Středoškolská technika 2022

Setkání a prezentace prací středoškolských studentů na ČVUT

SELF BALANCING ROBOT

Vítek Růžička

VOŠ, SPŠ a JŠ Kutná Hora
Masarykova 197 - Kutná Hora

VYŠŠÍ ODBORNÁ ŠKOLA, STŘEDNÍ PRŮMYSLOVÁ ŠKOLA
A JAZYKOVÁ ŠKOLA S PRÁVEM STÁTNÍ JAZYKOVÉ ZKOUŠKY
KUTNÁ HORA, MASARYKOVA 197

MATURITNÍ PRÁCE

ROK 2022

VÍT RŮŽIČKA

VYŠŠÍ ODBORNÁ ŠKOLA, STŘEDNÍ PRŮMYSLOVÁ ŠKOLA
A JAZYKOVÁ ŠKOLA S PRÁVEM STÁTNÍ JAZYKOVÉ ZKOUŠKY

ZADÁNÍ MATURITNÍ PRÁCE S OBHAJOBOU PŘED ZKUŠEBNÍ KOMISÍ

studijní obor: (26-41-M/01) ELEKTROTECHNIKA

Student: **Vít Růžička**

Třída: **E4A**

Školní rok: **2021 / 2022**

Téma: **22003 Dynbalabot - Dynamický balancující robot**

Konkrétní úkoly řešené v maturitní práci:

Řídící jednotka bude řešena na platformě ESP32. Řídící jednotka bude pohánět jednotku motoru, která se bude starat o otáčení dvou BLDC motorů. Systém bude schopný se sám balancovat a při tom jezdit podle pokynů uživatele, nebo případného strojového vidění. Řízení bude probíhat modelářským ovladačem, případně gamepadem s joystickem. Hlavním konstrukčním prvkem bude 3D tisk s hliníkovými díly.

Zadání schváleno dne: 30. 9. 2021

Termín dokončení práce a předání výstupů vedoucímu práce: 31. 3. 2022

Vedoucí práce: _____

Ing. Pavel Stejskal

Ředitel školy: _____

Ing. Josef Treml

ČESTNÉ PROHLÁŠENÍ

Prohlašuji, že jsem svou práci vypracoval samostatně, použil jsem pouze podklady (literaturu, SW atd.) citované v práci a uvedené v přiloženém seznamu a postup při zpracování práce je v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) v platném znění. Prohlašuji, že tištěná verze a elektronická verze soutěžní práce SOČ jsou shodné. Nemám závažný důvod proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) v platném znění.

V Kutné Hoře dne 31. 3. 2022 podpis:.....

PODĚKOVÁNÍ

S uskutečněním mi pomohl vedoucí práce Ing. Pavel Stejskal, kterému bych tímto rád poděkoval za užitečné rady. Dále patří díky řediteli školy Ing. Josefu Tremlovi za podporu v projektu. V neposlední řadě bych rád poděkoval Tomášovi Janouškovi, který mi pomohl s realizací konstrukce v CADovém programu a Janu Řehákovi, který mi pomohl vymyslet název.

ABSTRAKT
Cílem práce bylo vytvořit model robota, který bude sám balancovat na dvou kolečkách a bude mít proměnnou světlou výšku na základě jízdního profilu.

ABSTRACT
The aim of this project was to create a model of robot, designed to balance itself on two wheels and to dynamically change it´s height based on driving profile.

KLÍČOVÁ SLOVA
balancující robot, esp32, arduino, 3d tisk, CAD, CAN, IMU, BLDC, hoverboard, dynbalabot

KEYWORDS
self balancing robot, 3d printing, two-wheeled robot

OBSAH

ÚVOD

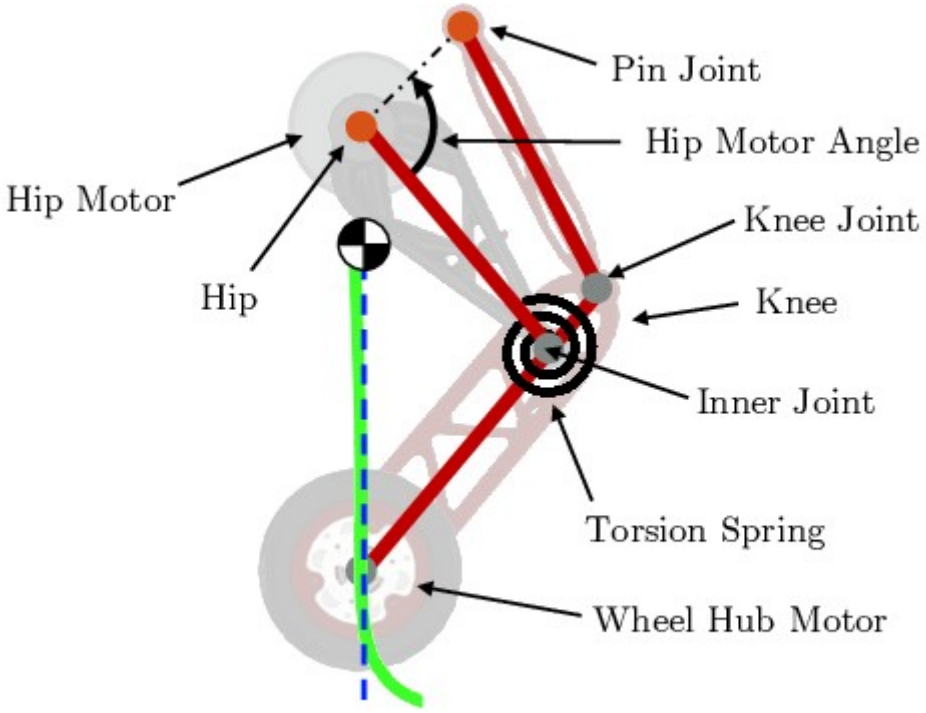
Na konci třetího ročníku jsem stál před volbou. Buď jsem mohl využít jeden z mnoha projektů, který jsem ve škole za mého působení realizoval (namátkou: elektrická bugyna, digitronové hodiny, herní automat, FPV dron, měření spotřeby v solárním okruhu, robot na Siemens Embedded Academy nebo loňská maturitní práce Pavla Hrstky - Model automatického skladu, na které jsem s ním spolupracoval) a mohl jsem k němu sepsat dokumentaci a mít hotovo. Já jsem si ale řekl, že to by bylo moc jednoduché. Zároveň jsem měl chuť vyzkoušet si práci s dynamickým systémem. Chtěl jsem něco, co by ukázalo rozsah mých schopností, které jsem získal na škole, a převedlo je do fyzické formy, za kterou se nebudu muset stydět před maturitní komisí. Během tohoto rozhodování jsem si také vzpomněl na diplomovou práci Adama Kollarčíka, která byla mimo Švýcarský projekt Ascento hlavní inspirací tohoto projektu. A tak se zrodil DYNBALABOT = Dynamický balancující robot. Myslím že název trefně odráží funkci.

1. TEORETICKÝ ROZBOR

1.1. Dynamika systému

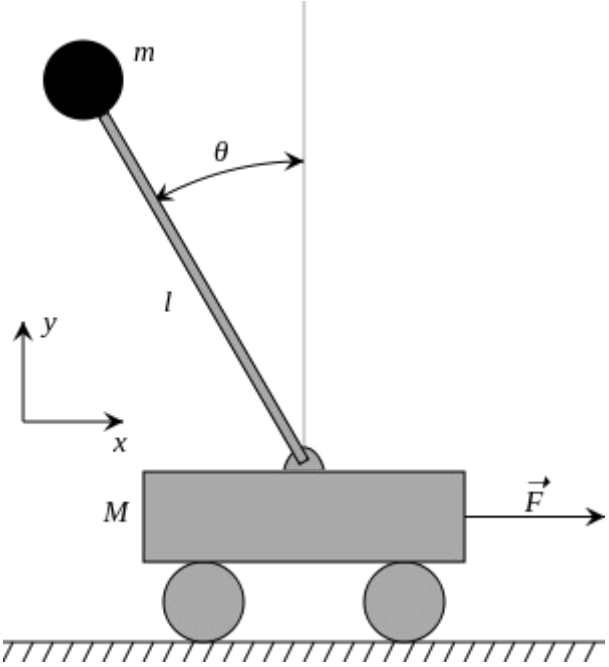
1.1.1. Princip obráceného kyvadla

Má konstrukce robota, ač se to možná nezdá, se vcelku přibližuje systému obráceného kyvadla. Cílem elektroniky obsluhující kola na konci nohou, je balancovat váhu nad nimi tak, aby kolmice vedená od středů kol protínala pomyslný hmotný bod robota,viz následující obrázek:



Obrázek č. 1 Kontrukce robota Ascento

Pro snazší stabilizaci v problému obráceného kyvadla je důležité rozložení váhy. Viz obrázek 2:



Obr. č. 2 Princip soustavy obráceného kyvadla

Je uvažováno, že veškerá hmota kyvadla je uložena v hmotném bodu m. Váha tyče l se zanedbává. Hmota M vozíku, respektive hmotného bodu, který akceleruje a deceleruje s cílem udržení hmotného bodu m kolmo na vozík M tudíž zachování ideálně nulové odchylky θ, by měla být co možná nejnižší, pro co nejnižší nároky na energii a trakci při akceleraci a deceleraci. Pro optimální funkci by měla být hmota m >> M. [1]

1.2. Způsob řízení

1.2.1. PID regulace

Takzvaný PID algoritmus [2], jehož název přesně popisuje jeho jednotlivé složky je jedním ze základních spojitých druhý regulace. Jde o soustavu, která má danou odezvu na jednotkový skok, která lze vyjádřit tzv, přenosem, který efektivně chování soustavy popisuje a lze následně integrovat do simulačního prostředí pro úspěšný výpočet konstant regulátoru. Konstanty, o kterých jsem se již okrajově zmínil jsou celkem 3. Proporční, Integrační a Derivační. Jejich jednotlivý vliv na změnu reakce systému je již popsaný v odborné literatuře a nebude součástí této práce. Přesto bych rád velice zjednodušeně napsal, k čemu tyto konstanty slouží a jak lze fyzicky pozorovat vliv jejich změny na chování robota.

Proporční složka je nejspíše tou nejjednodušší, jelikož její hlavní cíl je pouze konstantou P násobit chybu systému (rozdíl požadované veličiny – v našem případě úhel). V zásadě se dá říct, že udává jak prudce bude systém reagovat na zvyšující se chybu.
Derivační složka, je v pořadí ladění jednotlivých konstant po té proporční na druhém místě, proto jí také zařazuji na toto místo. Jejím úkolem je tlumit oscilace, které způsobí složka předchozí a naopak má tendenci robota “zlenivět” kolem setpointu(v našem případě úhlu, ve kterém robot setrvává v absolutní rovnováze).
Integrační složka se v podstatě stará o dorovnání chyby, která vznikne po předchozích dvou složkách regulátoru. Systém již dokáže zareagovat prudce na změnu (P), dokáže “dobrzdit” před požadovaným úhlem ale ani jeden z předchozích regulátoru není schopný citlivě dorovnat chybu na 0, tudíž robota na požadovaný úhel. A zde se hodí integrační složka, která akumuluje chybu za jednotku času nebo počítá integrál této křivky a podle toho přispívá do výstupu regulátoru.

Když jsem poprvé slyšel odborný výklad o těchto regulátorech, nedokázal jsem ho pochopit. K poznání funkce mi pomohlo až když jsem vidět kód, který obsluhuje program pro tento regulátor v jazyce C++, díky kterému jsem pojal jak celá tato problematika funguje. A naprosto nejlepší bylo, když jsem mohl moje PCB přidělat na robota a fyzicky vyzkoušet, co jednotlivé reguátory ovlivňují. Pro ukázku, zde je výstřižek kódu, který se stará o provádění algoritmu v mé práci:

```
error = (soucasnyUhel - offsetUhel) - cilovyUhel ;
soucetErr = soucetErr + error;

vystup = Kp*(error) + Ki*(soucetErr)*runTime + Kd*(soucasnyUhel-predUhel)/runTime;
predUhel = soucasnyUhel;
```

Obr. 3 PID smyčka

Názvy proměnných jsou vcelku sebe-vysvětlující. Tato část kódu se provádí přibližně ve frekvenci 1kHz.

Během mého testování jsem přišel na několik různých aspektů, které ovlivňují kvalitu regulace:
Pro úspěšnou stabilizaci robota je třeba zajistit několik různých věcí. Mezi hlavní viníky, kteří se podílejí na stabilitě systému patří:

- konstantní, lépe vysoká rychlost běhu řídící smyčky PID
- mechanická stabilita konstrukce, minimální vůle
- minimum šumu v gyroskopových datech
- co nejmenší latence
- konstantní a dostatečná trakce koleček s povrchem
- vhodné rozložení váhy v systému (viz kapitola 1.1.1)

Chtěl bych některé z těchto bodů více rozvést a také uvést, co mě vedlo k jejich prioritizaci.

1.2.1.1. Rychlost běhu PID smyčky

Konstantní frekvence provádění smyčky je, jak již bylo naznačeno, naprosto kritická. S tímto konstantním časem se totiž počítá v derivační složce PID algoritmu a je tím ovlivněn výpočet a kvalita stabilizace. K dosažení přesného časování je použit vnitřní timer u čipu ESP32. Později je podrobněji rozepsán v kapitole 3.1.2

1.2.1.2. Minimalizace šumu

Je vskutku důležitou součástí, v podstatě tou nejdůležitější. Když totiž robot přesně neví, že hodnota, kterou považuje skutečnou je jiná od reálné, nemůže správně vypočítat opravu této chyby a tudíž dochází k chybě měření (stabilizace). Z tohoto důvodu je u IMU, které používám, použit vnitřní procesor, který se jmenuje DMP a má za úkol odebrat zátěž z hlavního MCU. Mnohem důležitější je ale zpracování dat, které využívá přímý přístup do IMU a tím i vysokou rychlost zpracování dat. Senzor totiž kombinuje data, vyhlazuje je a navrací pouze odfiltrované hodnoty úhlů a akcelerací, které jsou dále zpracovány v hlavní smyčce na ESP32.
Zvýšené množství šumu je způsobeno jak rušením od zdrojů a spínaných měničů tak uložením desky až k EM rušení od BLDC motorů v kolech.

2. HARDWARE A SOFTWARE

2.1. Hardware

2.1.1. Elektronika - výběr čipu do základní desky

Při výběru MCU, které je mozkem celé práce jsem stál před dilematem. Vzhledem k datu realizace (přelom roku 2021/22) samotné práce a k tomu, že ve světě v té době byla probíhala krize na Taiwanu [3], která ovlivnila dodávky čipů do celého světa jsem musel volit čipy pečlivě. V potaz připadaly hlavně tyto volby:

- AVR
- STM32
- RP2040
- ESP8266 / ESP32

2.1.1.1. AVR

Taktéž známé jako “Arduino”. Jedná se o čipy vyráběné firmou Microchip, se kterými jsem měl již v minulosti při realizaci jiných projektů jisté zkušenosti. Nejběžnějším je nejspíše Atmega328p, obsažena ve všech nejoblíbenějších vývojových arduino deskách. Ta má frekvenci 16Mhz pomocí externího krystalu, 8-bitové jádro, 32kb FLASH a 2kb RAM. Je časem vyzkoušená, vyladění, spolehlivá a dobře zdokumentovaná o čemž svědčí 294 stránkový datasheet. Pro samotnou stabilizaci by snad vyhovovala, ale jak jsem vymýšlel další funkce, rychle bych narazil na limity v podobě paměti a výkonu. Také cena v důsledku krize vyletěla nad 10 dolarů. [4]

2.1.1.2. STM32

Procesory využívané hlavně v komerčních produktech jako základní průmyslově používané, spolehlivé a dobře zdokumentované. Beží oproti AVR na 3v logice, která je dnes již standardem. Nabízí také více paměti, ram, vyšší frekvenci jádra, 32bit architekturu a periferie jako USB a CAN transceiver. Konkrétně je řeč o STM32F103C8T8, což je procesor použitý například v populární vývojové desce označené jako BluePill. Nicméně dostupnost byla v době výběru na 0 u všech dodavatelů součástek a cena, v případě, že se objevil, byla astronomická. [5]

2.1.1.3. RP2040

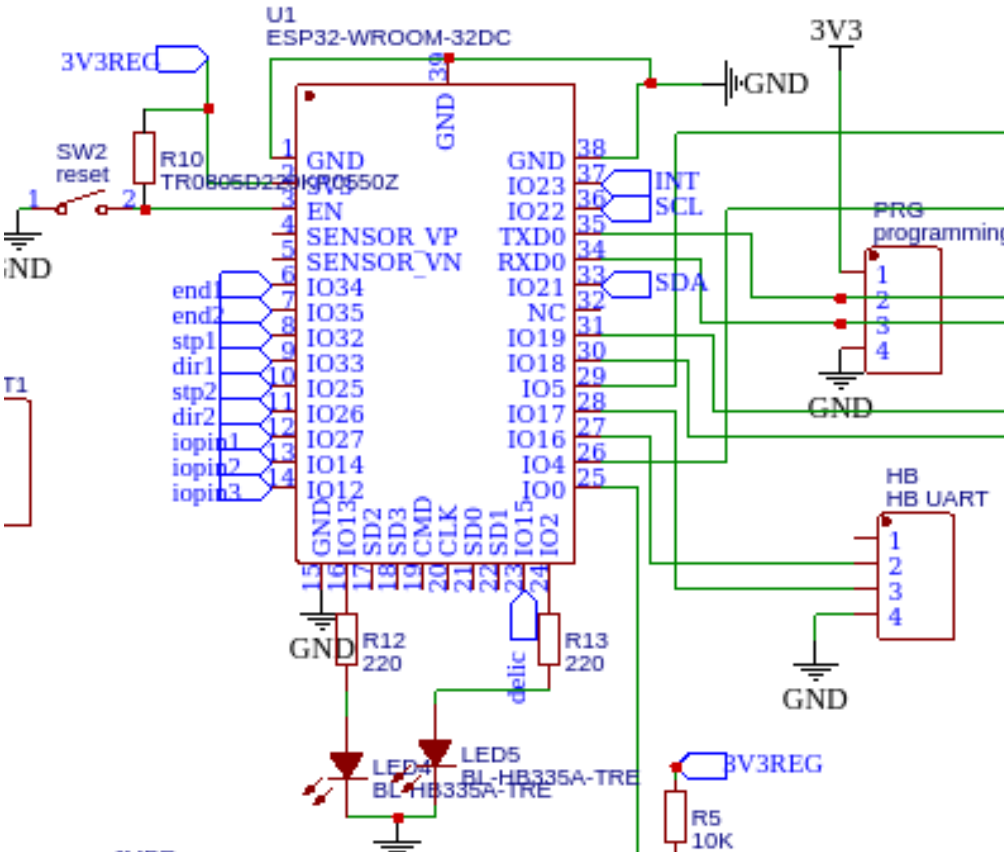
Velice nový hráč na trhu, který je navrhnutý a vyrobený organizací Raspberry Pi foundation jako jejich vstupenka na trh mikrokontrolérů, který doposud, na rozdíl od trhu s SBC, ještě nedobyla. Parametry jsou již velice zajímavé. Nabízí totiž hned dvoujádrový procesor s rychlostí 133MHz, 264Kb RAM a až 16MB FLASH. K tomu podporu usb a hardwarově akcelerovaných periferií. A nejlepší věc je, že cena je pouze 1 dolar, v případě že objednáváte přímo od nadace dokonce 0.7 dolaru, což vzhledem ke kontrastu s nedostatkem čipů je jako zásah shůry. [6]

2.1.1.4. ESP

Čipy od firmy Espressif jsou nicméně vítězi v této kategorii. ESP32 nabízí dvoujádrový procesor s frekvencí až 240MHz, 520kb RAM, 450kb ROM ale hlavně WIFI a Bluetooth. Spoustu funkcí na každém pinu, velká počet sběrnic a hardwarový čip pro kryptografii jsou už jen třešničkou na dortu. Zvolil jsem tedy tento čip a nelituji toho, protože jsem projekt mohl doplnit o velké množství nadstandardních funkcí. Cena, ne příliš ovlivněna krizí, zůstala na příjemných 3 dolarech (v CZ samozřejmě více), ale hlavně je všude dostupný. [6]

2.1.2. Motherboard

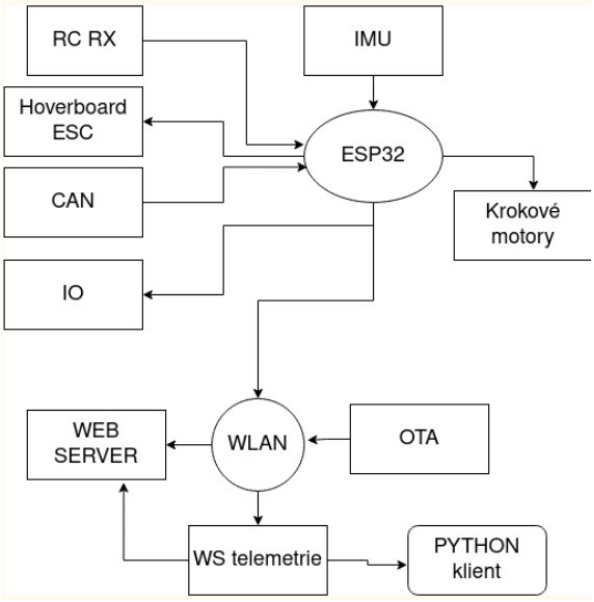
Po výběru MCU přišla řada na výběr ostatních komponent. Jako IMU jsem zvolil populární MPU6050, který již prověřil čas a také díky jeho DMP procesoru, který dovolí slévání dat. Pro komunikaci přes USB jsem integroval CH340E, jehož volbu bych z důvodu velikosti a absence resetovacího rozhraní v budoucnu spíše přehodnotil. Pro komunikaci s ostatními součástmi robota jsem využil CAN transceiveru v ESP a připojil CAN controller čip. Toto rozhraní mělo sloužit jako hlavní způsob pro komunikaci s dalším projektem dynbalaStepper, o kterém bude ještě v práci zmínka. V neposlední řadě jsem také přidal adresovanou led pro indikaci a dva řidiče krokových motorů.



Obr. 4 Výřez ze schématu

Základní deska si, jak je patrné (Obrázek 5, 6 v příloze), prošla jednou iterací. Důvodem bylo zmenšení rozměrů, přidání USB, opravení několika cest a zjednodušení celkového schématu. Také došlo ke zmenšení rozměrů z původních 122x81mm na 96x76mm. Zde jsou všechny funkce, které po hardwarové stránce plní deska:

- Reguluje napětí z 24V baterie pro 3.3V obvody
- připojuje k ESP periferie:
    - hoverboard motherboard
    - CAN
    - krokové motory
    - koncové snímače
    - 3x IO piny
    - adresovanou LED WS2812B
    - IMU
    - SBUS přijímač (pro příjem signálu z modelářského ovladače)
    - USB
    - 2x signální LED



Obr. 6.5 – Diagram sw funkce

2.1.3. DynbalaStepper

Myslím že by bylo vhodné také věnovat malou kapitolu této maturitní práci v práci.

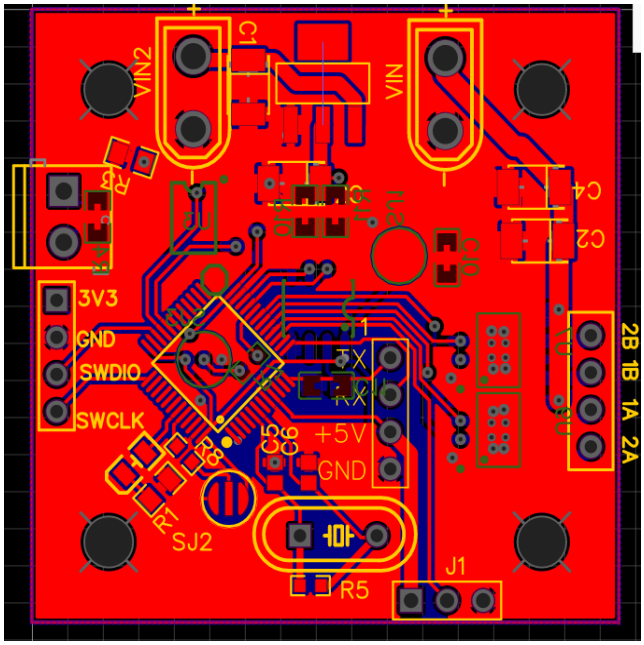
2.1.3.1. Cíl

Jelikož moji předchůdci na ČVUT [8] a ve Švýcarském Zurichu [9] používají proměnnou délku nohy pro zlepšení jízdních vlastností, chtěl jsem tuto funkcionalitu zachovat i u svého robota. Ascento využívá ANYdrive series-elastic actuators, což jsou velice nákladné zpětně řiditelné aktuátory. Na ČVUT šli levnější cestou a tak použili BLDC motory určené pro velké agro-drony a k tomu si vyrobili BLDC drivery vycházející z návrhu Bena Katze. Ty ovládali pomocí CANbusu. Jejich ambicí bylo ale stejně jako u Švýcarského teamu robota donutit ke skoku, pomocí vyvinutí velikého točivého momentu v kolenních kloubech. Mou prioritou to ale nebylo, a tak jsem zvolil výrazně levnější řízení pomocí krokových motorů NEMA 17 se kterými mám zkušenost z 3D tiskáren a z maturitní práce Pavla Hrstky [10], kde jsem je použil k hýbání s celým modelem skladníka. Motory jsem doplnil o magnetické enkodéry a cílem bylo, aby byly zpětně řiditelné a tudíž se chovaly jako pružiny.

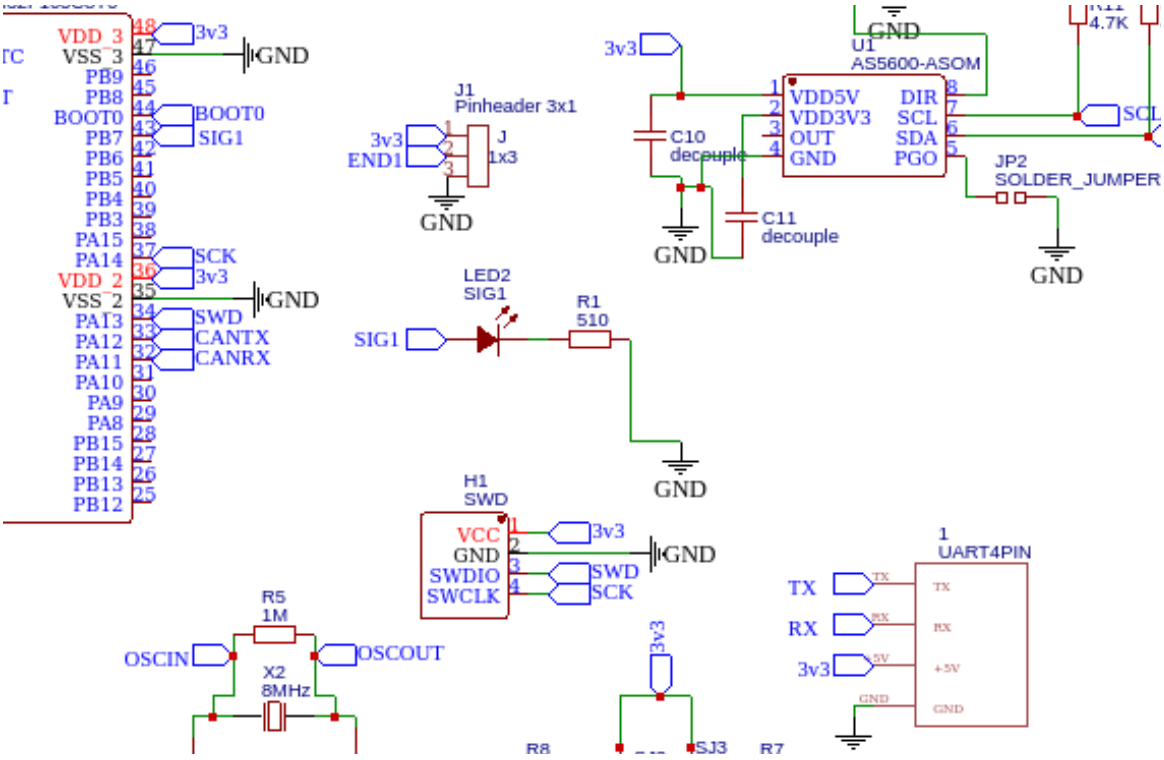
2.1.3.2. Realizace

Výše popsané je opravdu možné pomocí řízení se zpětnou vazbou přes FOC algoritmus. Vím to, protože firma Bigtreetech vyrábí přesně desky s touto funkcionalitou. Jejich řešení je ale poměrně finančně náročné a tak jsem se rozhodl pro vlastní návrh.

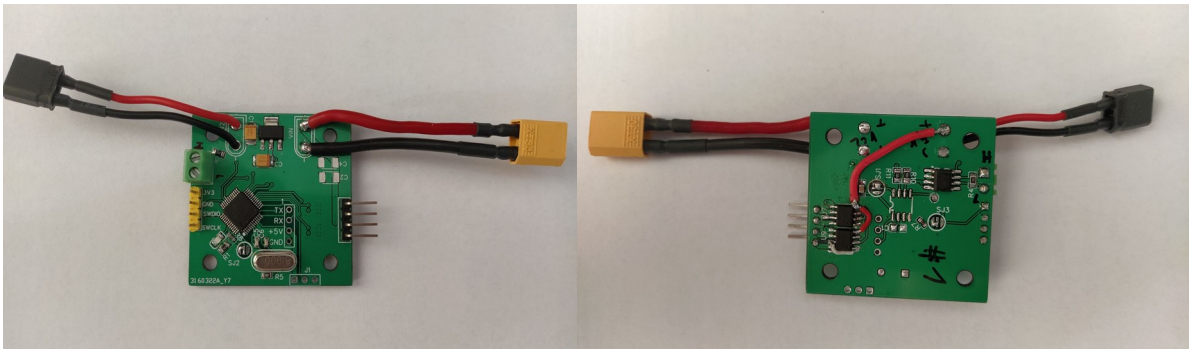
Začal jsem návrhem plošného spoje, který obsahoval procesor stm32, jelikož jsem si chtěl vyzkoušet práci s ním. Dále 2x A4950 H-můstek pro spínání fází krokového motoru (který je v podstatě bldc motor s jiným vinutím, větším počtem pólových dvojic (50 oproti 14) a menší vzduchovou mezerou mezi magnety a vinutím). Na desce byl také magnetický enkodér (as5600) komunikující přes i2c s samozřejmě CAN bus, což byla hlavní věc, jejíž absence mi vadila na komerčním řešení.



Obr. 7 PCB v měřítku 2:1- rozvržení DynbalaStepperu



Obr. 8 Výřez ze schématu zapojení DynbalaStepperu



Obr. 9 a 9.5 Realizovaný plošný spoj

2.1.3.3. Vývoj

Pro softwarovou část jsem se rozhodl využít již existující projekt SimpleFOC. Jedná se o open source knihovnu podporující celou řadu H můstků a magnetických enkodérů. Program jsem začal psát ve VScode s doplňkem platformio, který mi dovolil použít vyšší funkce z frameworku arduino, výrazně zjednodušující množství programu a naopak zvětšující možnosti co se týče použitelných knihoven. Zprovoznil jsem procesor i komunikaci přes CAN a to mě do jisté míry motivovalo.

2.1.3.4. Závěr

Problém ale nastal, když jsem se snažil rozhýbat motor. I přes nesčetné pokusy a testování magnetický enkodér neodpovídal a motor se v režimu otevřené smyčky točil pouze s minimem proudu, a tak docházelo k zastavení motoru dvěma prsty. Nepodařilo se mi žádným způsobem zvýšit proud a tím pádem dokončit realizaci tohoto subprojektu. Při testování jsem rovněž zničil 3 h-můstky a vzhledem k tomu, že nebyly v ČR dostupné jsem vývoj pozastavil. Také mě tlačil čas a tak jsem vsadil na mechanické řešení tlumení nárazů. Celý projekt by snadno mohl fungovat, ale jednalo se pouze o dílčí část a tak jsem se spíše soustředil na celý projekt.

2.1.4. Elektronika - řízení motorů v kolech

2.1.4.1. Prolog

Řízení motorů se obvykle liší na základě jaký typ motoru použijete.
U kartáčkových DC motorů stačí jeden tranzistor, pokud se řídí pouze v jednom směru, následně H-můstek, pokud ho chcete řídit i více směry. Bez zpětné vazby tudíž otáčky motoru závisí hlavně na napětí, které je na něj přivedeno a na zátěži, s kterou točí. Toto napětí se následně dá ještě regulovat pomocí poměru střídy signálu, tomuto postupu se říká PWM (Pulsně šířková modulace).
U například synchronního střídavého motoru [11] dochází při řízení rotoru pomocí rotačního magnetického pole. Rychlost rotoru přitom odpovídá rychlosti tohoto pole (proto název synchronní). K řízení větších motorů se používá frekvenční měnič, který obsahuje IGBT tranzistory.
BLDC motory jsou (BrushLess DC motor), jak název implikuje, stejnosměrné motory, k jejichž buzení není třeba komutátoru a tak odpadá starost v podobě opotřebení, prachu a jiskření. Vyznačují se vysokým výkonem, rychlostí, účinností, ale nižším držícím momentem.
A právě rychlost a točivý moment byl důvod proč jsem se rozhodl využít tento typ motorů i u mého projektu.

2.1.4.2. Sehnání motorů

Nevýhodou BLDC motorů je také cena. Oproti DC je podstatně vyšší a tudíž nejsou tolik rozšířené. Je ale několik odvětví ve spotřebním průmyslu kde v posledních letech zažívají boom a to je osobní transport a hračky, ideálně kombinovaně.

Obr. 10 Koloběžka Xiaomi

Právě elektrické koloběžky dostaly tyto motory mezi masy lidí.

A nebyly to jen koloběžky ale i takzvané hoverboardy - s odstrašujícím českým překladem KOLONOŽKY. A vypadaly takto:

Obr. 11 Hoverboard

Nikoliv tak, jak nám sliboval snímek Návrat do budoucnosti:

Obr. 12 Návrat do budoucnosti - hoverboard

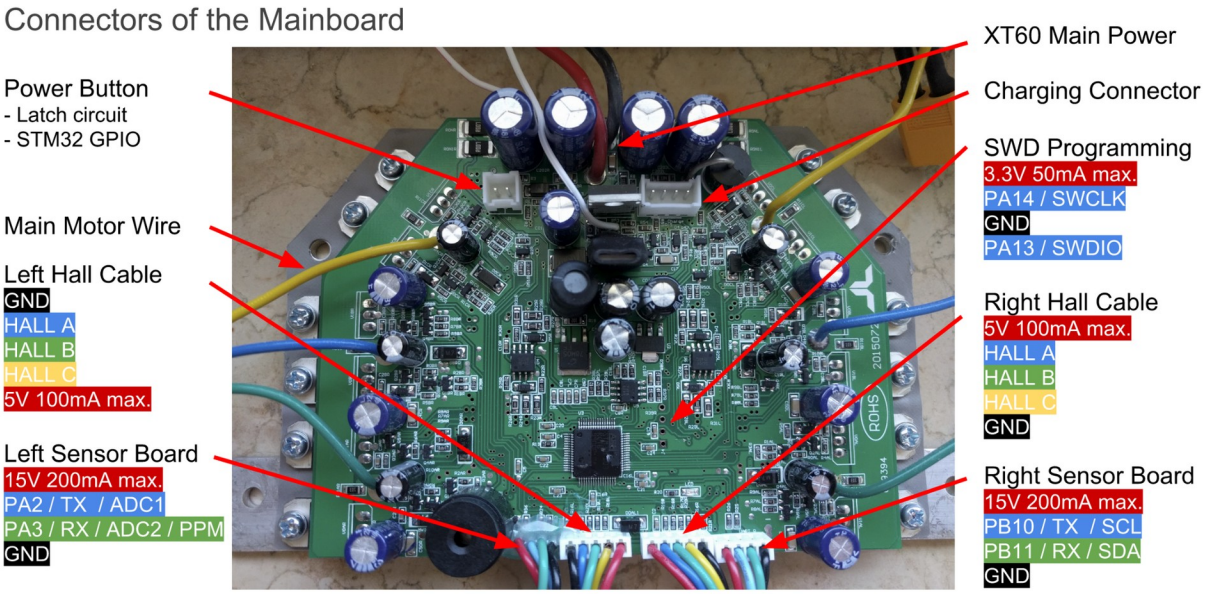
Hoverboard je mimo svou sebevražednou konstrukci poměrně zajímavým zařízením z hlediska elektrického. Zároveň je poměrně jednoduchý (motory přímo v kolech - inhub motor). Skládá se z několika hlavních součástí: [12]

- 2x 6”/8”/10” BLDC 36v 300W motor s 3x hallovými senzory v každém motoru
- základní deska s 12 FETy, STM32, měřením proudu
- postranní desky s IMU. STM32, 433 přijímačem a výstupy na LED světla, měřením přítomnosti na hoverboardu
- 36V 4.4Ah Li-ion baterie s BMS, které ale většinou nefunguje

Tyto hračky dorazily na trh na Vánoce roku 2015 a ihned se jich po bazarech nakupilo veliké množství. To dává výhodu lidem jako já, protože často “nefunkční” hoverboardy mají jen podbitou baterii a buď je vyplo BMS a baterie odmítá dobíjet nebo odešel některý článek a tím s sebou vzal do lithiového nebe i zbytek článků.

2.1.4.3. Využití stávajícího hardwaru

Jak jsem již zmiňoval, k řízení těchto motorů je třeba frekvenční měnič. Na hobby úrovni jsou zpravidla používány takzvané “ESC” (Electronic speed controller). Skládají se z MCU s nějakým vstupem, ať už PWM nebo digitálním seriovým jako Dshot, používaný v závodních FPV dronů, díky své rychlosti. Na základě vstupu MCU spíná zpravidla 3 half-bridge pro 3 fáze. ESC většinou nemají ani vstup na senzor (jsou i vyjímky jako Odrive nebo VESC), který by udával aktuální polohu motoru, místo toho se snaží odhadnout pozici rotoru ze zpětného elektrického impulzu generovaného motorem (Back EMF). Tento způsob je dostatečný pro drony a řízení ve vysokých otáčkách, ale ne příliš spolehlivý a přesný. Proto jsou často tyto motory dovybavovány enkodéry - nejčastěji na koloběžkách a hoverboardech.
Součástí hoverboardu je již avizovaná základní deska, která obsahuje vše potřebné k řízení motorů, jen se do ní nelze nabourat a ovládat motory podle potřeby. Tedy nebylo to možné, dokud v roce 2017 nepřišel Niklas Fauth[13], který úspěšně pomocí reverzního inženýrství dokázal zjistit zapojení desky a napsal pro ni svůj firmware, který otevírá možnosti pro použití v robotických aplikacích. Jeho firmware stačí nakonfigurovat a flashnout pomocí STlinku (programátor pro STM čipy). Firmware obsahuje mnoho variant používání z nichž jsem například jednu použil pro řízení motorů v elektrické bugyně. S firmwarem se dá komunikovat mnoha způsoby, pro mě nejrychlejší přišel vhod režim UARTU, který stačí spojit s příslušným konektorem vyvedeném na mé základní desce, nastavit komunikaci, která obsahuje i CRC a zpětnou telemetrii otáček a napájení z hoverboard desky a “bylo hotovo”.



Obr. 13 - Github - Zapojení desky z hoverboardu

2.2. Software

2.2.1. Prostředí

S programováním jsem před lety začínal v Javascriptu, postupně jsem objevil Arduino a pak Linux, takže jsem si prošel několika způsoby programování a hlavně několika jazyky. Pro arduino a embedded je běžné C/C++ a Micropython. Já jsem se rozhodl pro realizaci v C/C++ s Arduino frameworkem, který mi nabídne velké množství knihoven a vyšší výkon. Po různých editorech jsem nainstaloval na svou linuxovou distribuci Visual Studio Code, který mi dosud vyhovoval asi nejvíce. V něm je skvělý doplněk jménem Platformio IDE, který pod sebe integruje vývoj na mnoho embedded platformách, mezi které samozřejmě patří AVR. STM, 2040 i mnou použité ESP. Program jsem členil do souborů a snažil se využívat objektově orientované programování kvůli vyšší přehlednosti.

2.2.2. Funkce

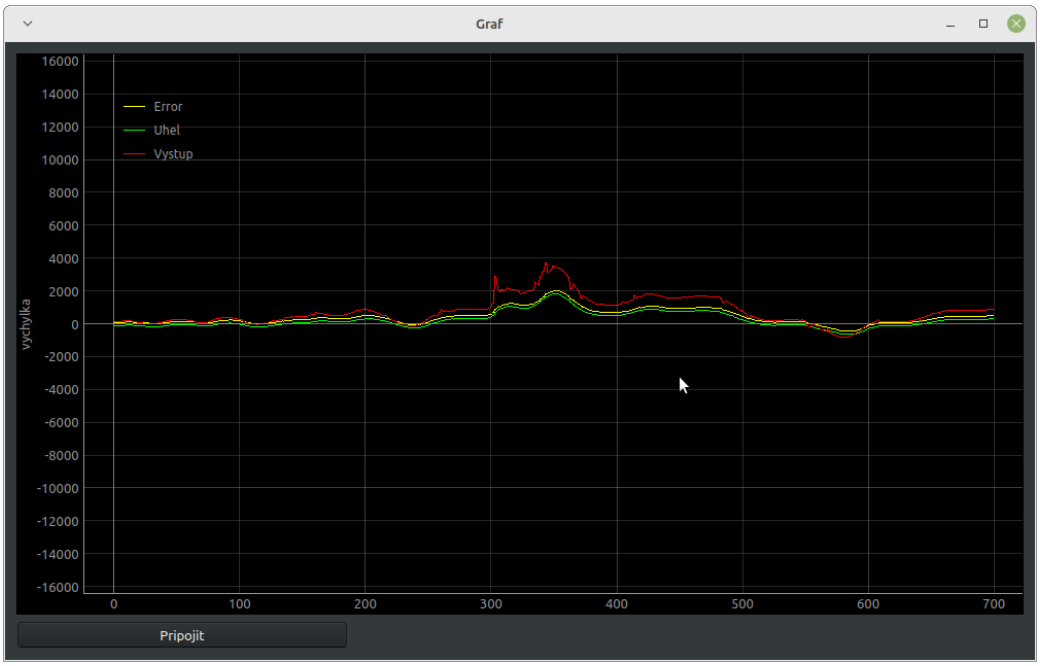
Po softwarové stránce jsem se snažil dosáhnout maximálního zjednodušení, jelikož se nepovažuji za dobrého programátora. I přesto se mi podařilo vytvořit tento souhrn vlastností:

- I2C komunikace IMU s f > 1kHz
- SBUS komunikace s modelářským přijímačem a indikací odpojení ovladače (failsafe)
- CAN komunikace
- hoverboard UART komunikace
- PID smyčka s f >= 1kHz
- WS2812 řízení adresovaných led
- Řízení krokových motorů
- USB sériová komunikace
- WIFI
    - Websocket data v prohlížeči (5Hz) (obr. 16)
    - Nastavování PID konstant přes webové prostředí (obr. 16)
    - Real time telemetrie do pythonu s vykreslováním reakce PID regulátoru(50Hz) (obr. 15)
    - OTA (nahrávání kódu do esp přes wifi)

Looptime se i s funkční wifi a připojenými klienty pohybuje okolo 400us. Celý program, mimo knihoven, tj. část napsaná mnou obsahuje cca. 1370 řádků kódu. Je to část, která mi na projektu zabrala nejvíce času.

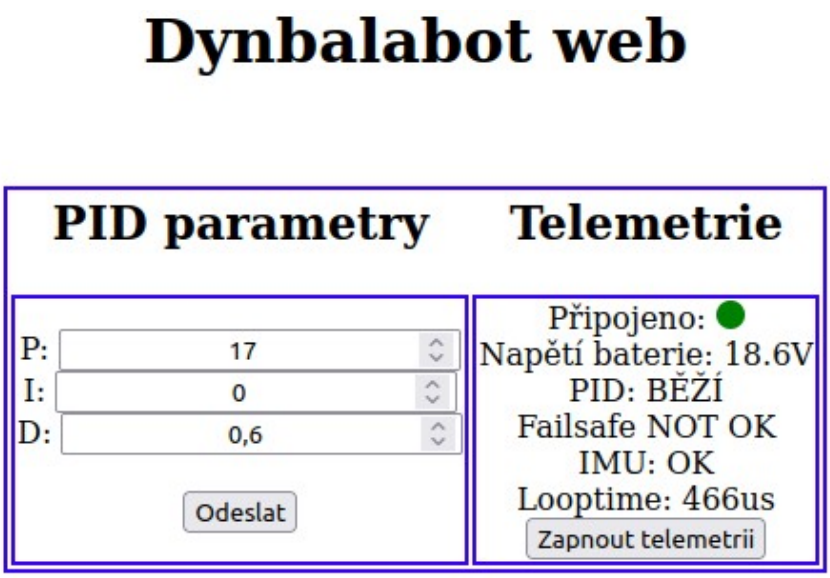
```
void odesliTelemetrii(int looptime){
  if(telemetrie){
    const uint8_t size = JSON_OBJECT_SIZE(6); //odpovídá počtu prvků
    StaticJsonDocument<size> json;

    json["nap"] = String(VCC);
    json["pid"] =  channels[4] < 500 ? "BĚŽÍ" : "NEBĚŽÍ";
    json["fs"] =  failSafe ? "OK" : "NOT OK";
    json["mpuOK"] =  devStatus == 0 ? "OK" : "NOT OK";
    json["lt"] = looptime;
    char data[150];
    size_t len = serializeJson(json, data); //prevedeni dat na JSON

    ws.textAll(data, len); // odeslani dat vsem klientums
  }
}
```

Obr. 14 Příklad kódu - odesílání telemetrie do webového rozhraní



Obr. 15 - Python skript - Live data z robota

U Obr. 15 není spuštěna PID smyčka, tudíž systém nereaguje na změny a nesnaží se proti nim kompenzovat.



Obr. 16 - Webové rozhraní

Webové rozhraní k nastavování PID parametrů a monitorování základních parametrů robota. Kód je součástí programu, který běží v robotovi na ESP32. Do webového prostředí se data posílají pomocí websocketů, což je relativně nová technologie (od r. 2011) používaná ve webových aplikacích. Jedná se o náhradu za technologii Ajax, která se používala do té doby. Význam obou technologií je v dynamickém obnovování obsahu stránky. WS jsou značně rychlejší, jelikož množství vyměněných dat mezi zařízeními je menší a také dochází k menšímu počtu dotazů a odpovědí. Rychlost WS se tak může pohybovat až ve stovkách Hz a mezi jejich nejčastější využití patří například našeptávání v prohlížečích, real time zobrazení sbíraných dat, jako jsou grafy a statistiky.
V prohlížeči je kód napsaný v HTML, který udává stránce strukturu, část kódu v CSS, který dává stránce poněkud vágní ale funkční vzhled (použití bootstrapu by bylo jistě dobré, ale v embedded aplikaci na to není paměť) a konečně hlavní část v javascriptu, která se stará o výměnu dat mezi C programem v robotu, prohlížečem a desktopovou aplikací v pythonu pro zobrazování grafů. Část javascript programu přikládám níže:

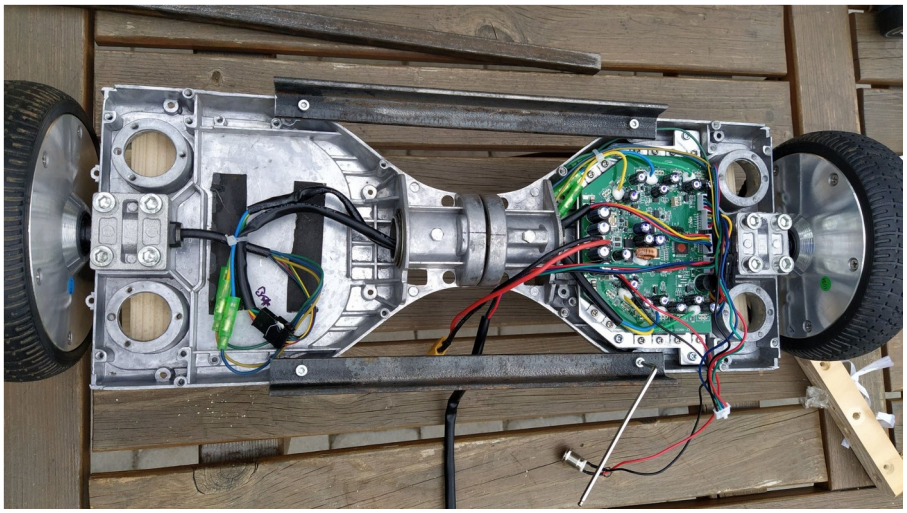
```
function onMessage(evt) {

    console.log("Received: " + evt.data); //evt.data

    let data = JSON.parse(evt.data); //parsování dat z jsonu z esp
    if(Object.keys(data).length == 3 && data.hasOwnProperty("kP") ) {
        parametrP.value = Number(data.kP);
        parametrI.value = Number(data.kI);
        parametrD.value = Number(data.kD) ;

    }
    else if(Object.keys(data).length == 5){
        napeti.innerHTML = Number(data.nap).toFixed(1) + "V";
        pid.innerHTML =           data.pid;
        failsafe.innerHTML = data.fs;
        mpu.innerHTML  =     data.mpuOK;
        looptime.innerHTML = Number(data.lt) + "us";
    }
}
```

Obr 17 - Javascript kód pro obsluhu WS zpráv

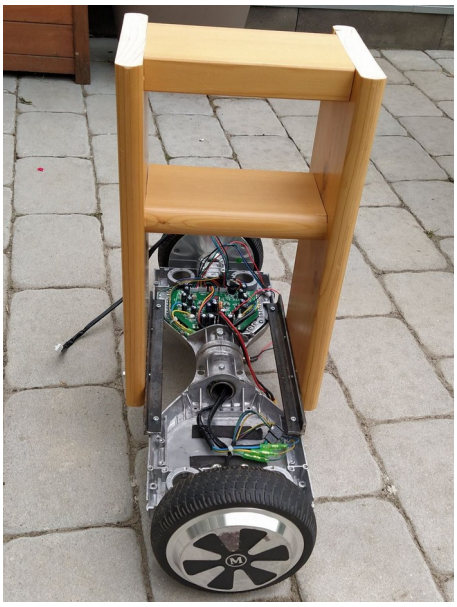
3. PROTOTYP

3.1. První prototyp

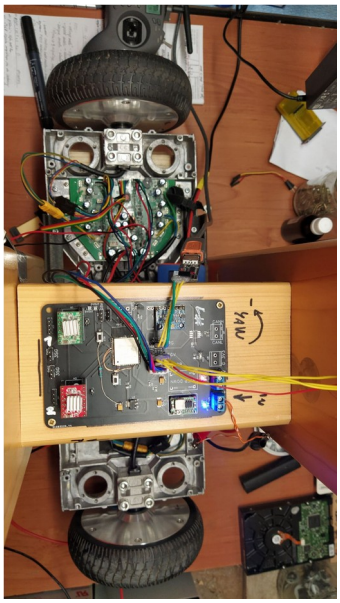
Nejdříve jsem realizoval celého robota na základním podvozku. Použil jsem již existující výlisek z hliníku, který slouží jako hlavní kostra hoverboardu. Na tento výlisek jsem vytvořil dřevěnou konstrukci, která sloužila ke zvýšení těžiště robota. První prototyp měl sloužit pouze k odladění PID regulátoru a zjištění, jestli je celý systém dost rychlý na to, aby se dokázal stabilizovat.



Obr. 18 - Původní hoverboard



Obr. 19 Spodek z hoverboardu s nástavbou

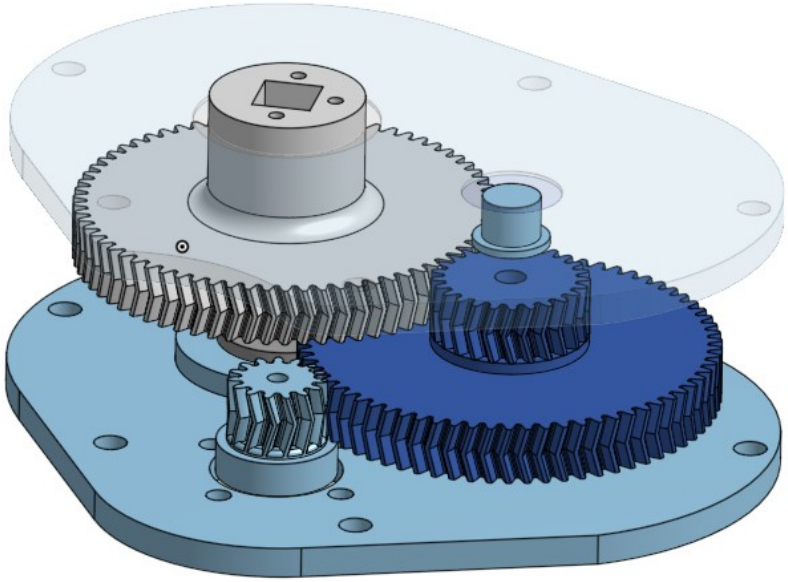


Obr. 20 Hoverboard s mojí řídící deskou

viz [Video 1] je vidět reakce prototypu na vysokou složku P v PID algoritmu.

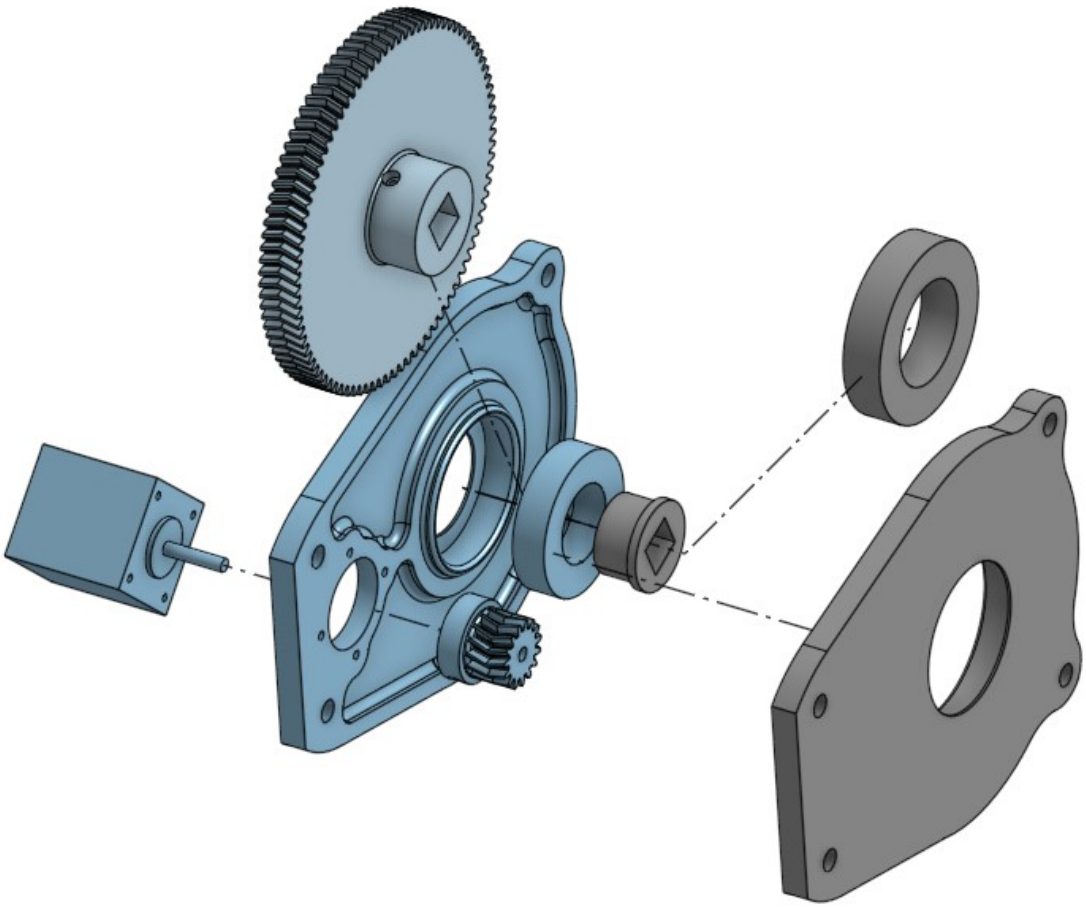
3.2. Druhý prototyp

Po realizaci prvního prvního prototypu jsem si byl jistý, že koncepčně robot bude fungovat a že musím upřednostnit rozložení váhy, které odpovídá obrácenému kyvadlu. Rozhodl jsem se také přidat vylepšení v podobě proměnné délky nohou robota, které mi umožní měnit světlou výšku a tím i umožnit průjezd pod některými překážkami.
Celý prototyp začal náčrtem na papíře, ze kterého jsem nadále vycházel, a podle kterého jsem začal modelovat první části robota. Při výsledné sestavě v cadu, jsem nicméně objevil veliké nedokonalosti v podobě nemožnosti vyrobit jednotlivé díly a tak jsem optimalizoval jednotlivé díly tak, aby šly vytisknout v jednom kuse na 3d tiskárně Ender 5. A tak vznikla první generace převodovky se třemi koly v poměru 15:25:70 zubů.



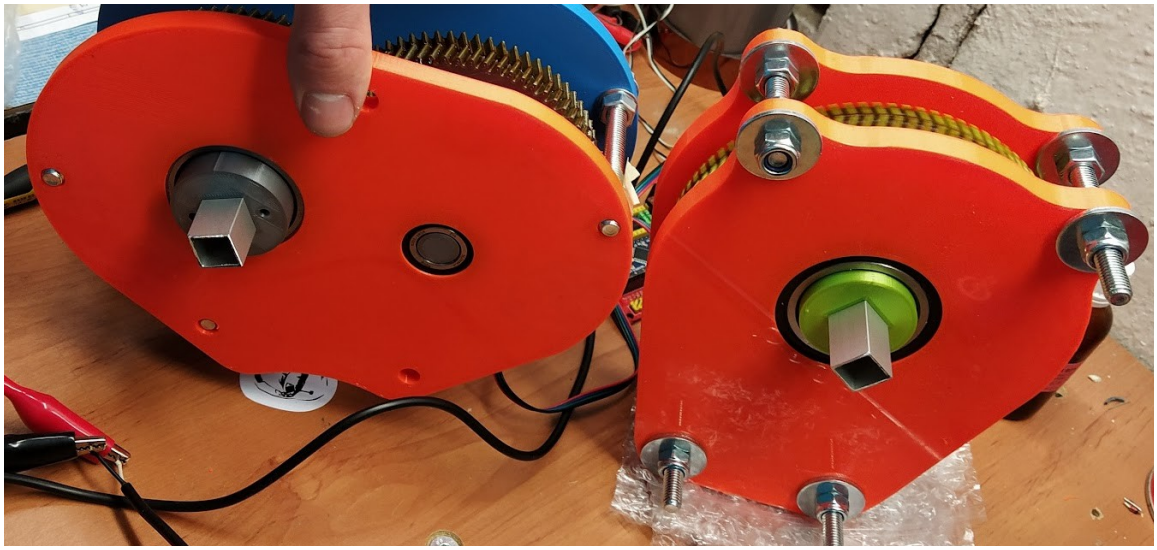
Obr. 21 Původní tříkolová převodovka

Po vytištění a sestavení sice převodovka fungovala dle očekávání, ale dostavil se problém v podobě nedostačujícího místa v zamýšlené konstrukci. Robot by pak přesáhl i předem definovanou váhu do max 10kg.
Navíc se mi zdálo zpřevodování zbytečně vysoké a tak jsem se dal směrem zmenšení a zjednodušení této převodovky. Výsledkem bylo toto : (viz obrázek 22).



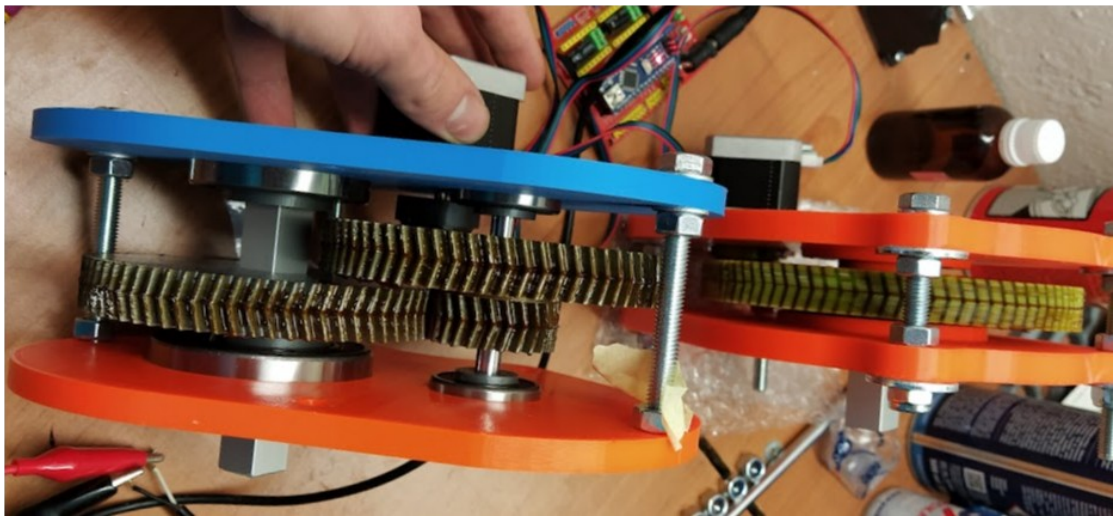
Obr. 22. Rozložený pohled na dvoukolou převodovku

Takto následně převodovka vypadala vytištěná a sestavená (vpravo) v porovnání s předchozí verzí (vlevo):



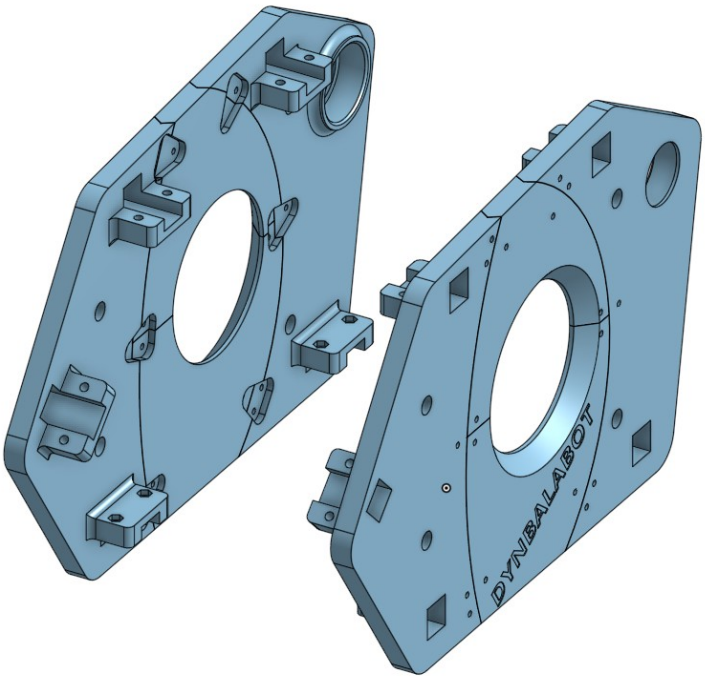
Obr. 23 Převodovky

Došlo hlavně ke zúžení převodovky, úpravu tištěných kol s dvojitým helixem, který byl použit kvůli snížení hlučnosti a axiální námahy na ložiska. Z převodovky byl také odebrán materiál.



Obr. 24 Porovnání tloušťky převodovek

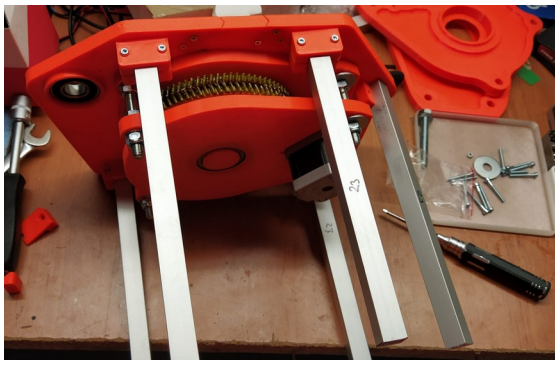
Poměr zubů u nové převodovky byl tak citelně změněn na 6:1 z původních 12.8:1.
Dle rozměrů převodovky jsem začal kreslit i tělo robota. To jsem původně nakreslil jako jeden kus, ale kvůli opravitelnosti a snazšímu tisku jsem ho rozdělil na menší segmenty, které lze rozebírat a v případě poškození nahradit bez nutnosti tisku celé poloviny robota.



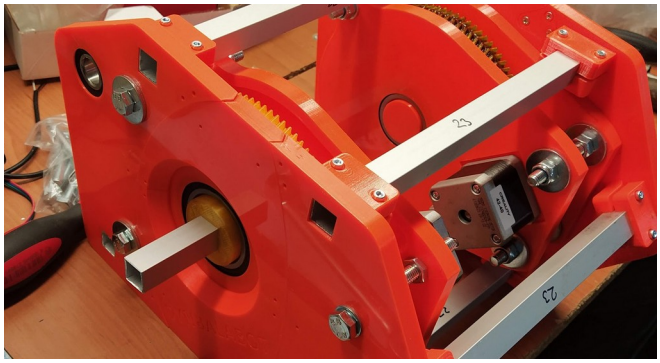
Obr. 25 Těla robota v zrcadlovém zobrazení

Převodovka hraje velmi podstatnou část v pevnosti robota, jelikož drží pohromadě jednotlivé segmenty, které jsou mimo ni přidržovány pomocí zářezů ve tvaru V a pomocných dílů, které slouží k přesnému uspořádání. Dvě poloviny jsou spolu spojeny pomocí hliníkových Al profilů rozměru 15x15mm. a pomocí spon. Ve spodní části je umístěno ESC z hoverboardu, které řídí BLDC motory, pod ním baterie. Úplně nahoře je hlavní řídící deska (motherboard v2). Na převodovkách jsou pak z vnitřní strany umístěny optické koncové snímače s vytištěnými offsetovými S planžetami, které jsou připevněny k výstupnímu kolu převodovky a slouží jako průchozí koncové snímače, aby se

převodovky, které mohou rotovat 360° mohly vynulovat. K řízení převodovek jsou použity dva motory velikosti nema 17, vyznačující se průměrným točivým momentem okolo 420 mN/m.



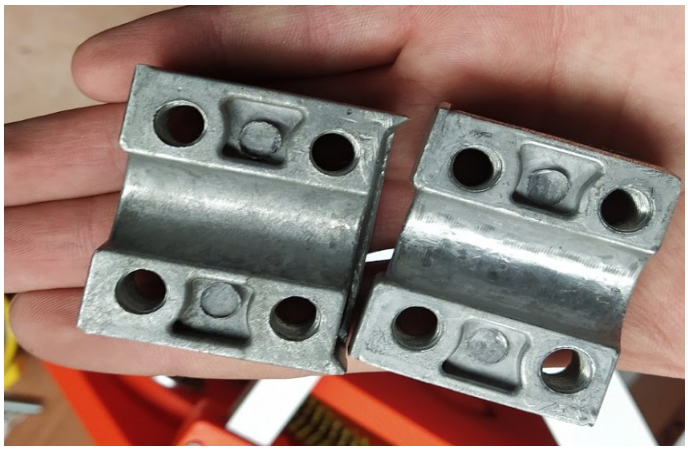
Obr. 26 Ze stavby



Obr. 27 Postavena základní konstrukce

Jako další jsem při stavbě musel vymyslet jak bude fungovat mechanismus nohou. Projekty jako Ascento a SK8O používají celovytištěné nohy, u kterých jsem předpokládal špatnou tuhost a tak jsem raději využil hliníkové profily. Tentokrát velikost 25x25mm. Geometrii nohy jsem se snažil napodobit z projektu Ascento (viz obrázek 1). K uchycená BLDC motorů k nohám jsem kvůli nedostatku vybavení na vyfrézování vlastních držáků využil již existující z hliníkového výlisku z hoverboardu, které jsem vyřízl úhlovou bruskou. Výsledek vypadal následovně (obr. 28, 29):

Obr. 28



Obr. 29 Držáky

Tyto držáky jsem následně připevnil na profil, sloužící jako lýtko robota. Na následujícím obrázku je vidět jak vypadá sestavená noha robota včetně dvou dílům, které označuji výrazem linkages.

Obr. 30 první sestava nohy

3.2.1. Problém

Při osazování elektroniky jsem narazil hned na několik problémů: drivery na PCB měly špatné napájení, pro optické koncové snímače byl oranžový filament příliš průhledný, než aby spolehlivě fungovaly, drivery také způsobovaly extrémní oscilace při provozu (později vyřešeno použitím jiných driverů) a konečne: Motory nedokázaly vyvinout dostatečný tah, aby zvedly váhu robota, která činila bez baterie 4300g.
Nastal tedy problém, který měl dvě zdánlivá řešení:

- zvýšit točivý moment motorů
- snížit váhu robota

Nemohl jsem udělat ani jedno z výše popsaných, protože ke zvýšení točivého momentu bych musel buď: Zvýšit proud do motoru a to by následně způsobilo vyšší zahřívání a možné přehřátí motorů a roztavení plastů převodovky, a nebo bych musel předělat převodovku na vyšší stupeň převodu a ani poté by nebyla jistota, že se nebude lámat 3d vytištěné ozubení.
Snížit váhu robota jsem také nemohl, protože celková váha motorů je přibližně 4000g a tím bych značně posunul těžiště níž a způsobil neideální soustavu pro stabilizaci. Zbývala mi jen jedna možnost: sehnat dvě totožné, přesto zrcadlově opačné torzní pružiny, které by generovaly ideálně lineární tah (nereálné) 2.5kg každá v úhlu 120-45° a tím bych odlehčil motorům. Při bližším studiu obou zmíněných podobných projektů jsem zjistil že skutečně oba používají torzní pružiny a že jejich sehnání v ČR je mírně řečeno obtížné. Obtíž je také, že se běžně pružiny o takové síle nepoužívají a tak je není jednoduché získat ani z hotového výrobku pro jiný účel. Tlačné pružiny by dávaly smysl ale není jednoduchý způsob jak je na robota uchytit a jak zaručit dlouhý lineární chod. Proto jsem se prozatím cíle se zvedáním robota odložil a plně se soustředil na optimalizace balancování a uživatelské přívětivosti celého systému

ZÁVĚR

Původně jsem očekával, že bych mohl tento projekt dokončit 3 měsíce od prvního nápadu. Jak se ale časem ukázalo, tento odhad byl přinejmenším přehnaně optimistický. Celá výroba mi zabrala 7 měsíců, kdy jsem musel kombinovat školu a zbývající čas investovat do projektu. Nedokážu odhadnout jaké množství času jsem na robotovi strávil, ale myslím že se pohyboval ve stovkách hodin.

Nejvíce časové náročná byla nejspíše práce na PC, i když jsem již znalosti programování a CADu měl, musel jsem se zdokonalit ve tvorbě plošných spojů a mechanickém designu. Většina práce tak vznikla nejdříve v PC, odkud jsem následně mohl realizovat model v realitě a doladit případné potíže, na které jsem narazil. S výsledným produktem jsem celkem spokojen. Podařilo se mi totiž zhmotnit myšlenku a v praxi si vyzkoušet chování dynamického systému v podobě sebe stabilizujícího robota, který jsem do té doby znal pouze z teoretických úloh a z internetu.

Práce mě naučila, že mechanická stránka celého projektu byla složitější než ta softwarová(i přesto, že jsem celkem použil 5 programovacích jazyků), a že není vhodné podceňovat komplexnost těchto dynamických systémů.

Během vývoje robota jsem i několikrát narazil. Ať už to bylo při vývoji DynbalaStepperu na složitost celého projektu a množství času, které bych musel věnovat zprovoznění tohoto systému, který není naprosto zásadní pro funkci robota, tak po realizaci mechanické konstrukce, kdy jsem velice špatně odhadl celkovou váhu a složitost.

Kdybych podobný systém realizoval dnes, použil bych například BLDC motory i v převodovkách, jelikož jejich konstrukce a vyšší točivý moment přispívají více odolnosti dynamického systému. Zvolil bych design zaměřený na odlehčení váhy a motory v kolech bych rozhodně zvolil lehčí kvůli energetické náročnosti, která vzniká při snaze rozpohybovat dva dvoukilogramové motory.

Celou dokumentaci jsem primárně formátoval pro Středoškolskou odbornou činnost a tak mohlo dojít k nedodržení některých pravidel formátování určených školou.

SEZNAM OBRÁZKŮ

ZDROJE A POUŽITÁ LITERATURA

[1] Inverted pendulum. *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2022-03-28]. Dostupné z: https://en.wikipedia.org/wiki/Inverted_pendulum

[2] PID controller. *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2022-03-28]. Dostupné z: https://en.wikipedia.org/wiki/PID_controller

[3] Chip shortage. *Cbsnews.com* [online]. New York City: CBS channel, 1927- [cit. 2022-03-28]. Dostupné z: https://www.cbsnews.com/video/global-computer-chip-shortage-could-last-for-years/

[4] AVR. *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2022-03-28]. Dostupné z: https://cs.wikipedia.org/wiki/AVR

[5] STM. *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2022-03-28]. Dostupné z: https://en.wikipedia.org/wiki/STM32

[6] RP2040. *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2022-03-28]. Dostupné z: https://en.wikipedia.org/wiki/RP2040

[7] ESP32. *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2022-03-28]. Dostupné z: https://en.wikipedia.org/wiki/ESP32

[8] Modeling and Control of Two-Legged Wheeled Robot. *Cvut.cz* [online]. Praha: CVUT, 2021 [cit. 2022-03-28]. Dostupné z: https://wiki.control.fel.cvut.cz/mediawiki/images/9/92/Dp_2021_kollarcik_adam.pdf

[9] Ascento: A Two-Wheeled Jumping Robot. *Ascento.ethz.ch* [online]. Zurych: -, 2019 [cit. 2022-03-28]. Dostupné z: https://www.ascento.ethz.ch/wp-content/uploads/2019/05/AscentoPaperICRA2019.pdf

[10] Model Inteligentního skladu. *Youtube.com* [online]. Kutná Hora: -, 2021 [cit. 2022-03-28]. Dostupné z: https://www.youtube.com/watch?v=rUgZJpnDh50&ab_channel=PavelHrstka

[11] Brushless DC electric motor. *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2022-03-28]. Dostupné z: https://en.wikipedia.org/wiki/Brushless_DC_electric_motor

[12] Ottermobile. *Hackaday.com* [online]. -: -, 2018 [cit. 2022-03-28]. Dostupné z: https://hackaday.io/project/160621-the-ottermobile/log/151358-overview#:~:text=These%20typical%20hoverboard%20motors%20come,more%20speed%20for%20bigger%20diameters

[13] Hoverboard firmware hack. *Github.com* [online]. -: -, 2018 [cit. 2022-03-28]. Dostupné z: https://github.com/lucysrausch/hoverboard-firmware-hack

PŘÍLOHA

Veškeré součásti práce, vč schémat, 3d modelů, programu, obrázků a videí se nachází v ZIP archivu RuzickaDYNBALABOT.zip, který je umístěn na serveru SOČ