory.

Napájecí soustava dokáže bez problémů současně napájet pohonné motory i obě serva. Baterie určené pro pohyb bylo nutné dobít až po více než dvou hodinách provozu, takže robot není při delších sledovacích úlohách zásadně omezen kapacitou napájení.

6.5.2 Ověření metod navádění

Samotné navádění bylo úspěšně ověřeno u všech tří uvažovaných metod. GPS navádění jsem testoval na betonovém hřišti, kde bylo možné dobře pozorovat reakci systému na změnu cílových souřadnic a orientaci vůči směru jízdy. Navádění pomocí UWB i navádění pomocí strojového vidění jsem testoval jak v interiéru, tak v exteriéru.

V případě UWB navádění se jako hlavní omezení ukázala citlivost na přesnost kalibrace a na geometrii rozmístění modulů. Přesto bylo řešení dostatečné pro orientační následování osoby. Navádění pomocí strojového vidění se ukázalo jako velmi dobře použitelné zejména v prostředí s dobrými světelnými podmínkami, kde systém dokázal stabilně sledovat vybranou osobu. GPS navádění bylo naopak nejvhodnější pro pohyb k pevně zadanému cíli v otevřeném prostoru, avšak méně vhodné pro přesné finální navedení.

Z testování celého systému vyplynulo, že jednotlivé naváděcí metody se vhodně doplňují. Strojové vidění je vhodné pro bezprostřední sledování cíle, UWB poskytuje záložní informaci o směru a vzdálenosti a GPS umožňuje orientační navigaci v otevřeném prostoru. Tato kombinace potvrzuje, že navržená platforma může sloužit jako univerzální základ pro další vývoj autonomních robotických systémů.

6.6 Shrnutí kapitoly

Provedené testy potvrdily funkčnost všech hlavních částí navrženého systému. UWB moduly poskytují použitelná data pro orientační určování vzdálenosti a směru, strojové vidění dosahuje dostatečné rychlosti i stability pro sledování osoby a GPS společně s magnetometrem umožňují základní orientaci v pro-

storu. Zároveň se při testování ukázala i hlavní omezení současného řešení, zejména nižší výkon použitých motorů a omezená přesnost některých senzorů.

Z hlediska cílů práce je však výsledný systém funkční a prakticky použitelný. Platforma umožňuje jak autonomní navádění různými metodami, tak i ruční zásah uživatele, a vytváří tak vhodný základ pro další rozvoj a budoucí konstrukční i softwarová vylepšení.



Kapitola 7

Závěr

Cílem této práce bylo navrhnout, realizovat a ověřit funkčnost cenově dostupné robotické platformy založené na mikropočítači Raspberry Pi 5, která umožní autonomní navádění pomocí strojového vidění a současně využití záložních metod navádění založených na technologiích UWB a GPS. Na základě provedeného návrhu, realizace i testování lze konstatovat, že se tento cíl podařilo splnit.

V rámci práce byl navržen a vyroben vlastní podvozek kombinující 3D tištěné konstrukční prvky s překližkovou nosnou částí. Toto řešení se ukázalo jako vhodný kompromis mezi výrobní jednoduchostí, možností rychlých konstrukčních úprav, pevností a cenou. Současně byla navržena a realizována elektronická část systému, zahrnující řídicí aplikaci na Raspberry Pi, webové rozhraní pro vzdálené ovládání a vlastní desky plošných spojů pro obsluhu UWB modulů. Důležitým přínosem práce je právě propojení mechanického návrhu, návrhu elektroniky a softwarové integrace do jednoho funkčního celku.

Praktické testy potvrdily, že navržený systém je schopen funkčního provozu ve všech třech uvažovaných režimech navádění. Navádění pomocí strojového vidění se ukázalo jako dobře použitelné pro přímé sledování osoby, a to zejména v prostředí s dobrými světelnými podmínkami. UWB navádění poskytlo použitelnou záložní informaci o vzdálenosti a směru cíle, přestože jeho přesnost byla limitována vlastnostmi použitých modulů a nutností pečlivé kalibrace. GPS navádění se osvědčilo především pro orientační pohyb k pevnému cíli v otevřeném prostoru. Výsledkem je tedy platforma, která umožňuje využití více naváděcích metod podle charakteru prostředí a konkrétní situace.

Za důležité považuji také to, že práce nevedla pouze k vytvoření jednorázového prototypu, ale k návrhu systému, který je dále rozšiřitelný. Architektura řídicí aplikace umožňuje výměnu modelů strojového vidění, další úpravy webového rozhraní i rozšíření o pokročilejší algoritmy řízení. Stejně tak konstrukce podvozku i návrh UWB desek vytvářejí základ pro další iterace a technické zlepšování. Výsledná platforma tak může sloužit nejen jako demonstrátor funkčnosti, ale i jako výchozí základ pro navazující vývoj auto-

nomního robota.

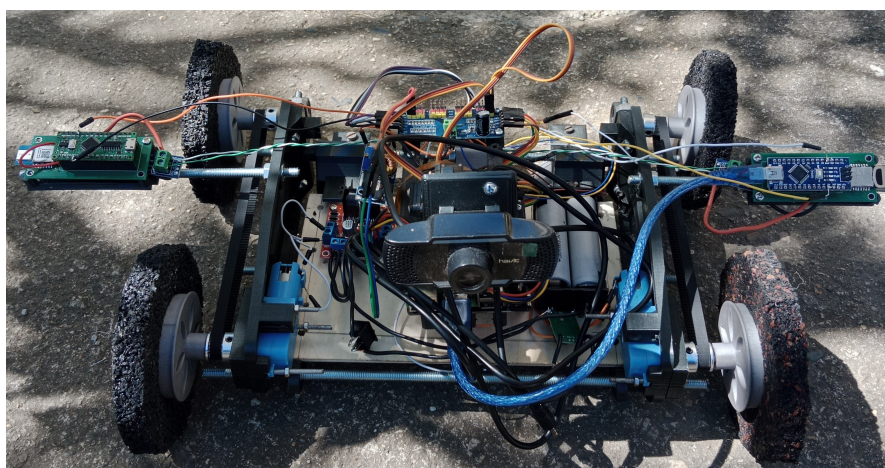
Současně se během řešení ukázala i omezení navrženého systému. Z mechanického hlediska jsou slabším místem použité pohonné motory, jejichž výkon není dostatečný pro všechny typy povrchů. Z pohledu sensoriky je nutné počítat s omezenou přesností magnetometru a s tím, že UWB měření je citlivé na kalibraci i okolní podmínky. U strojového vidění se pak jako limitující ukazují horší světelné podmínky, částečné zakrytí cíle nebo vlastnosti použitého video přenosu. Přesto tato omezení nebrání praktické použitelnosti systému pro účely, pro které byl navržen.

Do budoucna by bylo vhodné zaměřit se zejména na několik směrů dalšího rozvoje. Prvním je zlepšení mechanické části, zejména použití výkonnějších motorů a případně doplnění přesnější zpětné vazby o pohyb podvozku. Druhým směrem je zpřesnění navigace, například integrací dalších senzorů nebo použitím pokročilejších filtračních a regulačních metod. Třetím směrem je vylepšení přenosu videa a webového rozhraní, například přechodem na efektivnější způsob streamování s nižší latencí. Zajímavou možností představuje i nasazení specializovaných modelů strojového vidění pro konkrétní prostředí nebo konkrétní typ sledovaného cíle.

Z pohledu celkového zhodnocení práce považuji za největší přínos to, že se podařilo vytvořit funkční, modulární a relativně dostupnou platformu, která spojuje více metod autonomního navádění do jednoho systému. Práce tak splnila stanovené cíle a současně ukázala řadu praktických poznatků, které mohou být využity při dalším návrhu podobných robotických zařízení.

..... Příloha A

Přílohy



■ Obrázek A.1 Kompletní podvozek

Bibliografie

1. RPISHOP.CZ. *SunFounder Robotické auto Zeus Car, Arduino UNO* [online]. 2026. [cit. 2026-04-13]. Dostupné z: <https://rpishop.cz/498005/sunfounder-roboticke-auto-zeus-car-arduino-uno/>.
2. RPISHOP.CZ. *SunFounder Raspberry Pi AI Car Kit (PiCar-X)* [online]. 2026. [cit. 2026-04-13]. Dostupné z: <https://rpishop.cz/pro-raspberry-pi/6404-sunfounder-raspberry-pi-ai-car-kit-picar-x.html>.
3. RPISHOP.CZ. *4Tronix M.A.R.S. Rover Robot pro Raspberry Pi Zero* [online]. 2026. [cit. 2026-04-13]. Dostupné z: <https://rpishop.cz/517321/4tronix-m-a-r-s-rover-robot-pro-raspberry-pi-zero/>.
4. RPISHOP.CZ. *Waveshare JetRacer 2GB AI Kit, AI závodní robot pro Jetson Nano 2GB* [online]. 2026. [cit. 2026-04-13]. Dostupné z: <https://rpishop.cz/pro-jetson-nano/4291-waveshare-jetracer-2gb-ai-kit-ai-zavodni-robot-pro-jetson-nano-2gb.html>.
5. PRITZ, Helmut. *Autonomní řízení* [online]. 2023, 2023-7-11. [cit. 2026-04-13]. Dostupné z: <https://www.md-elektronik.com/cs/autonomni-řízení/>.
6. HANSEN Stig, Ulrik. *Object Detection: Models, Use Cases, Examples* [online]. 2023, 2023-4-14. [cit. 2026-04-13]. Dostupné z: <https://encord.com/blog/object-detection/>.
7. S.R.O., TOXIN. *Konvoluční neuronové sítě pomáhají detekovat a blokovat nevhodný obsah sociálních sítí* [online]. 2021, 2021-5-14. [cit. 2026-04-13]. Dostupné z: <https://encord.com/blog/object-detection/>.
8. GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
9. JOCHER, Glenn. *Performance Metrics Deep Dive* [online]. 2023. [cit. 2026-04-13]. Dostupné z: <https://docs.ultralytics.com/guides/yolo-performance-metrics/>.

10. BARTOSZAK, Rafal. *Raspberry Pi – technologie 21. století, která mění pravidla hry* [online]. 2023, 2023-10-5. [cit. 2026-04-13]. Dostupné z: <https://botland.cz/blog/raspberry-pi-technologie-21-stoleti-kt-era-meni-pravidla-hry/>.
11. RPISHOP.CZ. *Raspberry Pi 5 - 16GB RAM* [online]. 2026. [cit. 2026-04-13]. Dostupné z: <https://rpishop.cz/563516/raspberry-pi-5-16gb-ram/>.
12. VILROS.COM. *Raspberry Pi 5 Pinout* [online]. 2026. [cit. 2026-04-13]. Dostupné z: https://vilros.com/pages/raspberry-pi-5-pinout?srsltid=AfmB0oq75RL7a48j581T_hx6MtXDcmfYVFQXbQQAXq-fMeYacSZFgzp3.
13. WOSTL, Marek. *Maker UNO ve výuce (1/5): Co je to mikrokontrolér?* 2019-11. Dostupné také z: <https://blog.zonepi.cz/maker-uno-ve-vyuce-1-5-co-je-to-mikrokontroler/>. Citováno: 18. 4. 2026.
14. ARIAT TECH. *Comprehensive Analysis of Microcontrollers*. 2024-01. Dostupné také z: <https://www.ariat-tech.cz/blog/comprehensive-analysis-of-microcontrollers.html>. Citováno: 18. 4. 2026.
15. TIPAEU. *Jak na Arduino (1. díl): Co je Arduino a jak s ním začít*. 2017-01. Dostupné také z: <https://www.tipa.eu/cz/czjak-na-arduino-1-dil-co-je-arduino-a-jak-s-nim-zacit/t-1650/>. Citováno: 18. 4. 2026.
16. WIKIPEDIE. *Satelitní navigace* [online]. 2026. [cit. 2026-04-13]. Dostupné z: https://cs.wikipedia.org/wiki/Satelitn%C3%AD_navigace.
17. LVIV POLYTECHNIC NATIONAL UNIVERSITY. *About GNSS*. Geoterrace: GNSS Network, 2024. Dostupné také z: <https://geoterrace.lpnu.ua/en/gnss-network/about-gnss>. Accessed: 2026-04-13.
18. A.S., alza. *UWB je bezdrátová technologie budoucnosti* [online]. 2022, 2022-2-7. [cit. 2026-04-13]. Dostupné z: <https://www.alza.cz/uwb>.
19. INTERNATIONAL TELECOMMUNICATION UNION. *Recommendation ITU-R SM.1755: Characteristics of operational transceiver stations for the monitoring of emissions from satellite systems*. Geneva, Switzerland, 2006. Tech. zpr. ITU. Dostupné také z: https://www.itu.int/dms_publications/itu-r/rec/sm/R-REC-SM.1755-0-200605-I!!PDF-E.pdf.
20. POUZA, Jakub. *Co je to UWB? Nová technologie zastoupí Wi-Fi, Bluetooth i NFC a slibuje velké věci*. 2020-11. Dostupné také z: <https://www.zive.cz/clanky/co-je-to-uwb-nova-technologie-zastoupi-wi-fi-bluetooth-i-nfc-a-slibuje-velke-veci/sc-3-a-206752/default.aspx>. Citováno: 18. 4. 2026.
21. AIMTEC. *Váš signál je na přeletu opožděn*. 2020-02. Dostupné také z: <https://insights.aimtecglobal.com/vas-signal-je-na-priletu-opozden/>. Citováno: 18. 4. 2026.

22. RAJAN, Sree Ratna. Exploring Ultra-Wideband Technology for Micro-Location-Based Services. *Microwave Journal*. 2021. Dostupné také z: <https://www.microwavejournal.com/articles/36143-exploring-ultra-wideband-technology-for-micro-location-based-services?page=2>. Citováno: 18. 4. 2026.
23. MURPHY, Brendan. *What is the SPI Protocol and How to Debug SPI Communication?* [online]. 2023, 2023-12-31. [cit. 2024-04-13]. Dostupné z: <https://www.totalphase.com/blog/2020/07/what-is-spi-protocol-how-to-debug-spi-communication/>.
24. SHARMA, Meetu; KHAN, Afia. *I2C Interface Timing Specifications and Constraints* [online]. 2024. [cit. 2024-04-13]. Dostupné z: <https://www.design-reuse.com/articles/54776/i2c-interface-timing-specifications-and-constraints.html>.
25. JUSTDOELECTRONICS. *Understand I2C Communication Protocol* [online]. 2023. [cit. 2024-04-13]. Dostupné z: <https://justdoelectronics.com/understand-i2c-communication-protocol/>.
26. WIKIPEDIE. *RS-485*. 2023. Dostupné také z: <https://cs.wikipedia.org/wiki/RS-485>. Citováno: 18. 4. 2026.
27. WIKIPEDIE. *RS-232*. 2023. Dostupné také z: <https://cs.wikipedia.org/wiki/RS-232>. Citováno: 18. 4. 2026.
28. ARDUINO CC. *Arduino Nano Product Reference*. 2024. Dostupné také z: <https://docs.arduino.cc/language-reference/en/functions/communication/serial/>. Citováno: 18. 4. 2026.
29. KONDIK.CZ. *PWM – Slovník pojmů*. Dostupné také z: <https://www.kondik.cz/slovník-pojmu/pwm/>. Citováno: 18. 4. 2026.
30. WIKIPEDIE. *Pulzně šířková modulace* [online]. 2026. [cit. 2026-04-13]. Dostupné z: https://cs.wikipedia.org/wiki/Pulzn%C4%9B_%C5%A1%C3%AD%C5%99kov%C3%A1_modulace.
31. BOTLAND. *Arduino – řízení DC motorů*. 2020-04. Dostupné také z: <https://botland.cz/blog/arduino-řízení-dc-motoru/>. Citováno: 18. 4. 2026.
32. RAVEO. *Jak funguje servomotor a zpětnovazební řízení?* 2021-02. Dostupné také z: <https://www.raveo.cz/aktualita/jak-funguje-servomotor-a-zpetnovazebni-rizeni/>. Citováno: 18. 4. 2026.
33. MRÓZ, Mateusz. *Servo – co to je a k čemu se používá?* 2023-09. Dostupné také z: <https://botland.cz/blog/servo-co-to-je-a-k-cemu-se-pouziva/>. Citováno: 18. 4. 2026.
34. RASPBERRY PI FOUNDATION. *Using a servo*. 2024. Dostupné také z: <https://projects.raspberrypi.org/en/projects/grandpa-scarer/9>. Citováno: 22. 5. 2024.

35. ARTOO AI. *R2D2 Arm connected to RPi3 with PCA9685*. 2019-06. Dostupné také z: <https://www.artoo.ai/p/r2d2-arm-connected-to-rpi3-with-pca9685/>. Citováno: 18. 4. 2026.
36. LASKAKIT. *TT motor s převodovkou, celokovové převody*. Dostupné také z: <https://www.laskakit.cz/tt-motor-s-prevodovkou-celokovove-prevody/>. Citováno: 18. 4. 2026.
37. LASKAKIT. *H-můstek L298N, dvoumotorový modul*. Dostupné také z: <https://www.laskakit.cz/h-mustek-pro-krokovy-motor-l298n--dualni-motorovy-modul/>. Citováno: 18. 4. 2026.
38. MAGDY, Khaled. *Arduino L298N DC Motor Driver Interfacing Tutorial*. 2023-07. Dostupné také z: <https://deepbluembedded.com/arduino-l298n-dc-motor-driver-interfacing-tutorial/>. Citováno: 18. 4. 2026.
39. RPISHOP.CZ. *Coral USB Accelerator*. Dostupné také z: <https://rpishop.cz/coral/3195-coral-usb-accelerator.html>. Citováno: 18. 4. 2026.
40. INTEL CORPORATION. *Intel Neural Compute Stick 2: High Performance, Low Power for AI Inference*. 2018. Tech. zpr. Intel. Dostupné také z: <https://www.intel.com/content/www/us/en/content-details/842054/intel-neural-compute-stick-2-high-performance-low-power-for-ai-inference-intel-neural.html>. Citováno: 18. 4. 2026.
41. RPISHOP.CZ. *Raspberry Pi AI HAT+ (13 TOPS)*. Dostupné také z: <https://rpishop.cz/554034/raspberry-pi-ai-hat-13-tops/>. Citováno: 18. 4. 2026.
42. CIRKIT DESIGNER. *How to Use 2s BMS: Examples, Pinouts, and Specs*. Dostupné také z: <https://docs.cirkitdesigner.com/component/c338549a-886f-4684-87a3-8dabae539047/2s-bms>. Citováno: 18. 4. 2026.
43. SUPTRONICS TECHNOLOGIES. *X1205 v1.1: Power Management & UPS Shield for Raspberry Pi 5*. Dostupné také z: https://suptronics.com/Raspberrypi/Power_mgmt/x1205-v1.1.html. Citováno: 18. 4. 2026.
44. CHENGDU EBYTE ELECTRONIC TECHNOLOGY CO., LTD. *EWM550-7G9T10SP: UWB Wireless Ranging Positioning Module*. Dostupné také z: <https://www.cdebyte.com/products/EWM550-7G9T10SP/4>. Citováno: 18. 4. 2026.
45. QORVO, INC. *DW1000: Integrated Ultra Wideband (UWB) Transceiver IC*. Dostupné také z: <https://www.qorvo.com/products/p/DW1000>. Citováno: 18. 4. 2026.
46. QORVO, INC. *DW3110: Integrated Ultra Wideband (UWB) Transceiver IC*. Dostupné také z: <https://www.qorvo.com/products/p/DW3110>. Citováno: 18. 4. 2026.

47. QORVO, INC. *DW1001-DEV: Integrated Ultra Wideband (UWB) Transceiver IC*. Dostupné také z: <https://www.qorvo.com/products/p/DWM1001-DEV>. Citováno: 18. 4. 2026.
48. QORVO, INC. *DW3001C: Integrated Ultra Wideband (UWB) Transceiver IC*. Dostupné také z: <https://www.qorvo.com/products/p/DWM3001C>. Citováno: 18. 4. 2026.
49. AI-THINKER. *BU01 Product Specification*. Ai-Thinker Technology Co., Ltd, 2019. Dostupné také z: https://www.elecrow.com/download/product/DPW01041U/bu01_product_specification_en_new_v1.0.pdf. Citováno: 18. 4. 2026.
50. WAVESHARE ELECTRONICS. *LC76G Multi-GNSS Module*. Dostupné také z: https://www.waveshare.com/wiki/LC76G_GPS_Module. Citováno: 19. 4. 2026.
51. GME. *GY-511 Modul tříosého akcelerometru a snímače mag. pole*. Dostupné také z: <https://www.gme.cz/v/1508274/gy-511-modul-trios-eho-akcelerometru-a-snimace-mag-pole>. Citováno: 19. 4. 2026.
52. TROJER, Thomas. *arduino-dw1000*. 2024. Main branch. Dostupné také z: <https://github.com/thotro/arduino-dw1000/tree/master>. Accessed: 1. května 2026.
53. HAILO AI. *Hailo-RPi5-Examples: Collection of examples for Hailo-8L on Raspberry Pi 5*. 2024. Main branch. Dostupné také z: <https://github.com/hailo-ai/hailo-rpi5-examples>. Accessed: 1. května 2026.
54. ULTRALYTICS. *YOLO11 Documentation*. 2024. Dostupné také z: <https://docs.ultralytics.com/models/yolo11/>. Accessed: 2024-05-01.

Obsah příloh

src	
└─ arduino	zdrojové kódy pro Arduino
└─ application.....	zdrojové kódy pro Raspberry
hardware	
└─ 3D	adresář s 3D modely
└─ PCB.....	adresář s daty pro výrobu PCB
text.....	text práce
└─ src	zdrojová forma práce ve formátu $\text{\LaTeX}$
└─ thesis.pdf.....	text práce ve formátu PDF