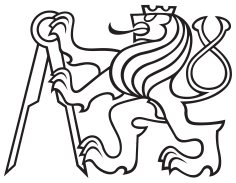
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra řídicí techniky

Soubor řešených úloh pro předmět Roboti s využitím stavebnice LEGO Education SPIKE Prime

Jan Mareš

Vedoucí: Ing. Martin Hlinovský, Ph.D.
Obor: Kybernetika a robotika
Květen 2023

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Mareš** Jméno: **Jan** Osobní číslo: **499031**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra řídicí techniky**

Studijní program: **Kybernetika a robotika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Soubor řešených úloh pro předmět Roboti s využitím stavebnice LEGO Education SPIKE Prime

Název bakalářské práce anglicky:

Set of solved tasks for the subject Robots using the LEGO Education SPIKE Prime kit

Pokyny pro vypracování:

1. Seznamte se s možnostmi stavebnice LEGO Education Spike Prime (současný stav, HW a SW vybavení).
2. Porovnejte LEGO Mindstorms EV3 a LEGO Education Spike Prime.
3. Vytvořte soubor řešených úloh v programovacím prostředí MicroPython pro předmět Roboti.
4. Popřípadě vytvořte přehledné webové stránky se souborem řešených úloh

Seznam doporučené literatury:

[1] https://www.instructables.com/MicroPython-on-SPIKE-Prime/
[2] https://www.eduxe.cz/p/353/45678-spike-prime-zakladni-souprava
[3] https://education.lego.com/en-au/news/spike-prime-tutorials

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Martin Hlinovský, Ph.D. katedra řídicí techniky FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **31.01.2023** Termín odevzdání bakalářské práce: **26.05.2023**

Platnost zadání bakalářské práce: **22.09.2024**

Ing. Martin Hlinovský, Ph.D.	prof. Ing. Michael Šebek, DrSc.	prof. Mgr. Petr Páta, Ph.D.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. Datum převzetí zadání

Podpis studenta

Poděkování

Rád bych poděkoval vedoucímu mé bakalářské práce Ing. Martinu Hlinovskému Ph.D., který mi byl při tvorbě práce nápomocen svými cennými radami a připomínkami. Dále bych poděkoval své rodině a přátelům za podporu během celého studia.

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.

V Praze, 25. května 2023

. . . . . .

Abstrakt

V úvodu této bakalářské práce jsou nejprve uvedeny vlastnosti nejnovější robotické stavebnice z řady LEGO Education, jež se nazývá LEGO Spike Prime. Následně je srovnána s předchozí generací LEGO EV3. A v poslední části této práce jsou pro ni adaptované a následně vyřešené úlohy pro předmět Roboti (B3B35RO1), vyučovaný v prvním semestru bakalářského programu Kybernetika a robotika na Fakultě elektrotechnické Českého vysokého učení technického v Praze.

Klíčová slova: LEGO, Mindstorms, Spike, Prime, EV3, Education

Vedoucí: Ing. Martin Hlinovský, Ph.D.
Katedra řídicí techniky,
Karlovo nám. 13,
121 35 Praha 2

Abstract

This bachelor thesis describes behaviours of the newest robotics building kit from LEGO Education series, which is called LEGO Spike Prime. In the second part is LEGO Spike Prime compared with previous generation LEGO EV3. In the last part are adapted and solved tasks for subject Robots (B3B35RO1) taught by the Department of the controll engineering in Czech technical university in Prague.

Keywords: LEGO, Mindstorms, Spike, Prime, EV3, Education

Title translation: Set of solved tasks for the subject Robots using the LEGO Education SPIKE Prime kit

Obsah

Obrázky

Tabulky

Kapitola 1

Úvod

Tato bakalářská práce pojednává o vlastnostech a možnostech využití stavebnice LEGO Spike Prime. Jedná se o nejnovějšího zástupce v populární rodině LEGO Education, kterou ke vzdělávání základů robotiky a programování používá obrovské množství škol po celém světě již desítky let. Jedním z příkladů využití stavebnice je naše fakulta, která tuto stavebnici využívá v prvním semestru studia bakalářského programu Kybernetika a robotika při výuce předmětu Roboti. Dále pro podporu zájmu o vzdělávání v této oblasti pořádá robosoutěž [1], která je založena na této řadě stavebnic. V ní většinou tříčlenné týmy studentů základních a středních škol soutěží mezi sebou v různorodých úkolech, při kterých musí překonat různé programovací a konstrukční problémy.

V úvodu této práce jsou nejdříve rozebrány vlastnosti hardwaru a softwaru této stavebnice s uvedením nejdůležitějších vlastností a parametrů těchto komponentů. Tyto údaje jsou následně porovnány se stavebnicí LEGO Mindstorms EV3 a zjišťuje se, zda-li ji bude možné bez problému nahradit při výuce v dostatečné kvalitě.

V závěru této práce byly vytvořeny řešené úlohy LEGO Spike Prime, které jsou určeny pro novou akreditaci předmětu Roboti, navržené v diplomové práci Ing. Matějem Štětkou [38]. Příklady řešení těchto úloh mají být nápomocny studentům při jejich implementování či je seznámit s řízením a možnostmi této stavebnice. Úlohy byly s ohledem na odlišné vlastnosti s LEGO Mindstorms EV3 upraveny a adaptovány tak, aby bylo možné dosáhnout rozumných výsledků.

[1] https://robosoutez.fel.cvut.cz

Kapitola 2

LEGO Education

Počátky LEGO Education sahají až do roku 1984, kdy se tehdejší ředitel LEGA Kjeld Kirk Kristiansen inspiroval pořadem, v němž profesor MIT Seymour Papert popisoval, jak učí děti programovat robotickou želvu, která se umí pohybovat vpřed/vzad, otáčet o specifické stupně vpravo/vlevo a uchycovat/pouštět tužku. Chtěl tak posunout již několik roků rozvíjející se propojení LEGO Technic s pneumatikou a motory. Tento rozhovor Kristiansovi vnukl myšlenku, že by nebylo špatné zapojit děti, pro jejich lepší rozvoj, do programování. Následně navštívil MIT Media Lab, kde hledal inspiraci a uzavřel partnerství. [10]

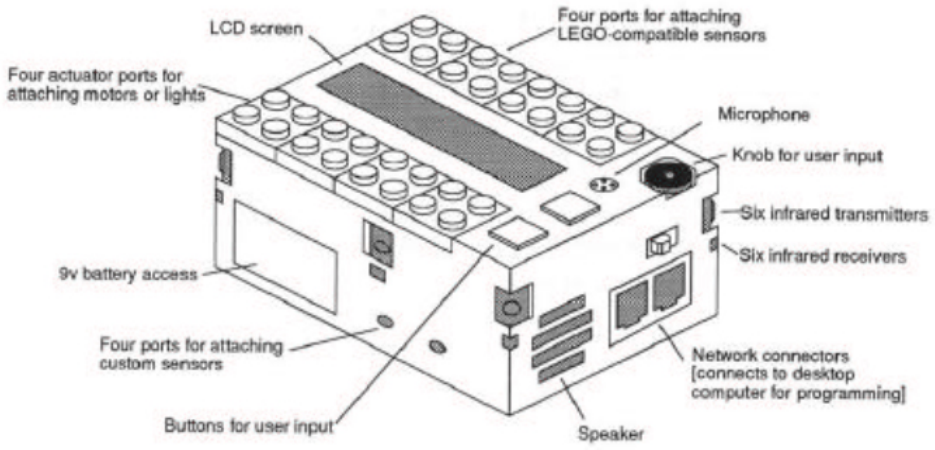
Výsledkem partnerství se stal software vycházející z upravené verze programovacího jazyka LOGO pro LEGO. Díky tomuto softwaru a stavebnici LEGO bylo možné posunout ovládání želvy na úplně jinou úroveň, pomocí kostek LEGO Technic bylo možné postavit jakéhokoliv robota a toho následně programovat dle libosti. Ovšem stavitelé byli limitováni tím, že muselo být vše propojeno s počítačem. V roce 1987 vznikl k tomuto software set LEGO TC[1] LOGO, který byl velice populární ve školách a prodalo se ho jen ve Spojených státech amerických bezmála 15000 kusů do základních a středních škol. Možnosti této stavebnice už tehdy zkoumali na ČVUT, kde má využití této rodiny velikou tradici. Na tento úspěch navázaly na počátku 90. let stavebnice LEGO Technic Control 0 určené pro základní školy, LEGO Technic Control 1 pro střední školy a LEGO Technic Control 2 pro komplexní robotické stavby. Tyto stavebnice byly bohužel dodávány pouze do škol.

Na tyto stavebnice určené pouze do škol navázaly roku 1990 sety 8094

[1] TC = technic control

Control Centre a 8485 Control Centre II, první zmiňovaná umožňovala pomocí jednoduchého ovládacího panelu ovládat až pomocí 50 příkazů 3 motory bez nutnosti připojení k počítači. Mezitím se původní školní stavebnice roku 1993 dočkala ještě aktualizace, která umožnila zapojení světelných a zvukových modulů, ovšem už se počítalo s brzkým uvedením přelomového následovníka. [25]

V laboratoři MIT byl vytvořen roku 1987 první prototyp programovatelné kostky, jež kladl důraz na flexibilitu a umožňoval již do ní nahrát program a fungovat bez nutnosti připojení k PC. Kostka běžela opět na derivátu jazyka LOGO a umožňovala připojit široké spektrum motorů a čidel. Z tohoto prototypu vznikla roku 1995 šedá kostka, jež je znázorněna na obrázku 2.1 a obsahovala LCD display, 8 senzorových portů, 4 motorové, infra červený vstup a zvukový výstup. Následně vznikl i třetí prototyp pojmenovaný červená kostka, který se lišil pouze odebráním 2 senzorových portů. Tyto kostky se ovšem nikdy nedostaly z laboratoře do prodeje, protože v té době nebyly ve školách ještě tolik rozšířené potřebné počítače, aby se produkce vyplatila.

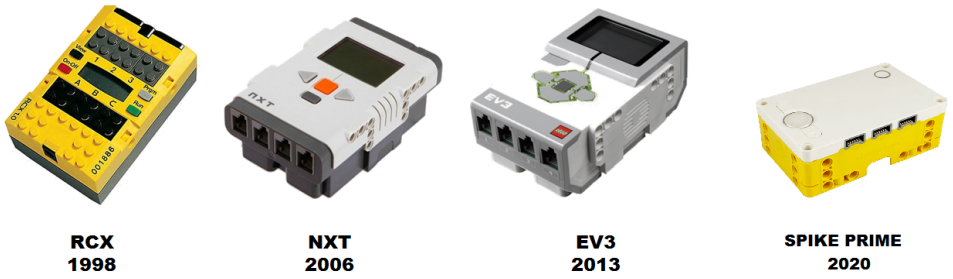


Obrázek 2.1: První prototyp programovatelné kostky. [24]

Na tyto prototypy v devadesátých letech navázali vytvořením programovatelného jazyka LogoBlocks, který vznikl společně s přípravami nové generace, jež byla roku 1998 uvedena na trh. Jednalo se o první generaci LEGO Mindstorms nazvanou RCX, jejíž podrobnější popis je v sekci 2.0.1. [24] Tyto produkty se drží konceptu STEM (Science, Technology, Engineering and Math) a kombinují osvědčené součástky řady Technic s programovatelnými součástkami, které pracují s motory a různými senzory. Po RCX následovaly další generace této populární řady LEGO Mindstorms, NXT (ve dvou verzích), EV3 a Robot Invertor. Pomyslným následníkem této řady LEGO Mindstorms je Spike se Spike Prime, který je derivátem Robot Invertora, jež má nově cílit

na generaci 8-12 let, místo 8-99 let. Na obrázku 2.2 můžeme vidět přehled všech těchto programovatelných hubů, kdy Spike Prime je shodný s Robot Invertorem kromě jeho barvy. Tyto stavebnice se staly významným pomocníkem při výuce na technických školách a využívají se na mnoha významných studentských soutěžích jako je FIRST® LEGO League[2] nebo Robosoutěž[3]. [10] [37]

V řadě LEGO EDUCATION se paralelně vytváří i stavebnice pro menší děti, které jsou blízké řadě LEGO DUPLO, ovšem toto není součástí této práce.



Obrázek 2.2: Vývoj programovatelných hubů. [23]

2.0.1 RCX

První zástupce řady Mindstorms byl vydán roku 1998. Tento kontrolér je možné programovat pomocí RCX (Robotic Command eXplorers) nebo ROBOLABU založeného na LabVIEW. V jádru kontroléru je schován 16 MHz procesor s 32K RAM. Kit dále obsahoval 2 motory, 2 dotykové senzory a světelný senzor. [37] Kostka byla plně bezdrátová, jelikož programování z počítače probíhalo pomocí infra červeného portu. [24]

2.0.2 NXT

Další zástupce této řady se mohl chlubit již 48 MHz procesorem s 64 kB RAM. K dispozici byly 4 vstupy a 4 výstupy, pomocí nichž mohl ovládat a komunikovat s periferiemi. V základním setu se nacházel 1 střední motor, 2 velké motory, 1 ultrazvukový senzor, 2 dotykové senzory a 1 barevný senzor. [4] Kontrolér se programoval pomocí dodávaného softwareu NXT-G. [37]

[2] https://www.firstlegoleague.org/
[3] https://robosoutez.fel.cvut.cz/

2.0.3 EV3

Jedná se o posledního „dospělého“ zástupce z řady LEGO Mindstorms. V kontroléru běžel na 300 MHz procesoru s 64 MB RAM Linux, který měl pro své programy k dispozici 16 MB Flash paměť a podporoval bezdrátové technologie jako WiFi či Bluetooth. Programovací prostředí bylo založeno na LabVIEW a bylo možno pomocí něj ovládat prvky, jež obsahuje hlavní kit. Těmito prvky jsou 2 velké motory, 1 střední motor, 2 dotykové senzory, 1 barevný senzor, 1 gyrospický senzor a 1 ultrazvukový senzor. [10]

2.0.4 Robot invertor

Jedná se o posledního zástupce řady LEGO Mindstorms, který je příbuzný se Spike Prime, jež ho má nejspíš nahradit. Detailní specifikace motorů, senzorů a hubu jsou identické se Spike Prime. Liší se pouze softwarem, ve kterém Robot Invertor ještě nepřišel například o možnost propojení s dalšími huby či možnost dálkového ovládání hubu. Vzhledově má hub pouze odlišnou barvu a v základním setu je jiná skladba konstrukčních dílů.

Kapitola 3

Spike Prime

Stavebnice LEGO Spike Prime byla představena v srpnu roku 2019 jako zástupce nejnovější generace robotické rodiny LEGO Education, jež je určena pro vzdělávání. Stavebnice rodiny LEGO Education mají více jak 40letou tradici, která se odráží ve zkušenostech, s níž jsou stavebnice vytvářeny. Stavebnice Spike Prime je pomyslným nástupcem řady Mindstorms, neboli navazuje na její poslední generaci, kterou je EV3, s níž ji v této práci srovnávám. Řada LEGO Mindstorms byla určena zejména pro rozvoj v oblastech robotiky a programování. [22]

3.1 Hardware Spike Prime

V této sekci popíšu komponenty, které se nachází v základním setu a rozšiřujícím setu. Pro tuto řadu vznikl nejen nový hub, motory či senzory, ale také nové konstrukční díly. Všechny motory a senzory jsou vybaveny pevně připojeným 25 cm vodičem zakončeným LFP2[1] konektorem.

[1] LFP2 = LEGO power function 2

3.1.1 Velký programovatelný hub

Programovatelný hub, znázorněný na obrázku 3.1, je základní součástka stavebnice, která zpracovává data z čidel a řídí aktivní prvky, jako jsou motory či jiné indikační LED, jež se nachází i v čidlech. Hub běží na systému založeném na MicroPythonu, tudíž i programy, kterými se řídí, jsou psány v Pythonu (ve verzi 3). Programy se nahrávají do hubu pomocí rozhraní USB či Bluetooth. Hub má integrovaný gyrosensor a akcelerometr, pomocí něhož známe parametry jeho pohybu a můžeme zpracovávat i jím provedená gesta. Gyrosenzor není moc přesný, což se například při rotaci s jejím růstem zhoršuje a například při 360° dosahuje chyby přibližně 10°. Dalším vstupem jsou programovatelná tlačítka vlevo a vpravo, prostřední slouží pouze k potvrzování v menu a zapínání/vypínání kostky/programu. Jako výstup hubu se dá použít LED v prostředním tlačítku, reproduktor či display. Všechny porty se dají využít v režimu vstupu i výstupu. Nabíjení probíhá pomocí microUSB konektoru, který je zároveň i datový pro nahrávání programu do hubu. Detailní specifikace je v tabulce 3.1.

Display	5x5 LED matice 10 intenzit LED každá dioda samostatně řiditelná
I/O porty	6x LFP2 I/O port 115 kB/s (porty A-D), vysoko-rychlostní (porty E-F) automatická detekce senzorů a motorů
6-osý gyrosenzor	3-osý akcelerometr 3-osý gyroskop módy - gyroskop, akcelerometr gesta - volný pád, třesení, klepnutí
Tlačítkový vstup	prostřední - zapnutí/vypnutí hubu, potvrzení výběru a je podsvícené levé/pravé - navigace v menu, programovatelné
Reproduktor	pro přehrávání zvuku zařízení maximální kvalita 12 bit 16 kHz (mono)
Bezdrátová konektivita	Bluetooth 4.2
Napájení	akumulátor 2100 mAh/7,3 V nabíjení pomocí micro USB
Systém	100 MHz M4 320 kB RAM 1 MB FLASH procesor 32 MB paměť pro program, zvuky a další obsah Embedded MicroPython operační systém

Tabulka 3.1: Přehled parametrů programovatelného hubu. [19]

Obrázek 3.1: Programovatelný hub. [7]

3.1.2 Motory

Střední motor

Střední motor se nachází v setu ve 2 kusech. Motor je velice svižný a poskytuje i zpětnou vazbu jako je jeho natočení či rychlost otáčení. Lze ho používat i jako vstup jeho ručním otáčením. Motor trpí velkou vůlí ve spojení s hřídelí rotoru a vyvedeným uchycením pro ostatní LEGO díly, kdy ho lze volně otáčet o několik stupňů, což znesnadňuje přesné pohyby jako otáčení o určitý úhel či rovnou jízdu. Charakteristika motoru je znázorněna v tabulce 3.2 a je vyobrazena na obrázku 3.2.

Bez zátěže	
Moment	0 Ncm
Otáčky	185 ot/min ±15%
Proud	110 mA ±15%
Maximální výkon	
Moment	3,5 Ncm
Otáčky	135 ot/min ±15%
Proud	280 mA ±15%
Na krátko	
Moment	18 Ncm
Otáčky	0 ot/min
Proud	800 mA ±15%

Tabulka 3.2: Specifikace středního motoru. [20]

Obrázek 3.2: Střední motor. [30]

Velký motor

Tento motor je větší bratr středního motoru, který oproti němu může vyvinout větší moment a má zanedbatelně menší maximální rychlost. V setu se nacházejí po 2 kusech. Velký motor nám poskytuje stejnou zpětnou vazbu jako jeho střední sourozenec a lze ho také použít jako vstup. Trpí také stejnými neduhy jako jeho menší bratr. Specifikace velkého motoru je v tabulce 3.3 a je znázorněna na obrázku 3.3

Bez zátěže	
Moment	0 Ncm
Otáčky	175 ot/min ±15%
Proud	135 mA ±15%
Maximální výkon	
Moment	8 Ncm
Otáčky	135 ot/min ±15%
Proud	430 mA ±15%
Na krátko	
Moment	25 Ncm
Otáčky	0 ot/min
Proud	1900 mA ±15%

Tabulka 3.3: Specifikace velkého motoru. [18]

Obrázek 3.3: Velký motor. [29]

3.1.3 Senzory

Senzor barev

Senzor barev, jež je vyobrazen na obrázku 3.4, je vstupní modul, jenž nám slouží ke čtení barev v několika režimech. První režim je rozpoznání barev, jež jsou zmíněny v tabulce 3.4 a odpovídají barvám kostiček nacházejících se v setu. Dalším režimem je měření intenzity odrazu světla, jež poslouží typicky pro úlohu sledování černé čáry. Posledním režimem je měření dopadajícího světla, což můžeme využít například pro rozpoznání dne a noci. Senzor má na svém čele i 3 segmenty výstupních LED, u nichž se dá každá řídit separátně. U tohoto senzoru je zapotřebí dodržet co nejpřesněji snímací vzdálenost, hlavně při měření intenzity odraženého světla, kdy s oddalováním/přibližováním velice rychle klesá rychlost vyhodnocení a přesnost.

Snímkovací frekvence	100 Hz
Ideální čtecí vzdálenost	16 mm
Režimy snímání	Rozpoznání barvy Měření intenzity odrazu Měření dopadajícího světla
Rozpoznávané barvy	žádný objekt, bílá, modrá, černá, zelená, žlutá, červená, bledě modrá, fialová
Výstupní LED	3 bílé LED s řízenou intenzitou

Tabulka 3.4: Specifikace barevného senzoru. [16]

Obrázek 3.4: Senzor barev. [31]

Senzor vzdálenosti

Senzor vzdálenosti je ultrazvukový senzor, který měří vzdálenost od předmětu v rozmezí 50-2000 mm, kde do vzdálenosti 300 mm měří velice rychle. Všechny parametry jsou k nalezení v tabulce 3.5. Okolo jeho „očí“, jež jsou vidět na obrázku 3.5, se nachází vždy 2 segmenty LED diody rozdělené na horní a dolní polovinu, tyto diody lze jednotlivě řídit a to včetně jejich intenzity.

Tento senzor se dá využít i pro připojení externích rozšíření, jež je možné připojit po oddělání zadního krytu, ovšem přijde tím senzor o jeho původní funkci. Pod zadním krytem je k dispozici 8 pinů, z nichž například pomocí rozšiřujícího modulu na obrázku 3.6 získáme tyto piny ze sběrnice:

- 3,3 V - napájecí napětí
- logická nula
- uart rx / gpio pin
- uart tx / gpio pin
- 0 až +9 V - řízení motoru
- 0 až -9 V - řízení motoru. [21]

Vzorkovací	100 Hz
Měřicí rozsah	50-2000 mm ±20 mm
Rychlé měření	do 300 mm
Rozšířitelnost	pod zadním krytem 8 pin konektor
Výstupní LED	4 segmenty bílé LED s řízenou intenzitou

Tabulka 3.5: Specifikace senzoru vzdálenosti. [17]

Obrázek 3.5: Senzor vzdálenosti. [32]

Obrázek 3.6: Senzor s rozšiřujícím modulem. [21]

Senzor síly

Tento senzor, jež je na obrázku 3.7, má dva režimy, ve kterých se dá použít buď jako tlačítko, nebo senzor síly stisku. V prvním režimu stačí tlačítko stisknout zlehka silou přibližně 1 N. Při měření síly je potřeba počítat s mrtvou zónou do 2,5 N a maximální hodnotou 10 N, která je nad tuto hodnotu fixní. Přehled těchto režimů s jejich parametry je v tabulce 3.6.

Vzorkovací frekvence	100 Hz
Režim stisku	
Aktivační zóna	0-2 mm
Aktivační síla	0,5-1 N ±10%
Režim měření síly	
Aktivační zóna	2-8 mm
Aktivační síla	2,5-10 N
Přesnost	±0,65 N
Rozlišení	0,1 N

Tabulka 3.6: Specifikace senzoru síly. [15]

Obrázek 3.7: Senzor síly. [33]

3.1.4 Kombinace s ostatními stavebnicemi LEGO EDUCATION

Tato robotická stavebnice je již několikátou verzí, která vyšla z dílen firmy LEGO. Kompatibilitu těchto systémů porovnám v následujících bodech [8]:

- SPIKE Essential - Všechny čidla a motory jsou kompatibilní.
- Mindstorms® EV3 / NXT - Stavebnice nejsou vůbec kompatibilní, jelikož používají rozdílné kabely a porty.
- WeDo 2.0 - Neexistuje žádná kompatibilita.
- Powered UP / Control+ - Control+ L a XL motory lze připojit k hubu a budou detekovány, ale jiné prvky nikoliv.
- Robot Invertor - Všechny prvky jsou kompatibilní.

3.2 Software

Ke stavebnici je k dispozici software LEGO Spike, který je dostupný na všechny běžné platformy jako Android, Mac OS, Windows, Chromebook a iPad [9], lze ho ale otevřít i ve verzi pro prohlížeč[2] bez nutnosti instalace, ovšem bez možnosti použít pro propojení s hubem Bluetooth. Na všech platformách je toto programovací prostředí identické, tudíž není problém mezi nimi přecházet.

V první části softwaru se nacházejí tutoriály k použití jednotlivých aktivních součástek stavebnice. Další část obsahuje výukové lekce sloužící k osvojení znalostí a možností práce se stavebnicí. Třetí část obsahuje manuály ke stavbě různých modelů. Poslední součástí jsou vlastní projekty.

V programovacím prostředí je k dispozici terminál, který lze použít k výpisům stavu robota pomocí funkce *print()* či se zde objevují chybová hlášení. Dále je zde možnost v horní části sledovat aktuální stavy všech senzorů, kde lze po rozkliknutí měnit i režim měření.

Projekty lze programovat pomocí několika programovacích jazyků. Nyní je software LEGO Education Spike k dispozici ve třetí generaci, pojmenované LEGO Education SPIKE, která ještě nemá aktualizaci, jež zprovozňuje Python, tudíž tuto práci zpracovávám v předchozí druhé generaci, pojmenované LEGO Education SPIKE Legacy App, v poslední verzi 2.0.10. [11] K programovacímu prostředí používám firmware odpovídající verze. K dispozici je zároveň webové prostředí, které je duplicitní s desktopovým softwarem, který popisuji a používám. Nevýhodou používání starší verze je nepřenositelnost projektů do novější verze softwaru. [28] Jednotlivé jazyky rozeberu v následujících bodech.

3.2.1 Python

Python je implementován pomocí MicroPythonu, který kód napsaný v Pythonu převádí pro běh na mikrokontrolérech a následně je nahrán do kostky. Implentace Pythonu odpovídá jeho verzi 3. Všechny prvky jsou potřeba inicializovat s parametrem portu, na kterém jsou zapojeny, a následně s nimi lze pracovat pomocí metod, jež obsahuje jejich příslušná třída. Potřebná

[2] https://spikelegacy.legoeducation.com/

dokumentace je k dispozici přímo v programovacím prostředí, které má ovšem jednu zásadní nevýhodu, neobsahuje našeptávač, tudíž je nutné si všechny funkce přesně pamatovat, nebo vždy znovu hledat. [26]

Příslušné třídy a vše, co lze pomocí příslušných metod řídit či číst, je rozepsáno v následujícím seznamu. Konkrétní funkci lze nalézt v nápovědě, která se nachází přímo v programovacím prostředí.

Třídy a metody

- PrimeHub

  Pomocí funkcí ovládá tlačítka, reproduktor, gyrosensor a display.

- ColorSensor

  Pomocí funkcí snímá barvy v různých režimech a ovládá příslušné LED.

- DistanceSensor

  Pomocí funkcí rozpoznává vzdálenost a ovládá příslušné LED.

- ForceSensor

  Pomocí funkcí snímá stisk v různých režimech.

- MotorPair

  Pomocí funkcí ovládá příslušný motorový pár.

- Motor

  Pomocí funkcí ovládá příslušný motor.

- Timer

  Pomocí příslušných funkcí ovládá časovač.

3.2.2 Icon blocks

Tento způsob programování je založen na jazyku Scratch. Ten zastiňuje graficky MicroPython, jež je popsán v bodě 3.2.1. Tento jazyk je určen pro prvotní seznámení s programováním, jelikož je nejvíce grafický a obsahuje pouze základní bloky. Programy lze cyklit do základních smyček bez podmínek a reagovat na základní vstupní reakce, jako stisknuté tlačítko, přiblížení na určitou vzdálenost či natočení kostky.

3.2.3 Word blocks

Stejně jako Icon blocks je založen na Scratchy, ovšem neobsahuje pouze základní grafické bloky, jeho bloky jsou rozšířené o další vstupní podmínky a možnosti reakce. Dále lze také používat matematické operace, podmínky či proměnné, jedná se tedy o „dospělejší“ programování oproti Icon blocks.

3.2.4 Použití softwaru Robot Invertor

Pokud se nachází v hubu firmware ve verzi 2.X.X, lze po jeho připojení k softwaru Mindstorms určeného pro Robot Invertor nahrát jeho firmware a využít rozšířené možnosti jeho softwaru. Tento software nabízí oproti originálnímu možnost propojovat huby mezi sebou, připojit externí ovladače od PlayStation 4 nebo Xbox One či ho ovládat na dálku pomocí telefonu. Firmware lze po připojení k originálnímu softwaru přehrát zase zpět.

3.3 Konstrukční díly

Tato stavebnice obsahuje mnoho osvědčených dílů z řady LEGO Technic, které jsou rozšířené o součástky potřebné ke stavbě robotů. Vzniklo i několik nových součástek přímo pro tento set, jedná se například o integrační kostku, která umožňuje propojit LEGO Technic s klasickou LEGO platformou. Dále vznikl technický rámeček a plát pro snadný návrh robotů. Nová kola díky své konstrukci nepotřebují na svoji výrobu tolik plastu, ovšem jsou úzká a mají hodně tvrdou gumu, která má problém na hladším povrchu s adhezí a snadno prokluzuje. Poslední novinkou, které si na první pohled každý znalý předchozích LEGO robotů všimne, jsou klipsy, pomocí nichž se dá zabránit nežádoucímu plápolání či motání kabelů. [6] Bohužel došlo i ke změně konstrukce kuličky, která je nyní plastová v plastovém uložení a má problém se na hladším povrchu otáčet, zanese se i malým množstvím prachu.

Kapitola 4

Porovnání EV3 a Spike Prime

V tabulce 4.1 je vidět porovnání hlavních rozdílů mezi stavebnicemi. Spike Prime přináší řadu vlastností, které jsou dnes jinde běžné, ale zároveň bohužel i zjednodušení, které jsou pro uživatele EV3 nepochopitelné. Některé změny pramení z toho, že je nová generace směřována na mladší věkovou kategorii a cílem bylo ji pro ni zjednodušit. Jednotlivé rozdíly rozeberu v následujících sekcích, v nichž porovnávám zásadní parametry, které se změnily, avšak kompletní výčet parametrů Spike Prime je v sekci 3.1. Největší změny se odehrály v hubu, ale ani ostatní prvky nezůstaly beze změny.

	EV3	**Spike Prime**
Baterie	nabíjecí akumulátor nebo 6x AA baterie	pouze speciální nabíjecí akumulátor
Motory	2 velký a 1 střední	1 velký a 2 střední
Externí senzory	síly, barvy, vzdálenosti, polohy	síly, barvy, vzdálenosti
Kabely	7 samostatných různě dlouhých	připojené k senzorům a motorům
Nabíjení kostky	pomocí externího zdroje	pomocí microUSB konektoru
Počet dílků	537	523
Porty hubu	4-vstupní a 4-výstupní	6 I/O portů
Jezdící kulička	kovová v plastovém uložení	plastová v plastovém uložení

Tabulka 4.1: Přehled hlavních rozdílů mezi EV3 a Spike Prime v základních setech. [27]

4.1 Hardware

Ke Spike Prime jsou k dispozici mnohem rozsáhlejší datasheety součástek než u EV3, pro něž jsou základní údaje popsány pouze v manuálu. V následujících bodech jsem vytvořil přehled základních údajů jednotlivých součástek a srovnal je. Vnější periferie nejsou zpětně kompatibilní z důvodu použití jiných konektorů, ty již nemají jisticí zobáček, který kabel v hubu fixuje před nechtěným vysunutím. Další nevýhodou u připojování vnějších periferií je to, že mají napevno v sobě připojené kabely, tudíž nelze měnit podle potřeby jejich délku a ani je v případě poškození vyměnit, tudíž se poškozením kabelu znehodnotí celá periferie.

4.1.1 Programovatelný hub

U programovatelného hubu se jako u hlavního komponentu stavebnice odehrálo nejvíc změn, ty hlavní hardwarové jsou shrnuty v tab. 3.1. Došlo ke změně operačního systému, z Linuxu na Embedded MicroPython, který má menší hardwarové požadavky a v této souvislosti došlo i k poklesu výkonu hardwaru. Další změny jsou již na první pohled zřejmé, došlo k náhradě černobílého 178x128 maticového displeje za 5x5 LCD display, který může vytvořit mnohem jednodušší grafický výstup a v důsledku toho byl ochuzen o rozsáhlé menu a obrovskou variaci grafických výstupů. V této souvislosti došlo i k redukci tlačítek z 4 směrového + potvrzovacího na 2 směrové (vlevo, vpravo) + potvrzovací. Důsledkem toho je že u Spike Prime lze změnit název hubu, spravovat nahrané programy či sledovat aktuální hodnoty periferií pouze v počítači, nelze tyto operace provést jako v EV3 přímo na kostce.

U vnější konektivity došlo k odebrání slotu na SD kartu, na níž bylo u EV3 možné nahrát jiný operační systém jako například Ev3dev[1], který pomocí upravené linuxové distribuce umožnil programování pomocí MicroPythonu. Na EV3 lze nahrávat projekty v omezeném množství velikostí, ovšem u Spiku je to limitované velikostí menu na 20. Spike má ovšem omezení i u přehrávání zvuků, kdy přímo z kostky je schopný vydávat pouze jednotlivé naprogramované tóny, ale další „složitější“ zvuky vydává již pouze pomocí počítače. Dále nelze Spike Prime propojovat s dalšími huby pomocí USB konektoru či WiFi. U nabíjení došlo také ke změně, kdy separátní napájecí konektor nahradil sdružený microUSB konektor sloužící i pro připojení k PC.

[1] https://www.ev3dev.org/

Obrázek 4.1: Porovnání hubů EV3 a Spike Prime. [1], [7]

	EV3	**Spike Prime**
Operační systém	Linux	Embedded MicroPython operating system
Processor	300 MHz ARM9 Controller	100 MHz M4
RAM	64 MB	320 kB
Flash paměť	16 MB	32 MB
Porty	RJ12 4x vstupní a 4x výstupní	6x LPF2 I/O ports
Port speed	max. 2400 bit/s - 460 kbit/s [12]	115 kB/s $E, F -$ "$high - speed$"
Display	černobílý maticový LCD 178x128 pixelů	5x5 matice z bílých LED s 10 úrovněmi intenzity
Komunikační rozhraní	Bluetooth WiFi USB 2.0 kom. s PC USB 1.1. kom. s jinou EV3	Bluetooth USB 2.0 kom. s PC

Tabulka 4.2: Porovnání programovatelných hubů. [13], [19]

4.1.2 Motory

Původní motory byly koncipovány tak, že střední motor byl menší a rychlejší s menší silou. Ovšem nyní je tento rozdíl potlačen a motory Spiku mají menší podobné otáčky a větší sílu. Další změnu přinesly konstrukce motorů, kde mají nyní sjednocený vzhled a mnohem více možností uchycení ke konstrukci. Nejzásadnější konstrukční změna se odehrála u středního motoru, který již nyní nemá otočenou osu otáčení oproti velkému motoru. Porovnání specifikací je v tabulkách 4.3 a 4.4 a na obrázcích 4.2.

Střední motory

	EV3	Spike Prime
Maximální moment	12 Ncm	18 Ncm
Maximální otáčky	250 ot/min	185 ot/min

Tabulka 4.3: Porovnání středních motorů. [13], [20]

Obrázek 4.2: Porovnání středních motorů EV3 a Spike Prime. [34], [29]

Velké motory

	EV3	Spike Prime
Maximální moment	40 Ncm	25 Ncm
Maximální otáčky	170 ot/min	175 ot/min

Tabulka 4.4: Porovnání velkých motorů. [13], [18]

Obrázek 4.3: Porovnání velkých motorů EV3 a Spike Prime. [36], [29]

4.1.3 Barevné senzory

Původní senzor má 10x rychlejší odezvu než jeho nástupce, který má jinak stejné pracovní režimy, liší se pouze v barvách, jež rozeznává. Barvy korespondují se součástkami, které jsou k dispozici v setu. Nový senzor má navíc 3 segmentový LED výstup okolo senzoru, kterému lze regulovat jas.

	EV3	**Spike Prime**
Snímací režimy	Rozpoznání barev Intenzita odraženého světla Intenzita dopadajícího světla	Rozpoznání barev Intenzita odraženého světla Intenzita dopadajícího světla
Snímací frekvence	1 kHz/s	100 Hz/s
Rozpoznávané barvy	žádný objekt	žádný objekt
Bílá	ano	ano
Modrá	ano	ano
Černá	ano	ano
Zelená	ano	ano
Žlutá	ano	ano
Hnědá	ano	ne
Červená	ano	ano
Bledě modrá	ne	ano
Fialová	ne	ano
Výstupní LED	nemá	3 segmenty bílé LED s řízenou intenzitou

Tabulka 4.5: Porovnání barevných senzorů. [13], [16]

Obrázek 4.4: Porovnání barevných senzorů EV3 [35] a Spike Prime [31].

4.1.4 Senzory vzdálenosti

Nový senzor vzdálenosti umožňuje snímání v menším měřicím rozsahu s menší přesností. Nová verze má ovšem do 300 mm vyšší rychlost, než do větší vzdálenosti. Nabízí ovšem také výstupní LED segmenty, kdy má 2 segmenty okolo každého snímače. Další novinkou je, že po oddělání krytu (jediná periferie, u které to lze) lze na svorky připojit jakýkoliv externí modul. Porovnání senzorů je na obrázku 4.5 a v tabulce 4.6.

	EV3	Spike Prime
Měřicí frekvence		100 Hz
Měřicí rozsah	30-2500 mm ±10 mm	50-2000 mm ±20 mm
Rychlé měření		do 300 mm
Rozšířitelnost		pod zadním krytem 8 pin konektor
Výstupní LED		4 segmenty bílé LED s řízenou intenzitou

Tabulka 4.6: Porovnání specifikací senzorů vzdálenosti. [13], [17]

Obrázek 4.5: Porovnání senzorů vzdálenosti EV3 [3] a Spike Prime [32].

4.1.5 Senzory stisku/síly

U senzoru stisku došlo k výrazné inovaci, kdy senzor nepracuje jen v módu, kdy rozlišuje pouze stavy sepnuto/rozepnuto, ale umí také rozlišovat sílu stisku přibližně v rozmezí 0-10 N. Detailní specifikace senzorů jsou uvedeny v tabulce 4.7 a jsou znázorněny na obrázku 4.6.

Obrázek 4.6: Porovnání senzorů stisku EV3 [2] a Spike Prime sensoru síly [33].

	EV3	Spike Prime
Snímkovací frekvence		100 Hz
Režim stisku		
Aktivační zóna	plný stisk	0-2 mm
Aktivační síla		0,5 - 1 N ±10%
Režim měření síly	nemá tento režim	
Aktivační zóna		2-8 mm
Aktivační síla		2,5 - 10 N
Přesnost		$\pm 0,65$ N
Rozlišení		0,1 N

Tabulka 4.7: Porovnání specifikací senzorů stisku EV3 a sensoru síly Spike Prime. [13], [15]

4.1.6 Akumulátory

EV3 bylo možné napájet buď pomocí 6x AA baterií, nebo pomocí akumulátoru o kapacitě 2,2 A/9 V [12]. Spike Prime je možné napájet pouze pomocí přiloženého akumulátoru o kapacitě 2,1 Ah/7,3 V [14].

4.2 Software

K oběma robotům je dodáván rozsáhlý software, který obsahuje specifikace, nápovědu či programovací prostředí. Originální software předchozí generace umožňoval programovat pouze ve scratchy, ovšem byla možnost pomocí paměťové karty nahrát do robota jiný firmware a programovat ho například pomocí Pythonu. Spike Prime o tyto možnosti přišel a nabízí pouze programování ve svém prostředí pomocí MicroPythonu či Scratche. Jedinou alternativou je přehrání firmwaru od Pybricks, který má vlastní modifikaci Pythoních knihoven.

4.3 Shrnutí

Spike Prime přišel s několika vylepšeními, zhoršení ale ve výsledku převažovala. Nejdřív bych shrnul vylepšení, která jsou dle mého názoru krokem vpřed, ovšem není jich mnoho. Senzor síly nahradil senzor stisku, který má nyní širší možnost využití. V konstrukčních dílech došlo k několika inovacím, kde jako pozitivní hodnotím vznik velkých polí na vytváření projektů a klipsny na uchycení kabelů. Na senzorech přibyly LED diody, které mohou signalizovat různé stavy. Hub lze nabíjet pomocí MicroUSB, tudíž uživatel nepotřebuje další odlišné adaptéry od těch, která má doma. Ovšem i přes vyzkoušení několika adaptérů mě nabíjení zlobilo a do robota pouštělo hodně malý proud, takže se mi nabíjel i přes 10 hodin. Mezi neduhy aktuálního firmwaru patří pravidelné zaseknutí po ukončení většího programu, při čemž se následně odpojí hub při připojení přes Bluetooth od PC či následuje hláška, ať zaktualizujeme firmware, ovšem tato aktualizace zpravidla selže.

Ostatní senzory mají podobné parametry až na snímací frekvenci a pevně uchycený kabel, zakončený LFP2 konektorem, který mu ubírá variabilitu a možnost opravy. Velký hub přišel se softwarem určeným pro Spike Prime o možnost propojení s ostatními huby i variabilitu LCD displeje či operačního systému. Dále na něm ubyly v součtu porty, i když jsou nyní všechny použitelné v režimu vstupu i výstupu. V konstrukčních dílech se bohužel zhoršila kulička, která již není kovová v plastovém uložení, ale plastová v plastovém. Toto uložení dost často drhne a má horší adhezi. Pokud se kulička při konstrukci robota kombinuje s novými koly, která mají díky své malé ploše a tvrdšímu materiálu problém s adhezí, bývá problém robota korektně řídit. Přesnost řízení ještě zhoršila obrovská vůle v uložení os motoru, kdy se s ním dá otočit o několik stupňů bez pohybu rotorem a zaznamenáním enkodérem.

Stavebnice Spike Prime je vhodná k učení se základům programování a robotiky pro děti maximálně na druhém stupni základní školy, ovšem ve srovnání se svou předchozí generací více ztrácí, než získává. U složitějších úloh je zapotřebí více přemýšlet nad tím, jestli lze s limitem vstupů a výstupů, které Spike má, úlohu implementovat. Lze si pomoci například jiným firmwarem (např. pro Robot Invertor), který umožní propojení více kostek, ovšem zvýší se značně složitost implementace. Dále je potřeba dávat pozor na jeho výpočetní výkon, který například rapidně klesne výpisem na terminál, kdy zvládá pouze okolo 20 výpisů za sekundu, které se u úloh, kde je zapotřebí rychlá regulace, jeví jako vysoce nedostatečné.

Kapitola 5

Řešené úlohy

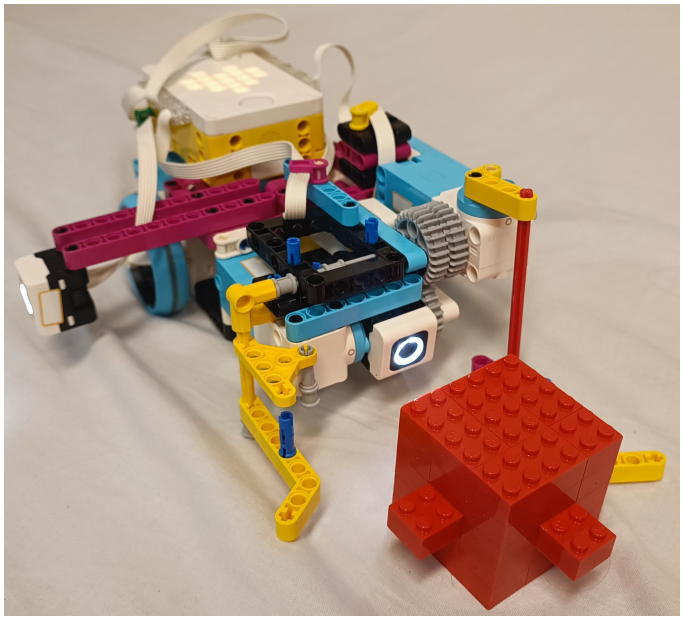
Cílem této části bakalářské práce bylo vyzkoušet úlohy, které ve své diplomové práci [38] vytvořil Ing. Matěj Štětka pro novou akreditaci předmětu Roboti (B3B35RO1), a vyřešit je se stavebnicí LEGO Spike Prime. Ovšem tyto úlohy byly navrženy pro stavebnici LEGO EV3, která, jak bylo rozebráno v kapitole 4, má lepší vlastnosti a dosahuje mnohem větších výkonů. Bohužel díky této skutečnosti jsme byli nuceni zjednodušit zadání, aby bylo možné se stavebnicí Spike Prime dosáhnout obstojných výsledků. Jednalo se převážně o konstrukční a výpočetně výkonové problémy.

Spike Prime je limitovaný počtem I/O portů, tudíž bylo potřeba velice dobře rozmyslet, jak umístit senzory, aby se daly jednoduše použít víceúčelově. Tento úkol byl nepřekonatelný z důvodu požadavku na překonání nájezdových a sjezdových ramp a v kombinaci s omezeným rozsahem pracovní vzdálenosti senzoru barev. Jelikož byl tento senzor určený nejen ke sledování černé čáry, ale i k rozpoznání konce dráhy, bylo nutné ho umístit dostatečně dopředu před osu zatáčení tak, aby snímal „propast“ a zároveň se eliminovalo dopravní zpoždění ke sledování černé čáry. Jako řešení tohoto problému byla navržena konstrukce, díky níž senzor mohl kopírovat terénní nerovnosti. Ovšem z důvodu potřeby umístit na tuto pohyblivou konstrukci ještě další senzory a mechanizmus na zvedání kostky (potřeba počítat i s hmotností vezoucí se kostky) se posunulo těžiště robota do této části. V důsledku přesunu těžiště robot ztratil adhezi (ve stavebnici se nachází pouze úzká kolečka s tvrdou gumou) a stal se celkově neřiditelným. Zároveň se i zvedací konstrukce stala přetížená a nekopírovala hladce terén. V důsledku toho jsme byli nuceni upravit úlohy tak, aby se vše odehrávalo ve vodorovném prostoru. Pravidla a bodování zůstávají zachovány, jen dráhy jsou modifikovány tak, jak je možno vidět v následujících sekcích.

5.1 Stavba robota

Robot byl konstruován tak, aby mohl co nejlépe plnit zadané úlohy, jeho ukázka je na obrázku 5.1. Požadavky na vlastnosti konstrukce robota vyplývající ze zadání úloh jsou:

- Barevný senzor umístit tak, aby se nacházel před osou zatáčení, s cílem minimalizovat dopravní zpoždění na rozumnou mez a zároveň aby nebyl robot moc dlouhý.
- Senzor pro sledování stěny je potřeba umístit do výšky maximálně 6,5 cm, jelikož stěny bludiště jsou vysoké 7 cm.
- Mechanizmus zvedání kostky je potřeba uzpůsobit na to, aby ho bylo možné zkombinovat s pohybem senzoru na čele robota, jelikož potřebujeme sdílet motory na tyto 2 činnosti.
- Robot nesmí být moc dlouhý, aby se snadno otáčel v bludišti, jedno pole má rozměry 280x280 mm a minimální vzdálenost rozpoznatelná ultrazvukovým senzorem je 5 cm, neboť ideální snímací vzdálenost světelného senzoru je 16 mm.
- V případě využití senzoru na sledování stěny je vhodné ho umístit tak, aby se robot pohyboval ve středu políčka.



Obrázek 5.1: Konstrukce použitého robota.

5.2 Stavba programu

Program se skládá z několika dílčích částí, to platí pro jakékoliv úlohy u Spike Prime, některé se mohou podle potřeby zkrátit či modifikovat.

Jako první se provádí import knihoven, všechny knihovny potřebné pro Spike Prime jsou v kořenovém adresáři projektů, tudíž je stačí vždy pouze naimportovat v programu. Jedná se o knihovny Spike, které slouží k ovládání periferií a spike.control pro čítače a časovače. To stejné platí i pro standardní Python knihovny. Níže je ukázka importu všech základních knihoven, toto se automaticky vygeneruje při založení projektu:

```
from spike import PrimeHub, LightMatrix, Button, StatusLight,
                                ForceSensor, MotionSensor,
                                Speaker, ColorSensor, App,
                                DistanceSensor, Motor, MotorPair
from spike.control import wait_for_seconds, wait_until, Timer
from math import *
```

Dále je třeba před použitím inicializovat periferie a všechny potřebné proměnné. Periferie se inicializují pomocí příslušných funkcí a vytvoří se instance jejich třídy, kde lze následně používat její metody. Níže je ukázka inicializace párového motoru (porty A, B) a barevného senzoru (port F):

```
motor_pair = MotorPair('C', 'D') #Inicializace paru motoru v
                                portech "C"-levy motor and "D"-
                                pravy motor
color_sensor = ColorSensor('F') #Inicializace barevneho senzoru
                                v portu "F"
```

Nyní se již píše hlavní program, kde je využit konstruktor jazyka Python, všechny LEGO periferie lze řídit pomocí metod jejich příslušných instancí. Následuje ukázka smyčky, kdy robot jede, dokud jeho senzor světla nezaznamená červenou barvu:

```
while(color_sensor.get_color() != "red"):
    motor_pair.start()
motor_pair.stop()
```

5.2.1 Inicializace periferií a knihoven

Před tím, než je možné přistupovat k metodám jednotlivých periferií, je musíme inicializovat. Jelikož jsou ve všech úlohách použity stejné periferie, jímž náleží stejné knihovny, ukážu je i s popiskem pouze zde.

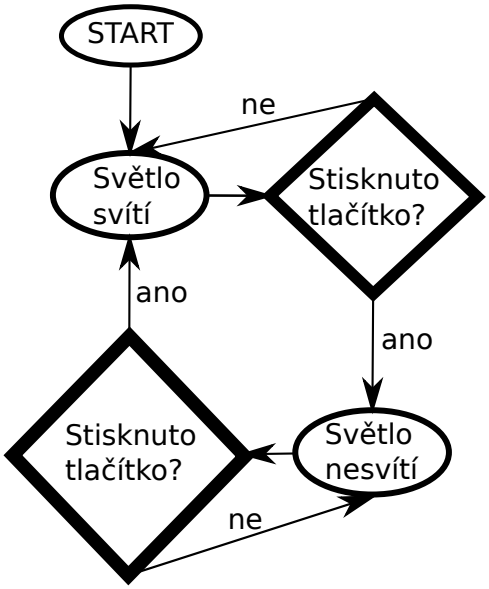
```
from spike import PrimeHub, LightMatrix, Button, StatusLight,
                                ForceSensor, MotionSensor,
                                Speaker, ColorSensor, App,
                                DistanceSensor, Motor, MotorPair
from spike.control import wait_for_seconds, wait_until, Timer
from math import *

hub = PrimeHub() #Inicializace Spike Prime HUBu
wall_sensor = ColorSensor('A') #Inicializace barevneho senzoru k
                                 rozpoznani kostky v portu 'A'
distance_sensor = DistanceSensor('B') #Inicializace senzoru
                                vzdalenosti v portu 'B'
motor_pair = MotorPair('C', 'D') #Inicializace paru motoru v
                                portech 'C'-levy motor and 'D'-
                                pravy motor
color_sensor = ColorSensor('E') #Inicializace barevneho senzoru
                                ke sledovani povrchu drahy v
                                portu 'E'
motor_arm = Motor('F') #Inicializace samostatneho motoru ke
                                zvedani kostky v portu 'F'
```

5.2.2 Stavový automat

Pro řešení 2.-4. úlohy je využit s výhodou stavový automat. Jedná se o metodu příkazování robota či jiného automatu, kdy robot vykonává určitou činnost podle toho, v jakém stavu se nachází. Tuto činnost opakuje, dokud není splněna podmínka k opuštění tohoto stavu, a podle splněné podmínky přechází do stavu jiného a v něm setrvá, dokud se opět nesplní podmínky k jeho opuštění.

Na obrázku 5.2 je ukázka stavového automatu na ovládání světla, kdy se stiskem tlačítka změní jeho stav. Můžete si povšimnout, že v kolečkách jsou stavy a v kosočtvercích jsou rozhodovací bloky. Mezi těmito obrazci se program pohybuje podle šipek a případných výsledků rozhodnutí zobrazených u nich.



Obrázek 5.2: Ukázka stavového automatu.

5.3 Kontrukce drah pro roboty

Dráhy jsou konstruovány v kombinaci drah tištěných na plotteru s bloky umístěnými v soutěžní ploše. Rozměry tištěných drah jsou šířky 1060 mm a délky dle potřeb příslušné úlohy. Bloky o rozměru 280 mm x 280 mm x 70 mm jsou umístěny v ploše o rozměrech 2560 mm x 1720 mm ohraničené stěnami vysokými 70 mm.

5.4 Řešení úloh

5.4.1 Jízda po čáře

Zadání úlohy

Cílem této úlohy je, aby robot dojel co nejrychleji podél černé čáry z pole start (číslo 1) do pole cíl (číslo 2). Příklad tvaru dráhy je na obrázku 5.3. Úlohu je možné zahájit stisknutím tlačítka, ovšem dále se již robot musí pohybovat autonomně a nesmí explicitně znát parametry tratě. Během své jízdy si robot nesmí jakkoliv zkrátit cestu a ani se vzdálit od čáry o více jak 20 cm.

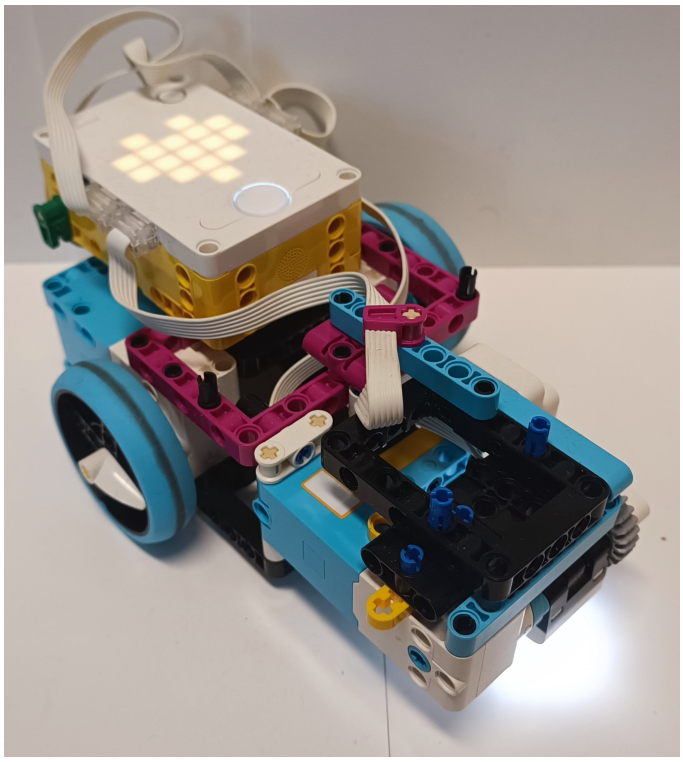
Rozbor řešení úlohy

Jedná se o typickou úlohu pro začátečníky, kteří se seznamují s programováním robotů, díky níž mají možnost pochopit funkci regulátoru na velice jasné úloze. Výhodou této úlohy je, že na ní mohou použít úplně jednoduchý bang-bang regulátor či i nějaký sofistikovanější a složitější jako je PID regulátor. Níže ukážu řešení pomocí obou těchto regulátorů. Regulace pohybu robota je založena na regulaci výkonu motoru podle hodnot barevného senzoru (v režimu rozpoznání barvy nebo intenzity odraženého světla). S regulátory se běžně setkáváme v mnoha aplikacích, které běžně používáme, například u tempomatů (reguluje rychlost), regulace topení (reguluje teplotu) či řízení motorů (reguluje otáčky).

Obrázek 5.3: Schéma dráhy pro robotí sledování čáry.

Stavba robota

Pro řešení této úlohy lze vytvořit robota téměř jakékoliv jednoduché konstrukce. Základní a nejjednodušší konstrukce je založená na zkombinování dvou paralelně umístěných motorů pro pohyb robota s kuličkou umístěnou minimálně 5 cm před nimi tak, aby byl robot vyvážen. Pro sledování černé čáry je potřeba umístit barevný senzor přibližně 7 cm před osu zatáčení tak, abychom eliminovali dopravní zpoždění a byli schopni robota jednoduše řídit. Tento senzor musí být natočeno směrem k povrchu, po kterém robot jede. Příklad konstrukce je na obrázku 5.4.



Obrázek 5.4: Konstrukce robota pro sledování čáry.

Regulační parametr

Regulační parametr, neboli požadovaná hodnota, je cíl řízení procesu, který nechceme sami stále řídit a nepřetržitě sledovat. Dle regulačního parametru řídíme akční zásah regulátoru tak, abychom měli na výstupu požadovanou hodnotu a tedy nulovou poruchu (rozdíl mezi skutečnou a požadovanou hodnotou). [5]

Jsou dva základní způsoby určení akčního zásahu. První způsob je založený na tom, zda-li se barevné čidlo nachází nad bílým nebo černým povrchem. Pro tento způsob používáme čidlo v režimu rozpoznání barev. Robot například při bílé barvě zatáčí vlevo a při černé vpravo, tudíž přejíždí stále přes levou hranu. V případě prohození reakcí na regulační parametr dojde pouze k prohození sledované hrany. Tohoto se zpravidla využívá u nejjednoduššího bang-bang regulátoru, který je popsán níže.

Jako regulační parametr se dá zvolit přímo hrana, díky čemuž může robot mnohem méně kmitat a jet rychleji. Pro tento způsob je nutné použít čidlo v režimu měření intenzity odraženého světla. Musíme si tedy změřit intenzitu odraženého světla na čáře a mimo ni, následně z těchto údajů spočítáme průměr. Tento údaj nám značí intenzitu odraženého světla na hraně čáry a my budeme robota regulovat tak, aby zatáčel na jednu stranu při snížení intenzity a na druhou při zvýšení. Díky takto odstupňované odchylce máme možnost regulovat i akční zásah (intenzitu zatáčení robota).

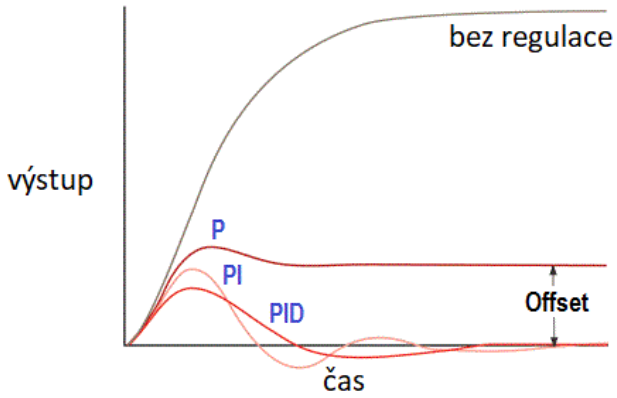
Bang-bang regulátor

Toto je nejjednodušší regulátor, který má pouze dva stavy. Čidlo rozpoznává bílou nebo černou barvu a následně podle toho zatáčí v závislosti na barvě doprava nebo doleva. K řízení motorů použijeme funkce pro řízení motorů v páru, která nám vyřeší situaci, při níž máme motory otočené o 180°. U Spike Prime jde kód napsat například následovně, ovšem robot projíždí dráhu velice pomalu:

```
while( not hub.left_button.is_pressed() ): #hlavni smycka
                                    programu, program zastavi
                                    stiknuti leveho tlacitka
    color = color_sensor.get_color() #ziskani barvy z barevneho
                                        senzoru
    if color == 'black': #stav cerna - zataceni doleva
        motor_pair.start_tank(20, 5)
    elif color == 'white': #stav bila - zataceni doprava
        motor_pair.start_tank(5, 20)
    else:
        print("ERROR: spatna barva")
```

PID regulátor

PID regulátor je univerzální regulátor s jednoduchým návrhem, jak již velká písmena v názvu napovídají, jedná se o zkratku jeho složek. Složky jsou **P**roporcionální, **I**ntegrační a **D**erivační. Pomocí jejich součtu vzniká reakce na regulační odchylku, která je váhovaná koeficienty jednotlivých složek, neboli akční zásah. Na obrázku 5.5 vidíme rozdíl v použití jednotlivých druhů regulátoru a můžeme pozorovat vlivy jednotlivých složek, jejichž přehled následuje.



Obrázek 5.5: Porovnání P, PI a PID regulátoru. [5]

- **P**roporcionální - Jedná se vlastně o zesilovač, kdy proporcionální složka násobí regulační odchylku a snižuje odchylku. Vlivem zvyšující se proporcionální složky může ovšem dojít k nestabilitě systému a v krajním případě až k dosažení dorazu a poškození zařízení. Toto může eliminovat integrační složka, pokud nemá soustava sama o sobě integrační charakter.

$$u(t) = K \cdot e(t), \tag{5.1}$$

  kde u(t) je akční zásah, K zesílení P složky a e(t) regulační odchylka. [5]

- **I**ntegrační - Akční zásah odpovídá době, po kterou měl systém danou regulační odchylku, a tím pádem zvyšuje neustále svůj akční zásah. Integrace má vliv na zpoždění akčního zásahu (jelikož má složka fázový posuv -90°) a jeho přehnanou reakci, tudíž je vhodná kompenzace derivační složkou.

$$u(t) = u_0 + \frac{1}{T_I} \int_0^{\tau} e(t)\, d\tau, \tag{5.2}$$

  kde u(t) je akční zásah, T_I určuje podíl integrační, τ doba integrace a e(t) regulační odchylka. [5]

- **D**erivační - Tato složka slouží ke zklidnění systému a tlumí zákmity systému. Zároveň také vyrovnává fázový posuv vzniklý integrační složkou, tím se eliminuje zpoždění reakce.

$$u(t) = T_D \frac{de(t)}{dt}, \tag{5.3}$$

kde u(t) je akční zásah, T_D určuje intenzitu integrační a e(t) regulační odchylka. [5]

Určení regulačního parametru PID regulátoru

Pro správnou funkčnost programu je zcela zásadní určení přesné hodnoty regulačního parametru. Tento parametr je vhodný měřit přímo robotem a v konstrukci, jaká bude použita pro jízdu po dráze. K jeho určení připojíme robota k programu Spike Legacy a otevřeme vlevo nahoře záložku připojení k hubu, rozbalíme záložku vlastností u světelného senzoru a přepneme na režim měření odraženého světla. Toto měření je vhodné opakovat, abychom eliminovali lokální poškození dráhy nebo nějaké nečistoty. Provedeme proto aspoň 3 měření pro černou i bílou barvu na dráze, které následně zprůměrujeme. Tyto zprůměrované hodnoty pro černou a bílou zprůměrujeme společně mezi sebou a tím získáme hodnotu odraženého světla pro hranu, po které chceme jet.

Použití PID regulátoru pro tuto úlohu

Použití PID regulátoru je v této úloze velice vhodné, pomocí postupného ladění jednotlivých složek lze naladit kvalitní regulátor. K naladění existuje i několik jiných sofistikovanějších metod, kdy jedna z nejpoužívanějších je Ziegler–Nicholasova metoda. Tato metoda se bohužel díky menší rychlosti regulace tohoto robota těžko aplikuje, jelikož se robot velice rychle rozkmitá. Proto je v tomto návodu zmíněna metoda postupného ladění jednotlivých složek. Tento postup se volí proto, že je mnohem obtížnější naladit všechny složky naráz, než je ladit postupně.

Nejdříve začínáme ladění **P** složky, s níž jsme schopni robota uregulovat na čáře, ovšem pohyb nebude tak hladký a akční zásah tak rychlý jako s použitím ostatních složek. Naladěním **P** složky získáme její přibližnou hodnotu k ladění dalších složek. Nyní se můžeme rozhodnout, zda budeme chtít vytvářet **PI** (**PID**) či **PD** regulátor, v tomto případě můžeme přeskočit ladění **I** složky. Při ladění **I** složky je potřeba postupovat po menších krůčcích než u **P** složky,

jelikož nám velice urychlí akční zásah. Potom, co naladíme regulátor s **I** složkou, můžeme přidat i složku **D**, která nám regulaci zklidní, a i zde je potřeba postupovat mnohem jemnějším krokem, abychom nepřetlumili akční zásah. U tohoto robota není vhodné použití I složky, jelikož nemá dostatečně rychlé čidlo a výpočetní výkon hubu, tudíž nestihneme zpracovat dostatečně rychle akční zásah.

Následující kód je ukázka implementace **PID** regulátoru, kde je zapotřebí nejprve inicializovat všechny proměnné a pustit si regulátor v cyklické smyčce, ukončené například stisknutím tlačítka jako v příkladu.

```
black = 5   #hodnota odrazeneho svetla pro cernou caru
white = 99  #hodnota odrazeneho svetla pro bile pozadi
center = (white - black)/2 + black  #vypocet stredni hodnoty

#hodnoty PID regulatoru
Kp = 1.45   #proporcionalni slozka
Ki = 0.01   #integracni slozka
Kd = 0.005   #derivacni slozka

#Pomocne promenne
Integral = 0    #promenna pro scitani odchylek
LastError = 0   #promenna pro ulozeni posledni odchylky
Derivate = 0    #promenna pro ulozeni zmeny

motor_pair.set_default_speed(55)
while( not hub.left_button.is_pressed()): #hlavni smycka
                                programu, program zastavi
                                stiknuti leveho tlacitka
    SensorValue = color.get_reflected_light() #aktualni hodnota
                                    odrazeneho svetla
    Error =SensorValue - center #aktualni chyba
    Integral = Error + Integral #secteni odchylek
    Derivate = Error - LastError #vypocteni zmeny
    Turn = Kp*Error + Ki*Integral + Kd*Derivate #vypocet
                                    konstanty zatoceni
    LastError = Error #ulozeni posledni chyby
    if (Turn >100): #saturace zataceni
        Turn = 100
    if (Turn < -100):
        Turn = -100
    if Integral > 1000: #saturace integracni slozky
        Integral =1000
    if Integral < -1000:
        Integral = -1000
    motor_pair.start(steering = int(Turn)) #ovladani zataceni
                                    motoru
motor_pair.stop()
```

5.4.2 Bludiště

Zadání

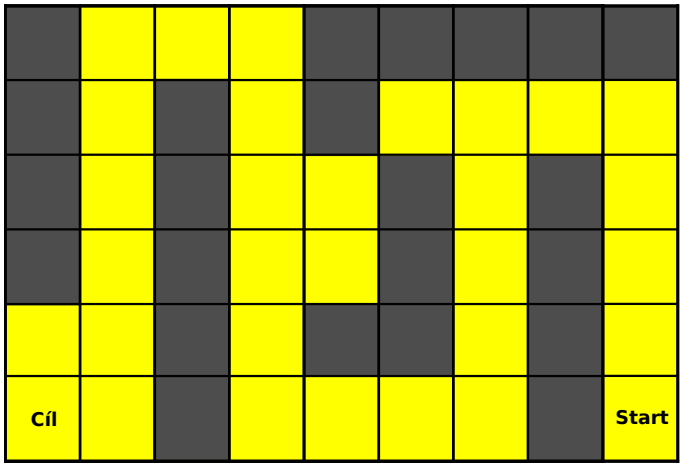
Cílem úlohy je, aby robot po stisknutí tlačítka vyrazil z pole start a dojel co nejrychleji do pole cíl. Na obrázku 5.6 je znázorněn plánek bludiště, kdy černě jsou označeny bloky o výšce 7 cm a žlutě je označena dráha, po které se může robot pohybovat. Tato dráha je ohraničena stěnou o výšce 7 cm. Robot nesmí mít jakkoliv explicitně zadány jakékoliv parametry trasy a musí se pohybovat autonomně.

Rozbor řešení úlohy

Klíčem k řešení této úlohy je správná kombinace typů a umístění senzorů a pohonů, rozpoznání stěn bludiště, následně pomocí získaných dat nalézt cestu ze startu do cíle bludiště. Potíž je v tom, že předem neznáme tvar bludiště, tudíž musíme cestu hledat a vyhodnocovat v každém kroku.

Nabízí se zde mnoho jednodušších či složitějších algoritmů, které nejdou buď z důvodu nízké přesnosti senzorů, nebo složitosti konstrukce, implementovat. Jedno z jednoduchých řešení, které se nabízí, je jízda rovně, při dojetí k překážce vždy zastavit a otáčet se o 90°, dokud před ním není volná cesta. Jenže toto řešení má mnoho překážek, které vyloučí i mnoho dalších jednoduchých algoritmů na projíždění čtvercového bludiště. První překážkou je spínání a regulace motorů, které nedokážou sepnout ve stejné chvíli, a tudíž nelze pomocí standardních funkcí vyjet rovně. Další překážkou je přesnost gyrosenzoru, která roste se vzdáleností od nuly tak moc, že při rotaci o 360° má chybu okolo 10°. Z těchto důvodů by se nám robot vychyloval z osy bludiště jízdou rovně i otáčením. Toto by se dalo kompenzovat konstrukcí, která by ho usměrňovala při přiblížení ke stěnám, ovšem v dalším úlohách by nám mohla být na škodu.

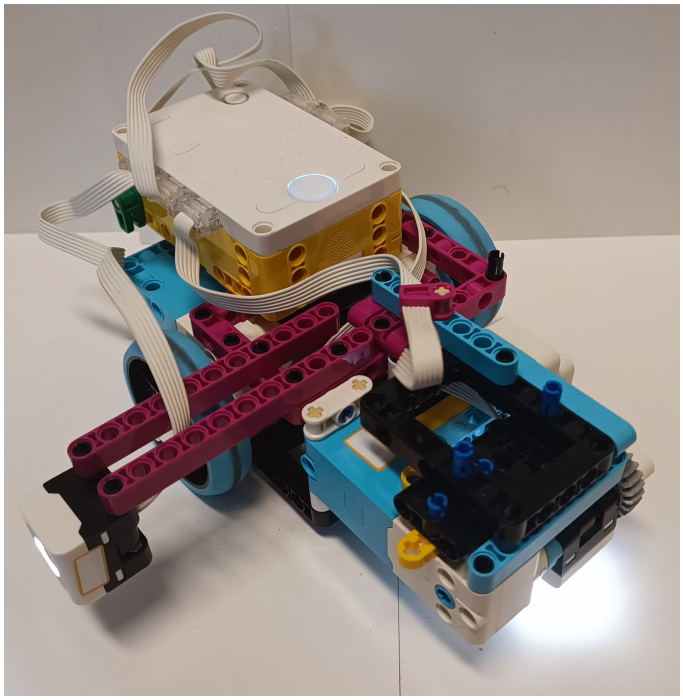
Pro kompenzaci výchylek je nutné sledovat pozici robota v bludišti po co největší část pohybu, například pomocí sledování stěny, podél které robot jede a také pomocí překážek před ním. K tomuto je vhodné použít barevný nebo ultrazvukový senzor. Pomocí barevného senzoru dosáhneme mnohem lepších výsledků díky mnohem vyšší rozlišovací schopnosti (citlivosti).



Obrázek 5.6: Schéma bludiště.

Stavba robota

V této úloze rozšíříme konstrukci robota z úlohy 5.4.1 o boční senzor pro sledování stěny, který umístíme do vhodné výšky tak, aby nebyl nad stěnou vysokou 7 cm. A zároveň do vhodné vzdálenosti tak, aby se robot pro jednodušší implementaci, jak je popsáno dále, pohyboval ve středu dráhy. Dále je potřeba otočit a umístit senzor, kterým jsme sledovali čáru, tak aby byl na čele a koukal směrem před robota pro rozpoznání překážek. Nejvhodnější je tento senzor umístit na pohon, který s ním bude otáčet pro režim sledování černé čáry a překážky před ním, což budeme potřebovat v další úloze. Příklad konstrukce je na obrázku 5.7.



Obrázek 5.7: Konstrukce robota pro průjezd bludištěm.

Implementace

Podle žádané polohy robota v bludišti při sledování stěny je nutné umístit barevný sensor tak, aby se robot pohyboval ve středu dráhy. Toho využijeme v závěrečné úloze, kde ve středu pole bude hledat barevnou kostku, nebo při otáčení. Díky tomuto umístění je potřeba řešit v zásadě jen dvě programovací podúlohy. Robot díky sledování stěny zvládá rovné úseky a otáčky ve směru přilehlém k čidlu sledujícímu stěnu. Za přilehlou stěnu zatočí díky sledování stěny. Poté už je potřeba vyřešit pouze reakce na překážku před robotem, na kterou je vhodné zatočit na odvrácenou stranu od umístění čidla. Kombinací těchto dvou pohybů je robot schopný projet celou trasu, implementace těchto pohybů je popsána níže.

Otáčení se o 90°

K zpřesnění otáčení robota je vhodné použít gyrosensor, který je přesnější než využití enkodéru na hřídeli kola. A to z důvodu vůle v uložení rotorů motoru, kterou nemáme jak snímat, tudíž se nám může stát, že po dokončení

pohybu o 90° a zabrždění motoru se nám ještě může vlivem setrvačnosti o několik stupňů robot natočit. Toto otočení není ani měřitelné enkodérem. K eliminaci chyby gyrosenzoru je vhodné jeho resetování před každým pohybem. Před otáčením je dále vhodné vyrovnat vzdálenost robota od stěny (pomocí pohybu dopředu/dozadu) tak, aby po otočení byl senzor stěny ve vhodné vzdálenosti a robot zbytečně při pohybu neosciloval.

V následujícím příkladu je znázorněno otočení doleva.

```
hub.motion_sensor.reset_yaw_angle() #reset yaw uhlu gyrosenzoru
motor_pair.set_stop_action("hold") #zastaveni motoru bez dojeti
motor_pair.stop()
while(get_yaw_360() > 290 and get_yaw_360() <= 359 or
                                get_yaw_360() <= 10 ): #rozjeti
    motor_pair.start_tank(left_speed=-40, right_speed=40)

while(get_yaw_360() >= 270): #pomalejsi dojeti
    motor_pair.start_tank(left_speed=-5, right_speed=5)
motor_pair.stop()
```

Sledování stěny

Ke sledování stěny je vhodné použít **PD** regulátor, který sleduje intenzitu odraženého světla od stěny a podle toho reguluje výkon motorů. Stačila by nám pouze **P**roporcionální složka, ale **D**erivační nám zklidní pohyb a nerozkmitá se například na nerovnostech stěny. Kód je pouhou modifikací kódu z první úlohy, změnila se sledovaná veličina a ubrala integrační složka.

```
Kp = 0.35
Kd = 1.5
LastError = 0
error = 0
while(True):
    SensorValue = wall_sensor.get_reflected_light() #aktualni
                                    hodnota odrazeneho svetla
    Error = wall_ref_int - SensorValue #aktualni chyba
    Derivate = Error - LastError #derivacni slozka
    LastError = Error #predchozi chyba
    correction = Error *Kp + Kd * Derivate #velikost akcniho
                                    zasahu
    motor_pair.start_tank(left_speed = int(speed + correction) ,
                                     right_speed = int(speed -
                                    correction))
```

5.4.3 Detekce barev

Zadání

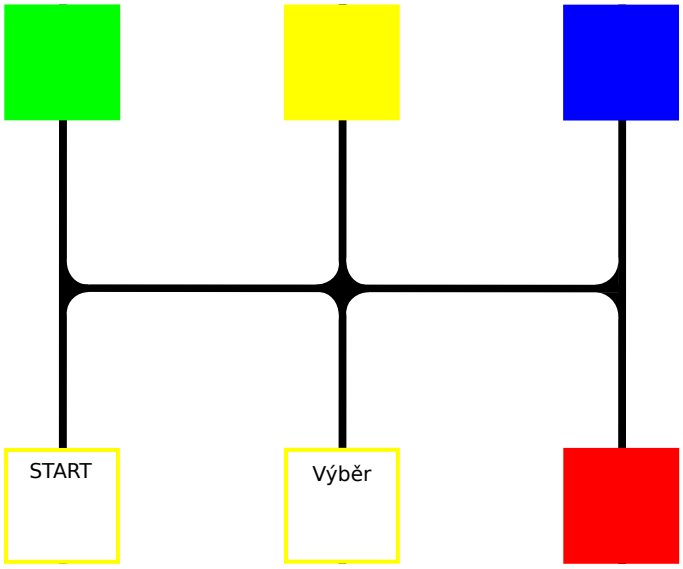
Cílem této úlohy je, aby robot pomocí sledování černé čáry dojel do pole *výběr*, ve kterém je barva cílového pole. Po rozpoznání a oznámení o jakou barvu se jedná, musí robot opět pomocí sledování černé čáry dojet do příslušného cíle odpovídající barvy, kde oznámí nalezení tohoto pole.

Robot se musí pohybovat autonomně, nesmí se vzdálit o více jak 20 cm od černé čáry a také nesmí mít explicitně zadanou cílovou barvu a rozměry dráhy. Po umístění robota do startovního pole a zahájení programu stiskem tlačítka se ho již nelze dotýkat a musí jet autonomně. Tvar dráhy je znázorněn na obrázku 5.8, rozmístění barev je statické, tudíž lze naprogramovat trasy podle detekované barvy.

Rozbor řešení úlohy

V této úloze přibylo několik problémů, které je potřeba oproti předchozím úlohám vyřešit. Jedná se o rozeznání, ve kterém úseku dráhy se robot nachází a zkombinovat rozpoznávání černé čáry a barvy rozhodovacích a cílových políček. K řešení této úlohy se dá přistupovat několika způsoby, můžeme použít víc barevných senzorů, kdy jeden sleduje čáru, po které jede, a druhý k rozeznání dalších cest z křižovatky. Toto se dá velice snadno zpracovat, pokud máme tato čidla paralelně k jeho kolům. Na toto řešení ovšem nemáme dostatek portů, jelikož potřebujeme počítat i s prvky potřebnými k dalším úlohám. Tudíž využijeme jiné řešení, kdy se dá úsek rozeznat pomocí gyrosensoru, který je integrovaný přímo v hubu. Čteme natočení robota, který se nám vždy v každé křižovatce otáčí díky sledování hrany čáry.

Podle potřeby dalšího směřování můžeme v křižovatce reagovat a nasměrovat se do dalšího správného úseku. Pro rozpoznávání barvy je potřeba eliminovat počet dotázání na snímanou barvu oproti měření intenzity odraženého světla, které pro plynulý pohyb robota potřebujeme snímat mnohem častěji. Je potřeba balancovat mezi tím, abychom neomezovali PID regulátor sledující černou čáru a zároveň stihli zachytit najetí na barevnou plochu dřív, než z ní regulátor odvede robota.



Obrázek 5.8: Schéma dráhy pro robota.

Stavba robota

Pro tuto úlohu není třeba změna konstrukce z úlohy 5.4.1, ale zároveň nám v ničem nepřekáží změna provedená v úloze 5.4.2. Lze tedy použít obě konstrukce.

Sledování čáry

Ke sledování čáry využijeme regulátor vytvořený v první úloze 5.4.1, u kterého může být potřeba poladit hodnoty PID regulátoru pro zatočení v křižovatkách, které jsou prudší, než jsme mohli vidět v první úloze. K zatáčení na křižovatkách využijeme toho, že čáru můžeme sledovat po obou hranách. Tuto techniku využijeme tak, že zvolíme sledování hrany přiléhající směrem, kam budeme chtít na křižovatce odbočit. Směřování lze hravě změnit zase za křižovatkou na druhou hranu podle potřeby projetí další křižovatky. Pro sledování opačné hrany stačí regulátor upravit na inverzní akční zásah. Nejsložitější je projet křižovatkou rovně, řešení těchto problémů je rozebráno níže v řešení stavového automatu.

Stavový automat

Program lze s přehledem rozdělit do několika dílčích úkolů. Díky tomuto rozdělení lze použít stavový automat, kdy tento program můžeme rozdělit na níže uvedené stavy. V jednotlivých stavech uvádím pouze kódy složitějších stavů, kompletní programy se nachází v příloze.

- **Stav 0** - Hledání cílové barvy

  Robot následuje pravou hranu, dokud senzor nezaznamená nějakou výběrovou barvu. Pokud se tak stane, robot zastaví, couvne, otočí se o 180° a vyrovná se senzorem na černou čáru, aby byl připraven k dalšímu vyjetí po obou hranách černé čáry. Následně se přepne stavový automat do stavu dle příslušné nalezené cílové barvy.

```
hub.light_matrix.write('0') #zobrazeni stavu na displayi
Turn = line_follow_right()  #drzeni prave hrany
if yaw >150 and  yaw < 210: #natoceni robota k vyberu cilove
                            barvy
  if color_counter == 0:
      color = ColorSensor.get_color()
      if color == "blue" or color == "green" or color == "
                                  yellow" or color == "
                                  red": # nalezeni
                                  cilove barvy s
                                  naslednym otacenim a
                                  najetim na cernou caru
          motor_pair.stop()
          motor_pair.move_tank(3, 'cm', left_speed=-25,
                                      right_speed=-25)
                                      #couvnuti od
                                      vyberu barvy
          motor_pair.stop()
          while(not(yaw<358 and yaw> 350)): #dorovnani se k
                                      cerne care
              motor_pair.start_tank(15, -15)
              yaw = get_yaw_360()
          motor_pair.stop()
          while(not (ColorSensor.get_reflected_light() <= 50
                                      )):
              motor_pair.start_tank(5, -5)
          motor_pair.stop()
      if color == "green":  #zmena stavu automatu podle
                                  nalezene barvy
          state = 1
          hub.light_matrix.write('GREEN')
          yaw =get_yaw_360()
      if color == "yellow":
          state = 2
          hub.light_matrix.write('Yellow')
      if color == "blue":
          state = 3
```

```
            hub.light_matrix.write('BLUE')
        if color == "red":
            state = 4
            hub.light_matrix.write('RED')
motor_pair.start(steering = int(Turn))
```

- **Stav 1** - Cesta k cílové barvě - zelená

  Robot sleduje levou hranu, dokud se neotočí o 90° doleva na úhel 270°, a poté se vyrovnává doprava, dokud nevidí senzor čistě černou čáru tak, aby mohl vzápětí přepnout režim na sledování pravé hrany. Robot pokračuje, dokud nezaznamená zelenou barvu, robot zastaví a přepne se do stavu 5.

```
hub.light_matrix.write('1')#zobrazeni stavu na displayi
if first_part: #pomocna promenna pro oddeleni useku drahy se
                                    stejnym natocenim robota
    Turn = line_follow_left()
    yaw = get_yaw_360()
    if yaw <= 290 and yaw >= 200: #natoceni doleva
        first_part = False
        while(not(ColorSensor.get_reflected_light() <= 20)):
                                        #vyrovnani na caru
                                        pro zmenu smeru
            motor_pair.start_tank(10, -10)
        motor_pair.stop()

if (not first_part):  #dojezd od posledni krizovatky do
                                    ciloveho pole
    Turn = line_follow_right()
    if color_counter == 0:
        color = ColorSensor.get_color()
        if color == "green":  #nalezeni cilove barvy
            motor_pair.stop()
            state = 5
```

- **Stav 2** - Cesta k cílové barvě - žlutá

  Robot sleduje pravou hranu, dokud se nevytočí o více jak 20° doprava, což značí křižovatku. V tomto případě se natočí zpět a jede 5 cm rovně tak, aby ji přejel a mohl pokračovat dál do cíle sledováním pravé hrany. Tímto pokračuje, dokud nezaznamená cílovou barvu, zastaví a přepne se do stavu 5.

```
hub.light_matrix.write('2') #zobrazeni stavu na displayi
if yaw >30 and yaw <= 300:  #kontrola nevytoceni na
                                krizovatce s naslednym
                                vyrovnanim
    motor_pair.stop()
    while (yaw> 10):
        yaw =get_yaw_360()
        print(yaw)
        motor_pair.start_tank(-15, 15)
    motor_pair.move_tank(5, 'cm', left_speed=25, right_speed
                                    =25)
```

```
else:
    Turn = line_follow_right()
    if color_counter == 0:
        color = ColorSensor.get_color()
        if color == "yellow": #nalezeni cilove barvy
            motor_pair.stop()
            state = 5
```

- **Stav 3** - Cesta k cílové barvě - modrá

  V tomto stavu robot dělá inverzní operace ke stavu jedna, jedná se pouze o zrcadlově otočenou situaci.

- **Stav 4** - Cesta k cílové barvě - červená

  Robot drží pouze pravou hranu, dokud nezaznamená cílovou barvu, na které zastaví a přepne se do stavu 5.

- **Stav 5** - Ukončení programu

  Robot v tomto stavu ukončuje program zvukovou signalizací.

5.4.4 Uklízeč (semestrální úloha)

Zadání

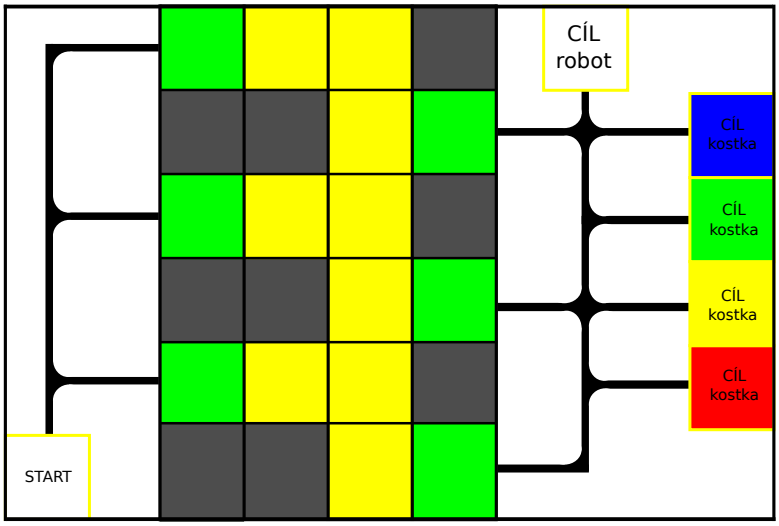
Cílem této závěrečné úlohy je nalézt v bludišti znázorněném na obrázku 5.9 barevnou kostku a tu dovézt do odkládacího pole odpovídající barvy. Na schématu šedá pole označují bariéry, žlutá místa, kde se může nacházet barevná kostka, kterou musí robot sebrat, a zelená slouží pouze k pohybu. Odkládací políčka nemají fixně dané barevné rozmístění. V okamžiku, kdy robot sebere kostku, musí na displeji ukázat, o jakou barvu se jednalo. Robot se musí během jízdy pohybovat autonomně a nesmí opustit hrací plochu. Je vhodné vzít řešení dílčích problémů z předchozích úloh a spojit je při řešení této závěrečné úlohy.

Rozbor řešení úlohy

V počátku úlohy je potřeba pomocí černé čáry vjet do bludiště a správně zkombinovat jízdu po černé čáře se sledováním stěny a práci gyrosenzoru. Je nutné přitom hlídat, zdali jsme někde na žlutých políčkách nenalezli barevnou kostku, tu je potřeba uchopit, abychom ji mohli odvést na odkládací políčko.

Při jízdě s uchycenou kostkou je potřeba brát v potaz vyšší setrvačné síly, které mohou způsobit nutnost přeladit regulátor na sledování stěny i čáry.

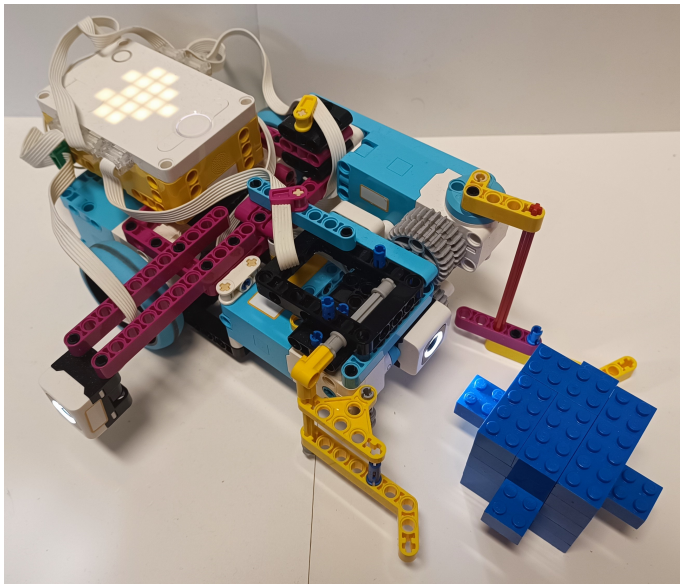
Pro průjezd bludištěm je vhodné co nejvíce využít sledování stěny, díky čemuž máme jistotu, kde se v prostoru nacházíme, jinde jsme závislí na gyrosenzoru, který, jak už jsme zmiňovali, nemá dostatečnou přesnost na nějaké větší pohyby. Pro tuto úlohu je opět vhodné vytvořit stavový automat, který bude odpovídat určitým úsekům dráhy a činnostem v nich.



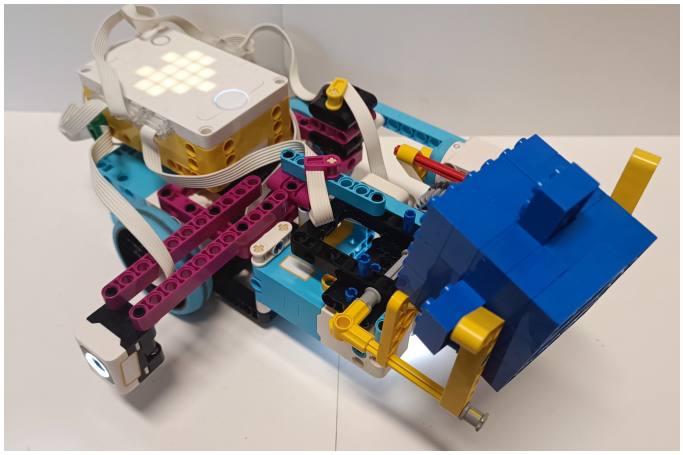
Obrázek 5.9: Schéma závěrečné úlohy.

Stavba robota

Při řešení této úlohy je třeba robota z úlohy 5.4.2 rozšířit o mechanizmus pro chytání kostky, kterou robot nalezne v bludišti. Je nutné zkombinovat motor pro otáčení barevného senzoru s mechanizmem zvedání kostky a dalším motorem s mechanizmem pro uchycení zvedané kostky. Na obrázku 5.10 je znázorněn robot pohybující se v bludiště, který hledá kostku na zvednutí, na obrázku 5.11 je robot s již zvednutou kostkou.



Obrázek 5.10: Konstrukce robota pro úlohu uklízeče při hledání kostky.

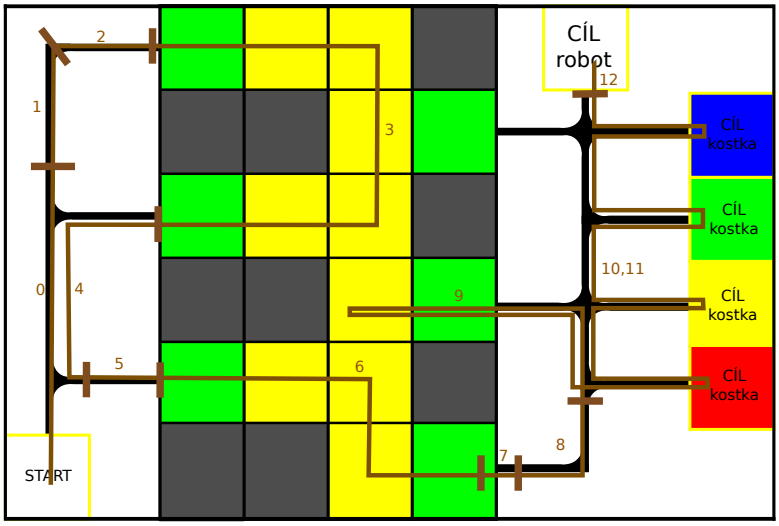


Obrázek 5.11: Konstrukce robota pro úlohu uklízeče se zvednutou kostkou.

Implementace

Tato úloha skrývá obrovské množství tras, jak lze projet bludiště, abychom nalezli schovanou kostku, odvezli ji na příslušné odkládací pole a následně robot dojel do cíle. Na obrázku 5.12 je příklad možné implementace trasy, kde bylo prioritou to, aby robot co nejvíce využíval buď sledování čáry nebo stěny. Tato trasa bude použita i v této implementaci. Dále byl využit gyrosenzor, který pomáhal k orientaci v prostoru, a další dva motory k pohybu barevného senzoru a zvedání barevné kostky. Jednotlivé kroky popíšu v následující sekci

na stavovém automatu.



Obrázek 5.12: Schéma závěrečné úlohy se zakreslenou trasou a stavy stavového automatu.

Stavový automat

V této sekci jsou popsány jednotlivé stavy stavového automatu s dílčími kroky konanými v každém z nich. Pro lepší představu o tom, jakou činnost v jednotlivém stavu robot vykonává, jsou u zakreslené hnědé trasy na obrázku 5.12 čísla stavů v příslušných úsecích dráhy.

- **Stav 0** - Projetí dvou křižovatek rovně

  V tomto výchozím stavu začíná robot svůj pohyb sledováním pravé hrany (čidlo se na začátku otočí dolů) a je resetovaný referenční yaw úhel, kdy během svého pohybu musí překonat 2 křižovatky. Křižovatku rozezná tak, že se na ní vytočí na úhel větší než 30°, ten následně vyrovná otáčením doleva do 0° a popojede rovně tak, aby projel křižovatku.

- **Stav 1** - Cesta za zatáčku směřující k hornímu vjezdu do bludiště

  V tomto stavu robot sleduje pravou hranu, dokud není splněna podmínka přechodu do dalšího stavu. Do dalšího stavu se přepne poté, když gyrosensor rozezná, že odbočil na poslední (třetí) křižovatce doprava (t.j.úhel je větší než 80°).

- **Stav 2** - Dojetí na konec černé čáry a najetí do bludiště horním vjezdem

  Robot sleduje pravou hranu, dokud se od ní nevychýlí o víc jak 20° (t.j.na úhel menší než 70°), což značí konec černé čáry. Nyní je potřeba

vjet do bludiště, kdy pro hladké chycení stěny se robot musí vyrovnat do osy bludiště (t.j.úhel 90°) a popojet několik centimetrů směrem do bludiště tak, aby senzor pro sledování stěny zaznamenal odražené světlo od stěny. Po najetí dovnitř je také zapotřebí natočit přední čidlo směrem dopředu tak, aby mohl hledat barevnou kostku.

- **Stav 3** - Sledování stěny do vyjetí z bludiště prostředním vjezdem na levé straně s případným zvednutím kostky a chycením se čáry

  Robot sleduje pomocí bočního čidla pravou stěnu, toto přeruší pouze za podmínky, že barevné čidlo na čele rozpozná barevnou kostku, nebo se robot natočí při jízdě na úhel větší než 180°, což značí konec stěny u výjezdu z bludiště. V případě nalezení barevné kostky ji pomocí upínacího mechanizmu uchytí, zapamatuje si její barvu a pokračuje sledováním stěny v tomto stavu. Jestliže nalezne konec stěny, je třeba robota otočit na místě doleva zpět na černou čáru. K tomu je zapotřebí otočit barevný senzor dolů a otáčet robota doleva, dokud přední senzor nezaznamená černou čáru (intenzita odraženého světla je větší než 50%), čímž se na ní vyrovná.

- **Stav 4** - Sledování čáry od levého prostředního vjezdu přes křižovatky až před spodní vjezd

  Robot sleduje levou hranu, dokud neprojede oběma křižovatkami doleva (t.j. úhel menší než 105°), poté se přepne do dalšího stavu.

- **Stav 5** - Najetí do bludiště spodním vjezdem, kterému předchází rozpoznání konce čáry a vyrovnání do osy bludiště

  Robot pokračuje ve sledování levé hrany až do chvíle, kdy projede obě křižovatky, na nichž zahne doleva a dojede až na konec čáry, což se pozná tak, že se úhel dostane pod 110°. Tuto výchylku poté vyrovná otáčením doleva zpět na 90° a robot popojede dopředu tak, aby čidlo na sledování stěny opět zaznamenalo nenulové odražené světlo.

- **Stav 6** - Průjezd spodní části bludiště od spodního vjezdu až do spodního výjezdu

  Na počátku tohoto kroku je zapotřebí v případě dosavadního nenalezení barevné kostky otočit senzor na čele směrem dopředu, abychom ji mohli hledat během průjezdu bludištěm. Robot se drží pravé stěny, dokud se nestočí na úhel 180°, jenž značí roh stěny a blížící se stěnu před ním, ke které se musí přiblížit a otočit se o 90° doleva. Po otočení následuje další stav.

- **Stav 7** - Výjezd z bludiště spodním výjezdem s následným chycením černé čáry

  Na počátku tohoto kroku je potřeba opět otočit čelní senzor dolů, pokud tam již není, a sledovat pravou stěnu do chvíle, kdy spodní senzor nezaznamená černou čáru. Na konci tohoto kroku je potřeba rozhodnout

podle toho, zda již robot našel barevnou kostku, v případě nalezení následuje stav 8, jinak následuje stav 9.

- **Stav 8** - Sledování čáry za první levotočivou zatáčku s následným vyrovnáním se na střed čáry

  Robot sleduje levou hranu, dokud se nenatočí pod úhel 30°, při čemž se vyrovná na střed čáry a následuje stav 10.

- **Stav 9** - Prozkoumání posledního neprozkoumaného žlutého políčka v případě, že robot stále neobjevil kostku

  Robot drží levou hranu, dokud se nenatočí na úhel $270° \pm 10°$, což značí směřování dovnitř bludiště. Dále si již hlídá pouze nepřekročení 280°, což značí konec černé čáry. Následně se dorovná zpět na úhel 270°, díky tomu se bude nacházet v ose bludiště. Nyní robot jede rovně, až do té chvíle, než narazí na barevnou kostku, která se již musí nacházet v tomto políčku.

  Následně se otočí o 180° a jede, dokud boční senzor nezaznamená stěnu, kterou následně začne sledovat až do chvíle nalezení černé čáry. Poté začne sledovat pravou hranu za první křižovatku (t.j natočení na $180° \pm 10°$), následně bude držet již levou hranu za další křižovatku (t.j natočení na $90° \pm 10°$) a následně se přepne do stavu 10.

- **Stav 10** - Hledání odkládacího pole se shodnou barvou s barvou kostky

  Robot se drží pravé hrany až do chvíle, kdy najede na pole, v němž zaznamená jednu ze čtyř možných barev kostky. Přitom počítá, kolik projel křižovatek z hlavní ulice. Na tomto poli se zastaví a v případě shody odloží kostku a následuje stav 11, případně 12, pokud se jednalo o poslední odkládací políčko. V případě neshody se tento stav opakuje, dokud nenalezne shodu.

- **Stav 11** - Průjezd všemi křižovatkami rovně s následným vyrovnáním se zpět na čáru za ní.

  Robot sleduje pravou hranu a hlídá vychýlení úhlu nad 20° s následným vyrovnáním zpět, projetím křižovatky rovně a chycením se čáry zpět. To vykonává, dokud neprojede poslední čtvrtou křižovatkou.

- **Stav 12** - Dojetí z poslední křižovatky po případném vyrovnáním úhlu do 0° do cílového pole

  Robot se vyrovná do nulového úhlu a popojede rovnou jízdou do koncového pole, kde zvukovou signalizací oznámí konec programu a ukončí ho.

5.4.5 Zhodnocení řešení úloh pomocí Spike Prime

Jak již bylo zmíněno v úvodu řešení úloh, úlohy bylo nutné adaptovat na zpracování stavebnicí Spike Prime. Její neduhy se nejvíc projevovaly při nastavení regulátorů využívající pohony a gyrosenzor. Největším problémem je jejich vůle v uložení, kdy se i po zastavení může motor otočit o několik stupňů bez otočení rotorem a zaznamenání enkodérem. A to buď vlivem setrvačnosti nebo jiných vnějších energií působících na hřídel. K tomuto neduhu se ještě přidala menší rychlost výpočetní smyčky a vzorkovací frekvence senzoru.

Z těchto důvodů bylo nutné při využívání regulátoru sledování černé čáry měnit jeho hodnoty napříč programem a bylo nutné volit mezi hladkým a pomalejším průjezdem nebo rychlostí. To stejné platilo i pro regulátor sledování stěny. Já jsem volil cestu rychlosti, tak jak je filozofie bodování úloh, u kterých chceme, aby robot projel dráhu co nejrychleji. Tento problém se promítal v zásadě do řešení všech úloh.

Dalším problémem, který se ukázal, je doba trvání některých funkcí, které zaměstnají procesor na mnohem delší dobu než ostatní, a mezitím robot jede neregulovaně dál. Jedná se například o funkce *print()*, *get_degrees_counted()*, nebo výpis více znaků na displeji pomocí *light_matrix.write()*.

Při pokusech využít k sofistikovanějšímu řešení úloh přesné orientace, využívající vlastní funkce kombinující odometrii a gyrosensor, jsem zjistil, že je velice nepřesný k řízení. Při pokusu otočení na místě o 360° se otočí pouze o 350° a tvrdí, že je opět v nulové pozici, která se nulovala před zahájením otáčení.

I přes všechny tyto neduhy se dají úlohy vyřešit pomocí jednoduchých programů a regulátorů, tak jsem je například úspěšně vyřešil já. Do budoucna by se dala nejvíc poladit asi poslední úloha a to z pohledu konstrukce robota a chytání kostky, kde by se dalo lépe vymyslet mechanizmus zvedání a otáčení barevného senzoru.

Kapitola 6

Závěr

V úvodu mé bakalářské práce jsou rozebrány vlastnosti stavebnice LEGO Spike Prime, jež je nejnovější robotická stavebnice z rodiny LEGO Education. Byly popsány hardwarové a softwarové vlastnosti této stavebnice, zdrojem byly dostupné datasheety a praktické pokusy s jednotlivými komponenty.

Další část práce porovnává tuto stavebnici s předchozí generací, která je nyní asi nejrozšířenějším zástupcem této rodiny, stavebnicí LEGO Mindstorms EV3. Hlavním výstupem tohoto srovnání je, že se na vlastnostech hardwaru i softwaru podepsalo zaměření firmy LEGO stavebnicí Spike Prime na nižší věkovou kategorii. Spike má mnohem nižší výpočetní kapacitu, což je částečně kompenzováno využitím firmwaru s mnohem nižšími nároky, který ovšem není tak stabilní a rychlý. Všechny senzory mají 10x nižší snímací frekvenci a celkově nižší přesnost. Pohony mají menší výkon a obrovské vůle v uložení os oproti přechozí generaci.

V poslední stěžejní části byly vytvořeny řešené úlohy pro předmět Roboti. Tyto úlohy byly navrženy pro stavebnici LEGO EV3, ovšem z již zmiňovaných důvodů je bylo třeba adaptovat na nižší možnosti stavebnice Spike Prime. Během řešení těchto úloh bylo zjištěno mnoho dalších vlastností, které nebyly zřetelné z parametrů. Největším problémem se jevilo rozklíčování zpomalené odezvy při použití určitých funkcí, které měly zjednodušit řešení úloh, naopak ho spíše ztížily. Jedná se například o funkce výpisu na terminál v programovacím prostředí na PC či dotázaní se na stav enkodéru motoru, které extrémně zbrzdí programovou smyčku.

Upravené úlohy se podařilo po adaptaci na Spike Prime úspěšně vyřešit a popsaný program s řešeními dílčích problémů je obsažen v této práci a příloze. Do budoucna bych pro výuku doporučil pokračování ve vývoji vlastní verze kostky, založené na verzi EV3 nebo použití jiné dostupné stavebnice na našem trhu, jelikož vlastnosti této stavebnice neumožnují smysluplně řešit sofistikovanější úlohy.

Příloha A

Literatura

[1] Amazon.com. LEGO Mindstorms EV3 Intelligent Brick . https://www.amazon.com/LEGO-Mindstorms-EV3-Intelligent-Brick/dp/B00G1L0278,. Dostupné online; vid. 18/2/2023.

[2] Amazon.com. LEGO MINDSTORMS EV3 Touch Sensor 45507. https://www.amazon.com/LEGO%C2%AE-MINDSTORMS%C2%AE-Touch-Sensor-45507/dp/B00E1PRQ48,. Dostupné online; vid. 12/2/2023.

[3] Amazon.com. LEGO Mindstorms Ev3 Ultrasonic Sensor . https://www.amazon.com/Lego-Mindstorms-Ev3-Ultrasonic-Sensor/dp/B00E1PTRAE,. Dostupné online; vid. 12/2/2023.

[4] Amazon.com. LEGO Mindstorms NXT 2.0 (8547). https://www.amazon.com/LEGO-Mindstorms-NXT-Discontinued-manufacturer/dp/B001V7RF9U,. Dostupné online; vid. 21/2/2023.

[5] Automa – časopis pro automatizační techniku, s. r. o. Co znamená PID. https://automa.cz/cz/casopis-clanky/co-znamena-pid-2003_03_28768_3811/„ 2003. Dostupné online; vid. 10/3/2023.

[6] Bre Burns. A new lease on learning with LEGO Education set 45678 SPIKE Prime [Review]. https://www.brothers-brick.com/2020/01/17/a-new-lease-on-learning-with-lego-education-set-45678-spike-prime-review/,. Dostupné online; vid.25/2/2023.

[7] EDUXE. 45601 LEGO®SPIKE ™ Prime Programovatelný hub. https://www.eduxe.cz/p/354/45601-programovatelny-hub,. Dostupné online; vid. 11/2/2023.

[8] LEGO GROUP. Can I use LEGO® SPIKE™ Essential with other LEGO robotics systems? . https://www.lego.com/en-us/service/help/LEGO_Education/spike-essential-with-other-robotics-systems-kA009000001KVc8CAG,. Dostupné online; vid. 5/2/2023.

[9] LEGO GROUP. LEGO® Education SPIKE™ App: download . https://education.lego.com/en-us/downloads/spike-app/software,. Dostupné online; vid. 7/2/2023.

[10] LEGO GROUP. LEGO® History: LEGO® Education. https://www.lego.com/cs-cz/history/articles/g-lego-education/,. Dostupné online; vid. 5/2/2023.

[11] LEGO GROUP. SPIKE™ Prime: The LEGO® Education SPIKE™ App. https://education.lego.com/en-us/teacher-resources/lego-education-spike-prime/start-here/lego-education-spike-prime-the-lego-education-spiketm-app,. Dostupné online; vid. 5/2/2023.

[12] LEGO GROUP. Lego mindstorms ev3 hardware developer kit. https://education.lego.com/en-au/product-resources/mindstorms-ev3/downloads/developer-kits„ 2013. Dostupné online; vid. 1/4/2023.

[13] LEGO GROUP. Lego Mindstorms EV3 User guide. https://www.robotworld.cz/lego-mindstorms-ev3/navody„ 2015. Dostupné online; vid. 1/4/2023.

[14] LEGO GROUP. Lego® education spike™ prime technical specifications: Technic™ large hub rechargeable battery. https://education.lego.com/en-us/product-resources/spike-prime/downloads/technical-specifications„ 2015. Dostupné online; vid. 1/4/2023.

[15] LEGO GROUP. Lego® education spike™ prime technical specifications: Technic™ technic™ force sensor. https://education.lego.com/en-us/product-resources/spike-prime/downloads/technical-specifications„ 2015. Dostupné online; vid. 11/2/2023.

[16] LEGO GROUP. Lego® education spike™ prime technical specifications: Technic™ color sensor. https://education.lego.com/en-us/product-resources/spike-prime/downloads/technical-specifications„ 2019. Dostupné online; vid. 11/2/2023.

[17] LEGO GROUP. Lego® education spike™ prime technical specifications: Technic™ distance sensor. https://education.lego.com/en-us/product-resources/spike-prime/downloads/technical-specifications„ 2019. Dostupné online; vid. 11/2/2023.

[18] LEGO GROUP. Lego® education spike™ prime technical specifications: Technic™ large angular motor. https://education.lego.com/en-us/product-resources/spike-prime/downloads/technical-specifications„ 2019. Dostupné online; vid. 11/2/2023.

[19] LEGO GROUP. Lego® education spike™ prime technical specifications: Technic™ large hub. https://education.lego.com/en-us/product-resources/spike-prime/downloads/technical-specifications„ 2019. Dostupné online; vid. 11/2/2023.

[20] LEGO GROUP. Lego® education spike™ prime technical specifications: Technic™ medium angular motor. https://education.lego.com/en-us/product-resources/spike-prime/downloads/technical-specifications„ 2019. Dostupné online; vid. 11/2/2023.

[21] Anton's MINDSTORMS Hacks. Distance Sensor Breakout board for SPIKE Prime and Robot Inventor. https://antonsmindstorms.com/product/distance-sensor-breakout-board-for-spike-prime-and-robot-inventor/,. Dostupné online; vid. 18/2/2023.

[22] Matt Hocker. A History of LEGO Education, Part 3: Mindstorms over matter [Feature]. https://www.brothers-brick.com/2020/02/03/a-history-of-lego-education-part-3-mindstorms-over-matter-feature/,. Dostupné online; vid. 21/2/2023.

[23] Jufrika. Sejarah Evolusi Lego Robotic Mindstroms dari Tahun ke Tahun. https://www.jufrika.com/2021/08/sejarah-evolusi-lego-robotic-mindstroms.html,. Dostupné online; vid. 13/2/2023.

[24] Matyho kostky. MINDSTORMS: 1998. https://matyhokostky.cz/historie/1998-2,. Dostupné online; vid. 21/2/2023.

[25] Matyho kostky. Počátky LEGO robotiky. https://matyhokostky.cz/historie/1985-2/,. Dostupné online; vid. 21/2/2023.

[26] George Robotics Limited. MicroPython. https://micropython.org/,. Dostupné online; vid. 18/2/2023.

[27] Peggy Reimers. New SPIKE App 3.2.0 Is Released by LEGO. https://blog.tcea.org/unboxing-spike-prime/,. Dostupné online; vid. 11/2/2023.

[28] Peggy Reimers. New SPIKE App 3.2.0 Is Released by LEGO. https://blog.tcea.org/lego-spike-app-3/,. Dostupné online; vid. 11/2/2023.

[29] Růžovka.cz. LEGO® Education 45602 SPIKE™ Prime Velký úhlový motor (Technic Large Angular Motor). https://ruzovka.cz/cs/2-stupen-zs-vxs/18007-lego-education-45602-spike-prime-velky-motor.html,. Dostupné online; vid. 12/2/2023.

[30] Růžovka.cz. LEGO® Education 45603 SPIKE™ Prime Střední úhlový motor (Technic Medium Angular Motor). https://ruzovka.cz/cs/2-stupen-zs-vxs/18008-lego-education-45603-spike-prime-stredni-motor.html?gclid=CjwKCAiAuaKfBhBtEiwAht6H798no1Y9_Lya1pgPmLF51JneXM02Dck8aiEKk4BRUBPtgrwXF6gLNRoCjvsQAvD_BwE,. Dostupné online; vid. 12/2/2023.

[31] Růžovka.cz. LEGO® Education 45605 SPIKE™ Prime Senzor barev (Technic Color Sensor). https://ruzovka.cz/cs/2-stupen-zs-vxs/18831-lego-education-45605-senzor-barev-technic-olor-sensor.html,. Dostupné online; vid. 12/2/2023.

[32] Růžovka.cz. LEGO® Education 45605 SPIKE™ Prime Senzor barev (Technic Color Sensor). https://ruzovka.cz/cs/2-stupen-zs-vxs/18831-lego-education-45605-senzor-barev-technic-olor-sensor.html,. Dostupné online; vid. 12/2/2023.

[33] Růžovka.cz. LEGO® Education 45606 SPIKE™ Prime Senzor síly (Technic Force Sensor). https://ruzovka.cz/cs/2-stupen-zs-vxs/18832-lego-education-45606-senzor-sily-technic-force-sensor.html,. Dostupné online; vid. 12/2/2023.

[34] Růžovka.cz. LEGO® Education Mindstorms® 45503 EV3 Servo motor střední. https://ruzovka.cz/cs/2-stupen-zs-vxs/16970-lego-mindstorms-45503-ev3-servo-motor-stredni.html,. Dostupné online; vid. 12/2/2023.

[35] Růžovka.cz. LEGO® Education Mindstorms® 45506 EV3 Color sensor. https://ruzovka.cz/cs/2-stupen-zs-vxs/16973-lego-mindstorms-45506-ev3-color-sensor.html?gclid=CjwKCAiAuaKfBhBtEiwAht6H704Wx-v5hveXZMT7PzImth1FW3i_38UJXEbW2OhOvVMh8-nv6kKPxhoCmbQQAvD_BwE,. Dostupné online; vid. 12/2/2023.

[36] Seznam.cz. LEGO Mindstorms 45502 EV3 Velký servo motor. https://www.zbozi.cz/vyrobek/lego-mindstorms-45502-ev3-velky-servo-motor/fotogalerie/,. Dostupné online; vid. 12/2/2023.

[37] Audrey Watters. Lego Mindstorms: A History of Educational Robots . http://hackeducation.com/2015/04/10/mindstorms,. Dostupné online; vid. 19/2/2023.

[38] Matěj Štětka. Tvorba nových výukových materiálů pro předměty využívající LEGO Mindstorms. https://dspace.cvut.cz/handle/10467/107201„ 2023. Diplomová práce; Dostupné online; vid. 5/3/2023.

Příloha B

Seznam digitálních příloh

B.1 Zdrojové kódy

Tento adresář obsahuje zdrojové kódy k řešeným úlohám určených pro software SPIKE Legacy. Obsahuje tyto zdrojové kódy *Sledovani_cary*, *Bludiste*, *Hledani_barvy*, *Uklizec* ve formátu .llsp.

B.2 Dráhy pro tisk

Tento adresář obsahuje návrhy drah pro tisk ve formátu .pdf v názvech odpovídajících příslušným zdrojovým kódům, jak je popsáno v jednotlivých zadáních u řešení úloh. Pro třetí úlohu jsou k dispozici dva separátní soubory odpovídající jednotlivým separátním drahám.