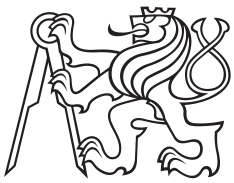
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra řídící techniky

Využití robota Lego Mindstorms - návrh a realizace speciálních projektů

Jiří Bauer
Kybernetika a robotika
bauerji3@fel.cvut.cz

Květen 2016
Vedoucí práce: Ing. Martin Hlinovský Ph.D.

České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra řídicí techniky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: **Jiří Bauer**

Studijní program: Kybernetika a robotika
Obor: Systémy a řízení

Název tématu: **Využití robota Lego Mindstorms - návrh a realizace speciálních projektů**

Pokyny pro vypracování:

1. Seznamte se s možnostmi robota Lego Mindstorms NXT a EV3 (současný stav, HW a SW vybavení).
2. Proveďte a realizujte návrh dvou speciálních projektů pro předmět B3B35RO Roboti:
"Sledování černé čáry" - úkolem robota je sledovat černou čáru na bílém podkladu s možným křížením, překážkami, tunelem, mostem a různým klesáním a stoupáním na hracím hřišti.
"Lego pinball machine" - pro reklamní účely předmětu B3B35RO Roboti realizovat pinball, kdy hráč posílá rotačním pohybem servomotoru na hrací plochu kuličky a odráží je pomocí dvojice páček tak, aby zasahoval jednotlivé cíle umístěné na stole, které mají přiděleny různé bodové hodnoty (účelem hry je získat co nejvyšší skóre). Mezi páčkami je mezera, pokud jí kulička propadne, vysílá se do hry nová.
3. Vytvořte webové stránky k realizovaným projektům (popis úloh, návrh regulátoru nebo principu činnosti, vysvětlení navrženého softwaru, fotogalerii a popřípadě návod na stavbu jednotlivých modelů).

Seznam odborné literatury:

[1] James Floyd Kelly – LEGO MINDSTORMS NXT-G programming Guide, Second Edition
[2] Daniele Benedettelli – Programming LEGO NXT Robots using NXC

Vedoucí: Ing. Martin Hlinovský, Ph.D.

Platnost zadání: do konce letního semestru 2016/2017

L.S.

prof. Ing. Michael Šebek, DrSc.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 1. 2. 2016

Poděkování / Prohlášení

V první řadě bych chtěl poděkovat vedoucímu práce Ing. Martinu Hlinovskému Ph.D. za cenné rady a připomínky při řešení projektů. Dále bych chtěl poděkovat své rodině, přítelkyni a přátelům za podporu během celého studia.

Prohlašuji, že jsem zadanou bakalářskou práci zpracoval sám s přispěním vedoucího práce a používal jsem pouze literaturu v práci uvedenou. Dále prohlašuji, že nemám námitek proti půjčování nebo zveřejňování mé bakalářské práce nebo její části se souhlasem katedry.

V Praze dne 12.5.2016

. . . . . .

Abstrakt / Abstract

Tato bakalářská práce se zabývá řešením dvou projektů použitím stavebnice LEGO Mindstorms. Projekty slouží k propagaci Robosoutěže a předmětu A3B99RO Roboti vyučovaného na ČVUT FEL v Praze. Práce má čtyři části, v první dochází k seznámení se stavebnicí a s jejími programovacími možnostmi. Ve druhé části se zabýváme tvorbou Lego Pinball Machine. Při sestavování automatu vycházíme z již vzniklých automatů, ale naším cílem je vytvořit automat zábavnější. Ve třetí části se věnujeme řešení úlohy Sledování černé čáry na bílem podkladu. Práce popisuje návrh konstrukce robotu a následně návrh regulátoru pro řízení pohybu podél čáry. V poslední části vytváříme webové stránky, kde prezentujeme řešení dvou projektů.

Klíčová slova: pinball, PID regulátor, robot, LEGO, leJOS, Java

This bachelor thesis describes solution of two projects using LEGO Mindstorms education EV3 building kit. The purpose of those projects is to advertise the Roborace and the lesson A3B99RO Robots at the CTU FEE in Prague. The thesis has four parts, in the first one we learn about the building kit and programming possibilities. In the second one we deal with creating Lego Pinball Machine. We got inspiration from some existing machines, but our goal is to create more entertaining machine. In the third part we solve the task of the line following robot. At first we construct a robot and then we desing a controller for controlling the position of the robot. The last part is about developing web pages, where we demonstrate our solutions of given two projects.

Keywords: pinball, PID controller, robot, LEGO, leJOS, Java

Title translation: Usage of the Lego Mindstorms Robot 2 - Design and realization of the special projects

Obsah

Tabulky / Obrázky

Úvod

Tato bakalářská práce se zabývá využitím robota Lego Mindstorms při výuce předmětu A3B99RO Roboti na katedře řídící techniky. Krátce se věnuje porovnáním dvou nejnovějších verzí stavebnice NXT a EV3, kde je větší důraz kladen na EV3, protože právě s její pomocí jsou řešeny zadané projekty.

První projekt je Lego pinball machine. Cílem je sestavit hrací automat, ve kterém hráč pomocí flipperů odpaluje kuličku po hracím poli do různých cílových míst za účelem získat co nejvyšší počet bodů. Pokud kulička propadne mezerou mezi flippery, je do hry vyslána znovu. Na tomto projektu je vysvětlen postup nahrání potřebného firmwaru do EV3 kostky, její programování v jazyce Java a komunikace kostek mezi sebou pomocí rozhraní bluetooth.

Druhým projektem je robot sledující černou čáru na bílem podkladu s možností křížení, překážek, tunelu, mostu a různých klesání a stoupání. Úkolem je sestavit robota ze stavebnice LEGO Mindstorms a navrhnout regulátor pro řízení pozice při pohybu podél černé čáry.

Další částí je tvorba webových stránek o projektech s jejich popisem a řešením, které budou umístěny na stránkách předmětu A3B99RO Roboti.

Obou projektů bude využito při propagaci a výuce předmětu Roboti katedry řídící techniky na elektrotechnické fakultě Českého vysokého učení technického v Praze.

Kapitola 1
LEGO Mindstorms

V této kapitole se seznámíme se stavebnicí LEGO Mindstorms. Nejprve se podíváme na nějaké základní parametry jednotlivých částí stavebnice. Druhá část potom bude o různých možnostech, jak lze inteligentní kostka programovat a jak s ní lze komunikovat.

1.1 Hardware

Stavebnicí LEGO Mindstorms NXT se již na katedře řídící techniky věnovalo několik bakalářských prací (např. [2], [3]), zaměříme se tedy spíše na novější verzi EV3 a její vylepšení oproti verzi předchozí.

1.1.1 Inteligentní kostka

Obrázek 1.1. Inteligentní kostka EV3 [8]

Nejdůležitější částí celé stavebnice je inteligentní kostka. Oproti verzi NXT je EV3 vylepšená téměř ve všech možných parametrech včetně procesoru, operační paměti, rozlišení displeje a možností rozšíření vnitřní paměti pomocí MicroSD karty. Nesmíme opomenout ani dva USB porty, kdy port USB 1.1 slouží k propojování kostek mezi sebou a port USB 2.0 ke komunikaci s počítačem, zatímco NXT byla vybavena pouze jedním portem USB 2.0. EV3 kostky je tedy možné propojovat pomocí rozhraní Bluetooth, USB a WiFi (při použití USB adaptéru), zatímco u NXT to bylo možné pouze pomocí Bluetooth. V neposlední řadě došlo ke změně v počtu výstupních portů pro připojení motorů, nově má kostka k dispozici čtyři oproti dřívějším třem. Detailnější porovnání obou kostek uvádí tabulka 1.1.

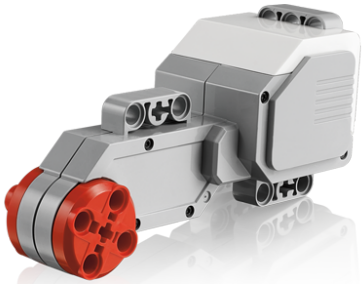
1.1.2 Servomotory

Hlavní pohonnou jednotkou sady EV3 je velký motor, který má stejné parametry jako velký motor v sadě NXT. Disponuje integrovaným rotačním senzorem s rozlišením jednoho úhlového stupně. Dokáže vyvinout rychlost až $160 - 170$ ot./min a točivý

	NXT	EV3
procesor	Atmel 32-Bit AMR 48 MHz 256 KB FLASH-RAM 64 KB RAM	ARM9 300 MHz 16 MB Flash 64 MB RAMt
vstupní porty (sensorové) výstupní porty	4 max 9600 bit/s (IIC) 4	4 max 460.8 Kbit/s (UART) 4
displej	maticový 100x64 px	maticový 198x128 px
komunikace	Bluetooth USB 2.0	Bluetooth USB 2.0 (EV3 - PC) USB 1.1 (EV3 - EV3)

Tabulka 1.1. Porovnání inteligentních kostek EV3 a NXT. [5], [6]

moment 20 Ncm. Jediný rozdíl od NXT velkého motoru je v rozmístění stavebního rozhraní, které by mělo být optimalizováno pro rychlejší a složitější stavění.



Obrázek 1.2. EV3 velký motor [8]

Úplnou novinkou je EV3 střední motor, který u verze NXT nebyl. Jak již z názvu vyplývá, jedná se o motor lehčí a menší. Také má integrovaný rotační senzor se stejnou přesností. Dosahuje maximální rychlosti $240 - 250$ ot./min a točivého momentu 8 Ncm, jedná se tedy o motor rychlejší a slabší.

Obrázek 1.3. EV3 střední motor [8]

Oba tyto motory jsou zpětně kompatibilní s NXT inteligentní kostkou, stejně tak lze ke kostce EV3 připojit motory starší verze.

1.1.3 Senzory

Asi nejzásadnější změnou u senzorů je zvýšení frekvence získávání vzorku ze 333 vzorků/s na 1000 vzorků/s a přechod na senzory digitální (výjimkou je dotykový senzor, který je analogový). Tím se ztratila zpětná kompatibilita s NXT kostkou, která komunikuje pouze s analogovými senzory.

EV3 dotykový senzor detekuje stisk a uvolnění tlačítka. Rozeznává tři stavy: stisk, uvolnění a náraz, tj. stisknutí a následné uvolnění. Právě poslední stav (náraz) je vylepšení oproti NXT dotykovému senzoru, bez této vymoženosti se ovšem lze snadno obejít detekcí po sobě následujících prvních dvou stavů.

Obrázek 1.4. EV3 senzory [8]

Dalším senzorem je EV3 barevný senzor, který lze použít ve třech různých režimech. V barevném režimu rozpoznává sedm barev a žádnou barvu (oproti šesti barvám u NXT). V režimu intenzity odraženého světla vysílá senzor ze zdroje pod objektivem červené světlo a intenzitu odraženého paprsku na škále 0 (tmavá) až 100 (světlá). Při režimu intenzity okolního světla opět využívá škálu $0 - 100$.

Natočení robotu lze měřit pomocí jednoosého gyroskopického senzoru, který detekuje úhel natočení v rozmezí $-90°$ až $90°$ s přesností $\pm 3°$. Senzor dokáže měřit také rychlost otáčení až do $440°$ za sekundu.

EV3 ultrazvukový senzor měří vzdálenost objektů před ním od 3 do 250 cm s přesností ± 1 cm (u NXT ± 3 cm). Nově je senzor vybaven světly kolem „očí“ senzoru, která indikují aktuální mód. Pokud světla svítí, je senzor v módu měření vzdálenosti, zatímco když světla blikají, je aktivní přijímací mód, kdy senzor detekuje přítomnost jiných ultrazvukových senzorů v jeho dosahu.

Novinkami jsou IR senzor a IR ovladač. IR senzor detekuje infračervené světlo odražené od pevných objektů nebo infračervené signály vyslané IR ovladačem. Senzor je možné použít v módu měření vzdálenosti (do 70cm) nebo v módu navigace, kdy senzor do vzdálenosti 2 m najde IR ovladač a odhadne vzdálenost a směr, ve kterém se ovladač nachází. IR ovladač dále samozřejmě umožňuje dálkové ovládat robota.

V neposlední řadě EV3 nabízí teplotní senzor, který měří 6.3 cm dlouhou sondou teplotu od $-20°C$ do $-120°C$ s přesností $0.1°C$.

Dále je možné použít senzory třetích stran, které jsou kompatibilní s robotickými systémy LEGO MINDSTORMS od HiTechnic, Mindsensors nebo Vernier. Více na [18].

1.2 Software

Inteligentní kostky EV3 lze opět programovat pomocí grafického prostředí vhodného především pro uživatele, kteří nemají větší zkušenosti s programováním. Další možností je využít leJOS Java Virtual Machine, která umožní programovat kostku pomocí jazyka Java. Podporu nabízí také Matlab a Simulink.

1.2.1 Grafické prostředí EV3-G

EV3-G vychází z předchozí verze NXT-G a v mnoha ohledech ji vylepšuje. Je zpětně kompatibilní a lze tedy pomocí EV3-G programovat inteligentní kostku NXT. Je ale nutné počítat s tím, že NXT nepodporuje všechny funkce nové verze (jako např. WiFi komunikaci). Programy se vytváří pomocí spojování jednotlivých programovacích bloků.

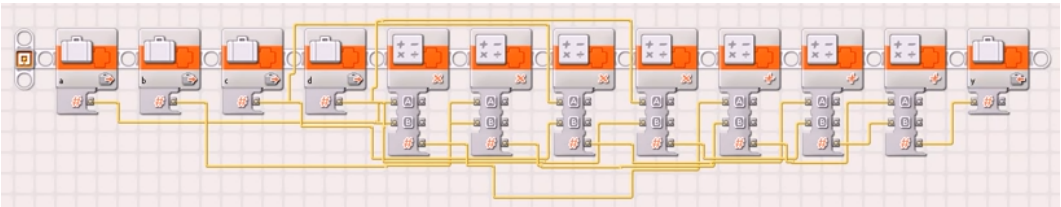
Tyto bloky se dělí do pěti skupin. První jsou akční bloky, mají zelenou barvu a ovládají akce programu. Ovládají se pomocí nich motory a zvuk, displej nebo světlo na EV3 kostce. Oranžové jsou bloky funkční. Ovládají funkce programu, například každý program začíná jedním funkčním blokem. Dále se pomocí funkčních bloků vytvářejí například cykly nebo zpoždění programu. Senzorové bloky mají barvu žlutou. Jak již jejich název napovídá, umožňují komunikovat se senzory a číst z nich hodnoty. Červené bloky jsou operační a zajišťují čtení a zápis do proměnných nebo porovnávání proměnných. Poslední skupinou bloků jsou bloky pokročilé. Značí se modrou barvou, umožňují uživateli například čtení a zápis do souborů nebo spravovat komunikaci přes bluetooth.

EV3 kostku lze programovat i na mobilních zařízeních. K tomu slouží aplikace EV3 Programmer, která obsahuje grafické programovací prostředí EV3-G. Uživatelé se ovšem musí obejít bez pokročilých a operačních bloků, které jsou přístupné pouze ve verzi na PC.

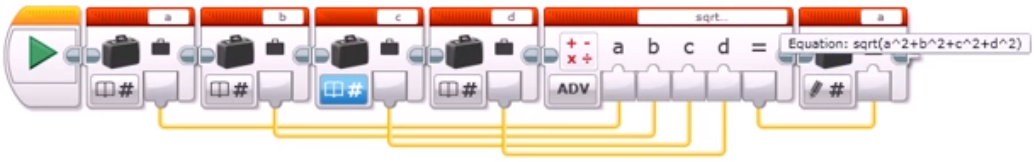
Jako příklad lze využít program pro výpočet rovnice

$$y = \sqrt{a^2 + b^2 + c^2 + d^2}. \tag{1}$$

Je zřejmé, že řešení lze v prostředí EV3-G (obrázek 1.6) zapsat mnohem kratším a srozumitelnějším způsobem nežli pomocí NXT-G (obrázek 1.5) díky matematickému bloku, ve kterém je možné zadefinovat potřebnou rovnici. V programu pomocí NXT-G je v rámci zanechání alespoň menší přehlednosti ignorována odmocnina v rovnici (1), protože pro ni není speciální blok. Další inovací je možnost psát parametry bloků (např. hodnotu proměnné) do políčka v horní části a není nutné už tedy bloky rozklikávat. Tím dojde ke zpřehlednění celého programu a urychlení nastavování parametrů.



Obrázek 1.5. Program v NXT-G [9]



Obrázek 1.6. Program v EV3-G [9]

1.2.2 leJOS EV3 JVM

Díky firmwaru leJOS (lego Java OS) lze programovat kostku v jazyce Java. Tím získáváme možnost využít objektově orientovaného programování, paralelních vláken a dalších vlastností jazyka Java. Seznam všech tříd a jejich podrobný popis je zpracován v přehledném leJOS API [13]. K podrobnějšímu seznámení s tímto způsobem programování se dostaneme v průběhu řešení speciálních úloh. Jenom pro úplnost ještě uvedeme řešení rovnosti (1) pomocí leJOS EV3.

```
public static void main(String[] args){

    int a = 1, b = 2, c = 3, d = 4;
    double y;

    y = Math.sqrt(a^2 + b^2 + c^2 + d^2);

}
```

Obrázek 1.7. Program v jazyce Java

1.2.3 MATLAB a Simulink

Podpora EV3 od Mathworks umožňuje využívat všech funkcí Matlabu se vstupy ze senzorů a výstupy zobrazit pomocí LCD displeje kostky nebo reproduktoru. Pomocí Simulinku je potom možné opět vytvořit algoritmus s využitím bloků, celý ho odsimulovat a poté nahrát do kostky a spustit. Nabízí také možnost měnit parametry modelu v Simulinku, zatímco algoritmus běží v EV3 kostce.

1.2.4 Aplikace pro mobilní zařízení

Pro chytré telefony a tablety jsou kromě již zmíněné aplikace EV3 Programmer k dispozici ještě další. Jedna z nich, Robot Commander, umožňuje využít zařízení jako dálkový ovladač k inteligentní kostce. Druhá je 3D Builder a obsahuje animované návody, jak krok po kroku sestavit pět základních začátečnických robotů.

Kapitola 2
Lego pinball

V této kapitole si nejprve řekneme něco o hře pinball a její historii [11]. Poté se už pustíme do stavby automatu samotného.

2.1 Pinball

Pinball je arkádová hra, většinou v podobě podlouhlé bedny s různými pohyblivými částmi a překážkami, ve které se pohybuje kovová kulička. Hráč pomocí dvou pohyblivých flipperů odpaluje kuličku ve snaze zamezit jí, aby propadla mezerou mezi flippery a naopak nasměrovat ji příhodným směrem, aby získal co nejvíce bodů. Body se sbírají nárazem nebo průjezdem kuličky různými cílovými místy.

Do takovéto podoby se ale pinball dostal až během posledních padesáti let. Původně vznikal jako stolní varianta her jako například bowling nebo kulečník, ve snaze dostat tuto formu zábavy do menších prostor uvnitř. Původně byl pinball založen na náhodě, hráč pouze vystřelil kuličku do hry a sledoval, jak se pohybuje po hrací ploše bez žádné další intervence. V té době tedy pinball byl jakousi formou hazardu a to vedlo k jeho zákazu.

Během třicátých let devatenáctého století došlo k elektrifikaci automatů, což přineslo automatický mechanismus pro vystřelování kuličky, odpalovací bumpery, různá světélka a zvukové efekty.

Nejzásadnější změnu ovšem přinesl automat *Humpty Dumpty* od amerického výrobce Gottlieb z roku 1947, který do hry vnesl i faktor umu hráče. Hrací pole nově obsahovalo i 6 hráčem ovládaných flipperů, kterými se snažil udržet kuličku ve hře co nejdéle.

Postupem času se většina automatů dostala do dnešní podoby, kdy je hra ovládána pomocí dvou flipperů, které jsou umístěny kolem mezery ve spodní části hracího pole. V dnešní době se výrobci automatů snaží své hry vyrábět tématicky podle nějakých filmů nebo třeba hudebních kapel.

2.2 Rozdělení sensorů a motorů

Naším úkolem je takovou hru sestavit pomocí stavebnice LEGO MINDSTORMS. Inspirace pro tuto práci vznikla v automatech nalezených na webu ([19] a [17]). K dispozici máme dvě základní soupravy LEGO® MINDSTORMS® Education 45544 EV3 a dvě soupravy doplňkových dílů 45560 EV3. Dále použijme standardní LEGO kostičky pro sestavení šasi celého automatu. Pro stavbu tedy můžeme využít dvě EV3 inteligentní kostky a 6 servomotorů (dva střední a čtyři velké). Senzory potom máme k dospozici 4 dotykové, 2 barevné, 2 ultrazvukové a 2 gyroskopické.

Nejprve se pokusíme rozdělit motory. Nejdůležitější jsou dva flippery, pro jejichž pohon použijeme velké servomotory. Dále budeme potřebovat jeden velký servomotor pro vystřelení kuličky do hry. Po propadnutí kuličky mezi flippery ji bude potřeba dopravit zpět ke startéru, k čemuž se pokusíme vytvořit dopravní pás. K jeho pohonu

by měl stačit střední servomotor. Ve středu hrací plochy bychom chtěli vytvořit výtah (podobně jako u [19]), k čemuž budeme potřebovat velký servomotor. Zbývá na konec ještě jeden střední servomotor, který bude pohánět jakýsi kolotoč, který bude odpalovat kuličku různými směry a dělat tak hru trochu zajímavější.

Se senzory začneme opět u flipperů. Pro jejich ovládání použijeme dvou dotykových senzorů, pomocí nichž vytvoříme dvě tlačítka pro snadné ovládání. Detekovat konec hry, tj. propadnutí kuličky mimo hrací pole budeme ultrazvukovým senzorem. Pro výtah využijeme další dotykový senzor, kdy budeme detekovat spadnutí kuličky do výtahu ve spodní pozici. Na hracím poli vytvoříme ještě další dvě místa pro získávání bodů a to opět pomocí dotykového senzoru (body tedy budou uděleny za náraz do překážky) a dále pomocí barevného senzoru budeme detekovat průjezd kuličky místem na ploše. Celkem nám dvě kostky umožňují připojení až osmi senzorů, ale my si vystačíme s těmito šesti.

Nazvěme jednu kostku kostkou hlavní a druhou kostkou řídící. Hlavní kostka bude komunikovat s prvky, na kterých hráč získává skóre a zároveň jej bude zobrazovat na LCD displeji. Kostka řídící bude řídit ostatní prvky jako flippery, odpalovací mechanizmus aj. Komunikaci mezi kostkami budeme realizovat pomocí rozhraní bluetooth. Díky tomu, že se skóre bude manipulovat pouze kostka hlavní, jediné informace, které budeme muset mezi kostkami přenášet, budou indikace začátku/konce hry.

2.3 Příprava kostky

Než se pustíme do stavby jednotlivých částí automatu a programování obslužného programu, musíme si připravit software v počítači pro komunikaci s kostkou a následně do kostky nahrát nejnovější verzi leJOS firmwaru, aby bylo možné kostku programovat v jazyce Java. Budeme postupovat podle návodu na leJOS wiki [14].

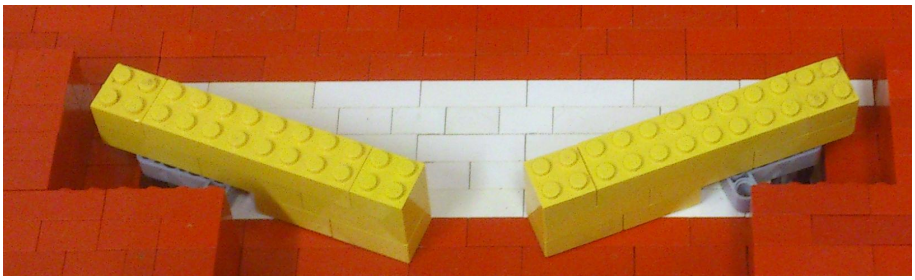
Firma leJOS nabízí podporu pro vývojářské prostředí Eclipse [12] prostřednictvím Eclipse pluginu. EV3 kostka má slot pro micro SD karty, použijeme proto bootovací SD kartu (ovšem pozor, karty SDXC nejsou podporovány a kapacita karty musí být do 32GB). Nejprve ji musíme naformátovat jako FAT32, následně na ni nahrát Oracle JRE (Java Runtime Environment) [16] a konečně i nejnovější verzi firmwaru. Konkrétně se jedná o verzi leJOS EV3 0.9.1-beta z listopadu 2015, která kromě oprav několika bugů starších verzí přináší hlavně lepší možnosti propojování kostek do PAN (Personal Area Network) pomocí bluetooth, WiFi nebo USB. Po vložení SD karty do kostky a zapnutí se začne kostka konfigurovat. Paměť na kartě se rozdělí na dva oddíly a nabootuje se nový operační systém. V menu už potom jenom najdeme bluetooth a spárujeme kostku s počítačem.

2.4 Flippery

Asi bude logickým krokem začít u flipperů, které celou hru ovlivňují nejvíce.

V automatech [17] a [19], ve kterých jsme se inspirovali, jsou flippery ovládány mechanicky pohybem ovládací páčky. My se ovšem pokusíme ovládat je pomocí servomotorů a tlačítek, která je budou spouštět, čímž se více přiblížíme reálným pinballovým automatům. Postavíme je z LEGO kostiček, pomocí osičky připojíme k velkým motorům a ty připevníme k hrací ploše. Jejich vzájemnou vzdálenost upravíme tak, aby mezi nimi byla v dolní poloze mezera přibližně dvakrát větší než je průměr kuličky (2cm), tedy 4cm. Naším cílem bude, aby se po zmáčknutí tlačítka dostal flipper do horní pozice a setrval v ní, pokud bude tlačítko stále zmáčknuto. Po jeho uvolnění se flipper vrátí

zpět do dolní pozice. Rychlost pohybu musí být dostatečná, abychom dokázali odpálit kuličku do patřičné vzdálenosti a zároveň se budeme snažit minimalizovat překmit a dobu ustálení.



Obrázek 2.1. Flippery

Než se pustíme do programování regulátoru pro pohyb flipperů, podíváme se do leJOS API [13]. Pro velké servomotory lze využít dvou základních objektů a instancí třídy *EV3LargeRegulatedMotor* nebo *UnregulatedMotor*. Třída *EV3LargeRegulatedMotor* se jeví jako lepší varianta, protože obsahuje metody pro přímé řízení pozice motoru s přesností na jeden úhlový stupeň.

Po spuštění programu zjišťujeme, že regulátor implementovaný v třídě *EV3LargeRegulatedMotor* není nikterak přesný, při nastavení rychlosti dostatečné k odpalu kuličky už je překmit a hlavně doba ustálení příliš vysoká a pokud zmáčkneme tlačítko pro ovládání flipperu několikrát rychle za sebou, začne se otáčet nekontrolovatelně i do pozic, kam by se vůbec neměl dostat. Pokusíme se tedy využít třídy *UnregulatedMotor* a navrhnout regulátor vlastní.

Jelikož neznáme parametry flipperů, použijeme pro návrh PID regulátoru Ziegler-Nicholsonovu metodu [4]. Zavedeme zápornou zpětnou vazbu a budeme zvyšovat přímovazební zesílení, dokud nezačne flipper ustáleně kmitat. Poté podle zesílení a periody kmitu vypočteme jednotlivé konstanty regulátoru. Změřili jsme stejnosměrné zesílení $k_u = 7.5$ a periodu kmitání $P_u = 0.5s$, pomocí nichž vypočteme jednotlivé parametry $k_P = 4.5$, $k_I = 18$ a $k_D = 0.094$ pro PID regulátor s přenosem

$$\frac{U(s)}{E(s)} = k_P + \frac{k_I}{s} + k_D s. \tag{1}$$

Takto navržený regulátor se jeví trochu lépe než při použití třídy *EV3LargeRegulatedMotor*, ale stále nedosahuje parametrů, kterých bychom si přáli. Pokusíme se tedy navrhnout PID regulátor ještě jedním způsobem a to heuristicky pomocí poučky pana doktora Zdeňka Huráka *„když-je-to-pomalé-zvyš-P-a-když-je-tam-chyba-zvyš-I-a-když-to-kmitá-zvyš-D“*.

Touto metodou jsme se dopracovali k nejlepším výsledkům pro hodnoty regulátoru $k_P = 7$, $k_I = 0$ a $k_D = 8$. Můžeme si tedy povšimnout, že jsme schopni dosáhnout nulové odchylky od referenčního úhlu i s nulovou integrační složkou, tedy pouze pomocí PD regulátoru. Takto ovládané flippery už se chovají podle našich představ, a proto tento regulátor použijeme v našem programu. Jedná se o třídy *LeftFlipper* a *RightFlipper* vnořené do třídy *PinballController*. Vnořené třídy využijeme z důvodu použití sdílených proměnných pro všechna vlákna uvnitř třídy *PinballController*. Celý program pro obsluhu řídící kostky je přiložen jako příloha A.2.

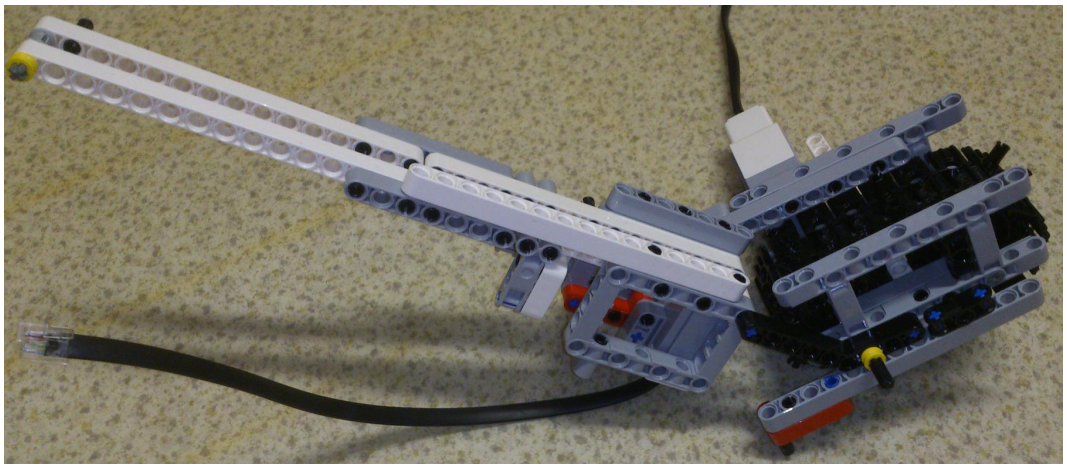
2.5 Mechanizmus pro odpálení kuličky

Po propadnutí kuličky mezi flippery musíme tuto skutečnost detekovat a kuličku dopravit k mechanizmu, který ji bude schopen odpálit zpět do hry. Začněme tedy de-

tekcí propadnutí kuličky. Použijeme ultrazvukový senzor, který umístíme pod flippery. Budeme pomocí něj měřit vzdálenost od protější stěny automatu. Pokud tudy potom kulička projede, naměřená vzdálenost se změnší a tím dojde k detekci. Jak se ale ukazuje při testu funkčnosti, senzor není v dostatečné vzdálenosti od flipperů a reaguje i na jejich pohyb v dolní fázi. Pokud by měl tento systém fungovat, museli bychom posunout ultrazvukový senzor dále od flipperů, ale tím by došlo k dalšímu zbytečnému prodloužení celého automatu. Je tedy potřeba najít jiný způsob.

Mohli bychom se pokusit hlídat průjez kuličky ve stejném místě pomocí barevného senzoru. Využili bychom ho v režimu měření intenzity odraženého světla za předpokladu, že intenzita odraženého světla od projíždějící kuličky by měla být vyšší než intenzita světla odraženého od protější stěny automatu. Vzdálenost od protější stěny je ale poměrně velká a když kulička projíždí v její blízkosti (tedy ve větší vzdálenosti od barevného senzoru), změní se intenzita jenom nepatrně, a proto toto také není vhodný způsob, jak zabezpečit rozpoznání vypadnutí kuličky z hracího pole.

Ještě se nám nabízí jedno místo, kde bychom mohli rozpoznávat pohyb kuličky. Pokud vytvoříme v mechanizmu pro dopravu kuličky k odpalovači koridor, kterým se bude kulička bezpečně pohybovat, můžeme právě do něj umístit barevný senzor a tím hlídat konec hry. Jednoduše tedy postavíme nakloněnou rovinu s mezerou uprostřed, kterou se bude směrem dolů pohybovat kulička, a na jejím konci umístíme barevný senzor, který bude namířen směrem vzhůru. Pokud nad ním projede kulička, zvýší se intenzita odraženého světla. Tato změna bude dostatečně veliká vzhledem k umístění celého mechanizmu přímo pod krytem automatu.



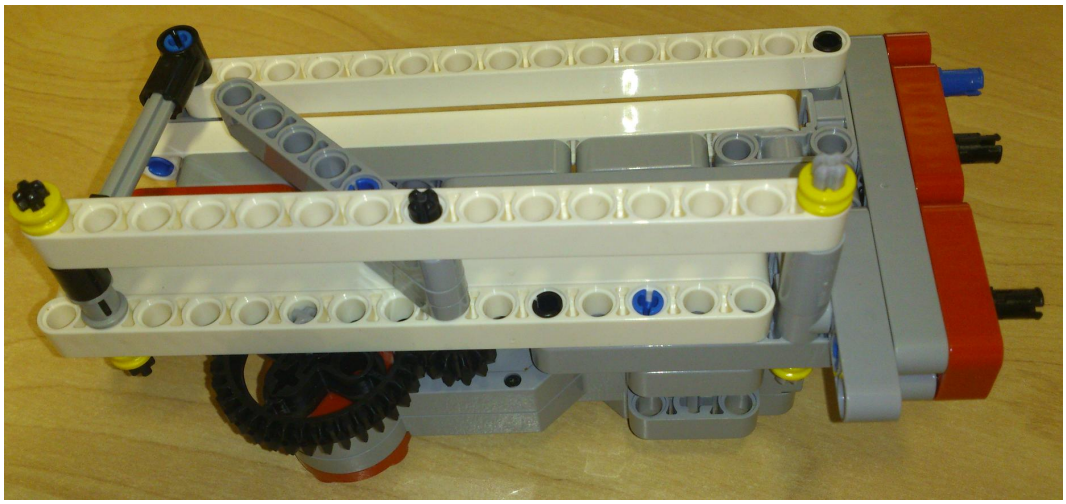
Obrázek 2.2. Dopravní mechanizmus

Za senzor umístíme dopravní pás poháněný středním servomotorem, který kuličku vyveze zpět nahoru na úroveň hrací plochy k odpalovacímu mechanizmu. Aby byl pás schopen nabrat dole kuličku, vybavíme ho ostny, které vytvoříme připojením spojovacích čepů k pásu. Pokud je pravidelně rozmístíme po celé délce pásu, mělo by být zařízení schopno kuličku nabrat a bez problému ji dovézt až nahoru. Celé zařízení pro dopravu kuličky je na obrázku 2.2.

V programu představuje obsluhu dopravního pásu třída *Belt*. Její funkce je velice jednoduchá. Když rozpozná kuličku před pásem, začne jím otáčet a tím dopraví kuličku k odpalovači.

Pro vystřelení kuličky do hry použijeme velký EV3 servomotor, který bude pohánět páčku a ta kuličku odpálí. Abychom ušetřili trochu místa, použijeme pro odpálení poměrně krátkou páčku. Proto je nutné využít ozubených koleček pro zpřevodování výkonu motoru, aby byl vůbec schopen kuličku odpálit s dostatečnou razancí.

Při konstrukci musíme dbát na to, aby se kulička bezpečně a plynule přemístila z pásu k odpalovači. Také musíme dát pozor, aby se ostny pásu nezadrhávaly o konstrukci, což by mohlo způsobit zaseknutí celého mechanizmu. Musíme počítat s tím, že kulička může k odpalovači přijet i shora, když ze hry vypadne zpět do vstupního koridoru. Proto je nutné zabezpečit, aby kulička zůstala v příhodné pozici pro odpálení nejen při pohybu z dopravníhu pásu, ale i při pohybu ze hry. Toho docílíme mezerou v podložce v místě, ve kterém je pozice kuličky nejvhodnější pro odpal. Kulička se nám tedy v této pozici zastaví, čímž zajistíme její bezpečné odpálení.



Obrázek 2.3. Mechanizmus na vystřelní kuličky do hry

Odpálení kuličky je spouštěno stiskem prostředního tlačítka řídící kostky. O ovládání celého vystřelovacího zařízení se stará třída *BrickButtons*. Za jistých okolností může nastat situace, kdy dopravní pás nedokáže nabrat kuličku a dovést ji k odpalovači z důvodu opakovaného přeskakování kuličky přes ostny na pásu. Pokud k tomu dojde (což zjistíme, když nedojde k vystřelení kuličky po stisku tlačíka), můžeme spustit celý mechanizmus od pohybu dopravního pásu až k vystřelení kuličky znovu stiskem tlačítka *dolů* na řídící kostce. Z tohoto důvodu potřebujeme, aby měla dvě vlákna přístup k ovládání motoru pro pohon pásu. Musíme tedy zajistit, aby v každém okamžiku k motoru přistupovalo pouze jedno vlákno a po dokončení jeho obsluhy ho uvolnilo.

Celá konstrukce se jeví poměrně pevně, připojíme k ní tedy stojnou nohu celého automatu. Do nohy ještě umístíme ovládací tlačítko pro pravý flipper. Pomocí páky docílíme zmáčknutí dotykového senzoru, pokud hráč zmáčkne tlačítko směrem vzhůru, a tím uvede flipper do pohybu.

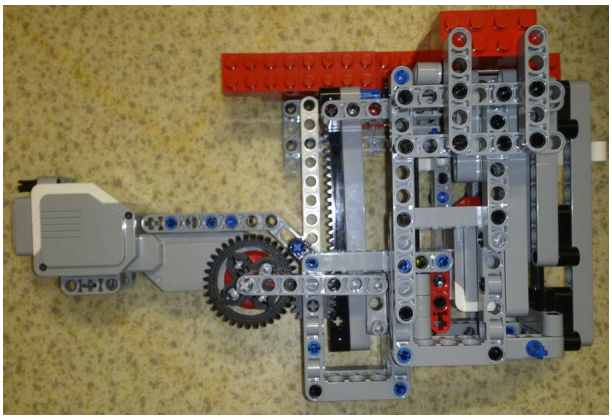
2.6 Výtah

Hlavním místem pro sbírání bodů by asi měl být výtah ve střední části hracího pole. Naším cílem bude vytvořit výtah, který po průjezdu kuličky bránou v levém horním rohu stolu sjede do dolní pozice, kde setrvá po dobu přibližně deseti vteřin. Pokud do něj spadne kulička, přičtou se body, výtah vyjede zpět nahoru a vrátí kuličku zpět do hry. Pokud do něj kulička nespadne, vrátí se po uplynutí dané doby do horní polohy bez přičtení bodů.

Výtah budeme realizovat pomocí pohyblivé plošinky, která bude poháněna velkým servomotorem. V dolní pozici bude plošinka položena na dotykový senzor, který bude

stlačen, pokud na ni dopadne kulička. Musíme tedy dbát na to, aby byla plošina dostatečně lehká. V opačném případě by totiž mohla dotykový senzor aktivovat při svém pohybu dolů.

Při spojování LEGO kostiček se součástkami LEGO Mindstorms zjišťujeme, že jejich vzájemná kompatibilita není úplně dokonalá. Musíme si tedy se spojováním patřičně pohrát a najít tu správnou kombinaci dílků tak, aby se pohyblivé části nedřely o ty pevné, ale aby zase nevznikaly nevhodně veliké mezery.



Obrázek 2.4. Výtah

Bránu pro aktivaci výtahu vytvoříme pomocí barevného senzoru, kdy budeme opět měřit intenzitu odraženého paprsku (při průjezdu kuličky se intenzita zvýší). Pro snížení vlivu rušení okolním světlem postavíme kolem senzoru bránu. Tuto bránu umístíme do levého horního rohu hrací plochy, aby ji nebylo příliš snadné kuličkou trefit.

Průjezdná brána je v programu reprezentována třídou *DriveThrough* a funguje jednoduše tak, že po průjezdu kuličky nastaví sdílenou logickou proměnnou *goDown* na hodnotu logické 1 po dobu přibližně deseti vteřin (není to úplně přesných deset vteřin kvůli nedokonale spolehlivé synchronizaci vláken). Pokud je hodnota *goDown* již v logické 1 a opět dojde k detekci průjezdu kuličky, začne se 10 vteřin počítat opět od začátku. Po průjezdu kuličky se přičte ke skóre 5 bodů.

Výtah je reprezentován třídou *Elevator* a reaguje na hodnotu proměnné *goDown*. Pokud je v logické 1, sjede do dolní polohy a zpět nahoru vyjede, pokud dotykový senzor rozpozná dotyk kuličky, nebo vyprší daná doba a hodnota *goDown* se změní na logickou 0. Po trefení výtahu ve spodní pozici se ke skóre přičte 10 bodů. Celý program pro obsluhu hlavní kostky je v příloze A.1.

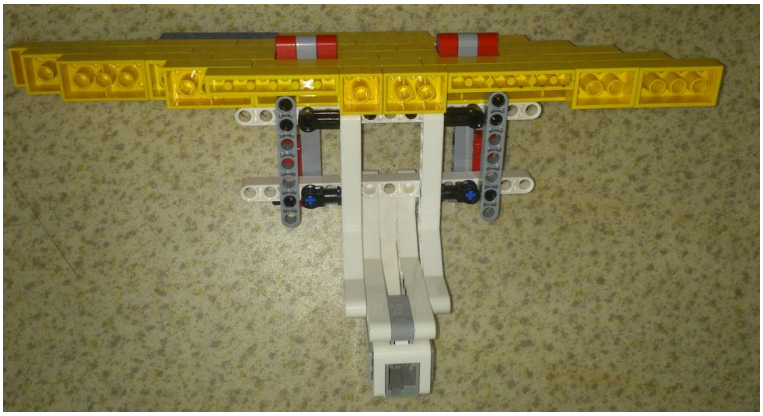
Protože je konstrukce výtahu nejvyšší částí (tj. nejvíce přesahuje pod hrací plochu), je nutné postavit ji patřičně robustně, aby ji bylo možné použít jako jeden z nosných bodů celého automatu.

2.7 Nárazníky

Jako další cílové místo pro získávání bodů vytvoříme dva nárazníky v horní části stolu. Bude se jednat o páčky, které budou nad povrch hrací desky přesahovat tak, aby je bylo možné trefit kuličkou. Obě páčky budou připojeny k jednomu dotykovému senzoru, který bude detekovat náraz.

Páková konstrukce zařízení vyžaduje, aby byly nárazníky trefeny s velkou razancí, jinak nebude dotykový senzor aktivován. Ve výsledku to je vítaná věc, jelikož klade větší nároky na hráčovu přesnost. Nárazníky umístíme nahoru, až k zadnímu mantinelu tak,

aby za nimi již nebylo místo pro kuličku. Pokud by se za ně kulička dostala, snadno by se za nimi zasekla, což by narušovalo plynulost hry.



Obrázek 2.5. Nárazníky

V obslužném programu se jedná o třídu *Bumper*. Jde o cyklické čtení hodnot z dotykového senzoru. Pokud dojde k jeho aktivaci tj. nárazu kuličky, dojde k přičtení tří bodů ke skóre a kostka vydá krátký zvuk jako signalizaci zásahu.

2.8 Centrifuga

Hru se ještě pokusíme trochu urychlit a znesnadnit rotujícím objektem, který bude mít za úkol odpalovat kuličku do různých směrů. Bude se jednat o čtyři ramena připojená ke střednímu servomotoru. Abychom do hry zanesli trochu nevyzpytatelnosti, necháme generovat náhodné hodnoty pro rychlost centrifugy a dobu otáčení danou rychlostí. Pouze rychlost omezíme polovinou maximální rychlosti středního servomotoru, protože při rychlosti vyšší už motor vydává poněkud rušivý zvuk. Dobu otáčení danou rychostí omezíme deseti vteřinami.

Centrifugu reprezentuje třída *Centrifuge* a pouze nastaví motoru danou rychlost a po uplynutí času vygeneruje rychlost novou.

Zařízení umístíme do pravé horní části hrací plochy ke vstupnímu koridoru, takže může docházet k situacím, kdy vystřelená kulička do hry bude ihned odražena zpět k odpalovači nebo naopak bude její pohyb urychlen podle toho, jakým směrem a jakou rychlostí se bude centrifuga zrovna otáčet.

2.9 Bumpery

Po vzoru hry *3D Pinball Space Cadet*, kterou Microsoft instaloval spolu s operačním systémem Windows 95 - Windows XP, se pokusíme po stranách automatu hned nad flippery vytvořit dva bumpery. Ve *Space Cadet* bumpery kuličku po nárazu odrážely poměrně důrazně opačným směrem. Jejich funkci nahradíme jednoduše použitím gumičky, kterou natáhneme kolem LEGO kostiček, které necháme vyčnívat nad úroveň hrací plochy. Po nárazu do gumičky je kulička skutečně odražena do opačného směru, i když už ne s takovou razancí.

Opět je potřeba dodržet sklon na všech částech bumperu, aby na nich nedocházelo k zaseknutí kuličky. Dále se musí dbát na umístění gumiček do dostatečné výšky, aby je kulička nepřeskočila a neuvízla mezi nimi.

2.10 Zobrazování skóre

Skóre budeme zobrazovat na hlavní kostce, která ovládá části, na kterých se získávají body. Kostku umístíme na levou stranu automatu vedle flipperů a zabudujeme ji do těla automatu. leJOS nabízí zobrazování na displeji ve třech velikostech písma, ale ani ta největší velikost není pro zobrazení skóre dostatečná. Nezbývá nám tedy nic jiného, než vytvořit číslice vlastní.

Vzhledem k rozlišení displeje 198x128 pixelů a velikosti přírůstků skóre (3, 5 a 10 bodů) se jeví jako nejlepší možnost vytvořit displej se třemi a půl digitu. Takže budeme schopni zobrazit maximální hodnotu 1999, což by mělo na drtivou většinu her stačit. Celé číslice budou mít velikost 198x50 pixelů (výška x šířka) a jednička na začátku 198x20 pixelů. Jednotlivé číslice vytvoříme v textovém dokumentu, uložíme je do domovského adresáře na paměťovou kartu hlavní kostky a při spuštění programu podle nich vytvoříme pole s hodnotami pixelů (1 pro vykreslení pixelu, 0 pro nevykreslení) pro maticový displej.

Obrázek 2.6. Zobrazení skóre na kostce

O zobrazování skóre se stará třída *Display*. Po vytvoření instance této třídy dojde k načtení číslic a nejvyššího nahraného skóre z textových souborů. Potom už se jenom v průběhu celé hry zobrazuje aktuální skóre. Nutnost zobrazit jinou hodnotu je indikována logickou proměnnou *scoreToChange*, pomocí které jednotlivé prvky automatu dávají najevo, že navýšily skóre. Po konci hry se nahrané skóre porovná s uloženým dosavadním nejvyšším. Pokud je nahrané skóre vyšší, zahraje kostka vítěznou melodii a nové skóre uloží, v opačeném případě zazní bzučák jako indikace konce neúspěšné hry.

Pro vymazání nejvyššího skóre slouží program *DeleteScore*, který je potřeba spustit na kostce místo standardního řídícího programu.

Díky umístění hlavní kostky v levém spodním rohu hrací plochy se nabízí možnost využít ji jako poměrně pevnou součástku pro připojení opěrného bodu pro stojnou nohu celého automatu. Tím získáme již třetí (po jedné na výtahu a jedné na startovacím mechanizmu), čímž bychom měli zajistit dostatečnou stabilitu. Podobně jako do nohy pod odpalovačem i do této umístíme tlačítko pro ovládání flipperu, tentokráte levého.

2.11 Komunikace kostek

Komunikace mezi kostkami po rozhraní bluetooth je díky nejnovější verzi firmwaru leJOS EV3 0.9.1-beta poměrně jednoduchá. Při nastavení PAN budeme postupovat dle návodu [10].

Nejprve nastavíme hlavní kostku, která bude sloužit jako přístupový bod naší sítě. V menu *PAN* vybereme volbu *Access Point*, dále již není potřeba jiné hodnoty nastavovat, IP adresa je pro přístupové body defaultně 10.0.1.1. Na řídící kostce vybereme v menu *PAN* položku *BT Client*. Dalším krokem je vybrání kostky, ke které se má připojit. Ze seznamu již spárovaných zařízení vybereme hlavní kostku (pokud v seznamu není, musíme je nejprve spárovat). Dále je nutné nastavit IP adresu, protože pomocí ní budeme

vytvářet soket. Zvolme tedy adresu 10.0.1.3. Dále je vhodné zaškrtnout položku *Persist*, čímž dovolíme kostce, aby se pokoušela k hlavní kostce periodicky připojovat a ne jenom při jejím zapnutí (nebude tedy nutné důsledně zapínat hlavní kostku s předstihem před řídící).

V hlavní kostce obstarává komunikaci třída *CommunicationMaster*. Nejprve pomocí vzdáleného přístupu spustí program na řídící kostce a vyčká na potvrzení komunikace. Následně už jenom čte hodnoty z datového streamu, popřípadě do něj zapíše, pokud je program vypnut, tzn. dá druhé kostce najevo, že se má také vypnout.

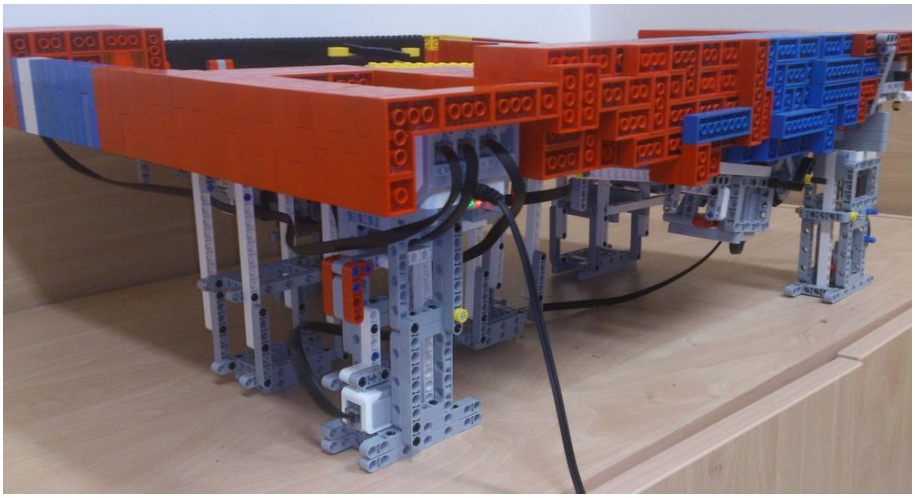
V řídící kostce potom jde o třídu *CommunicationSlave*. Ta zajistí po potvrzení navázání komunikace zápis do datového streamu, pokud kulička propadla nebo byla opět vystřelena do hry. Dále kontroluje datový stream, zda nedošlo k ukončení komunikace ze strany řídící kostky.

K nenásilnému vypnutí programu slouží tlačíko *Zpět* na hlavní kostce. Jeho stisknutí způsobí případné dokončení potřebných činností a ukončení všech běžících vláken. Není tedy vhodné ukončovat program současným stiskem tlačítek *Dolů* a *Enter*, jak je u inteligentních kostek zvykem, protože by například mohl výtah zůstat v dolní pozici a to by při opětovném spuštění způsobilo problémy.

2.12 Kompletace automatu

Když pomocí LEGO kostiček spojíme všechny vytvořené části dohromady, zjišťujeme, že celá konstrukce není příliš robustní a že ve středu hrací plochy dochází ke zlomu. Je tedy nutné automat nějakým způsobem ve střední části vystužit nebo přistavit další stojné nohy. Ukazuje se, že po doplnění tří do míst kde docházelo ke zlomu celou situaci vyřeší.

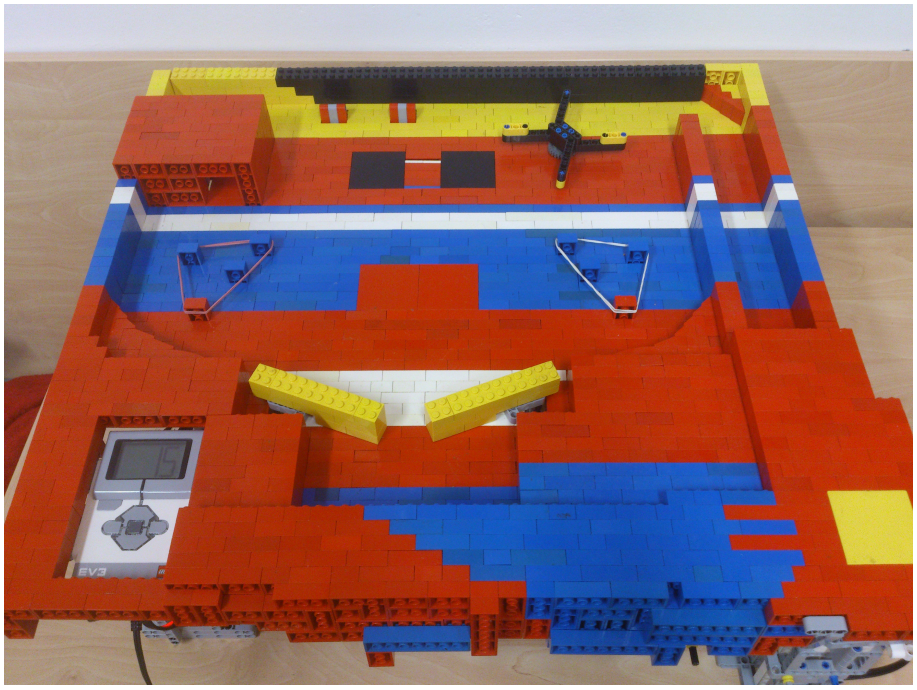
V tuto chvíli tedy stojí automat na šesti nohách, což mu dodává potřebnou stabilitu. Menší problém ovšem nastává při jeho přemisťování. Pokud je jej nutné nadzdvihnout, je vhodné podepírat ho ve více než dvou bodech, stále totiž hrozí zlomení v půli.



Obrázek 2.7. Celá konstrukce automatu

Po kompletaci dostává automat výslednou podobu (obr. 2.8). Hráč automat ovládá z pozice jako je vyobrazen na obrázku. Ovládací páčky flipperů jsou pro snadnou manipulaci umístěny po stranách automatu ve spodní části. Při hře může dojít k jemnému zadrhnutí kuličky v některých částech (např. o hranku u výtahu) z důvodu nedokonalé kompatibility stavebnice MINDSTORMS s LEGO kostičkami. V takovém případě

je nutné pohybem celého automatu kuličku uvolnit, což nakonec může umožnit hráči výraznější zážitek ze hry.



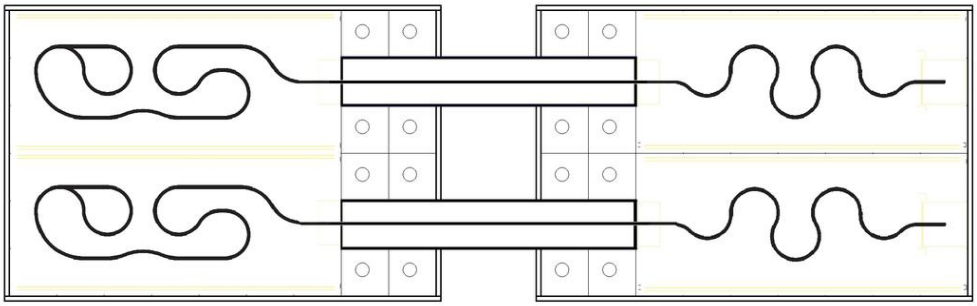
Obrázek 2.8. Automat shora

Po několika testovacích hrách se ukazuje, že obava z nízkého maximálního skóre kvůli omezené zobrazovací schopnosti displeje kostky, byla zbytečná. Mezera mezi flippery je dostatečně velká na to, aby hráč musel vyvinout patřičnou snahu k udržení kuličky ve hře. Také pozice výtahu přímo nad mezerou mezi flippery příliš nenahrává snadnému průběhu hry. Pokud totiž výtah vyveze kuličku nahoru, začne hned přirozeně padat přímo do mezery, čemuž lze zamezit patřičným pohybem celého automatu.

Tyto skutečnosti tedy dělají hru složitou a snad i zajímavou, a proto by mohlo být dosažení maximálního skóre velkou výzvou.

Kapitola 3
Sledování černé čáry

Druhou úlohou, kterou se v této práci budeme zabývat je Sledování černé čáry. Naším úkolem je sestavit robota, který bude sledovat černou čáru na bílém podkladu s možným křížením, překážkami, tunelem, mostem a různým klesáním a stoupáním na hracím hřišti.



Obrázek 3.1. Uspořádání soutěžní plochy

Úloha vychází ze zadání Robosoutěže 2016 pro Základní školy [15]. Soutěžní plocha je uspořádána dle obrázku 3.1 a 3.2. Robot může být do startovní pozice umístěn ručně a na pokyn startéra spuštěn stiskem tlačítka. Dále již musí robot pokračovat sám bez vnější pomoci (ovládání robota pomocí hlasu, bluetooth či jiných komunikačních kanálů není dovoleno).

Obrázek 3.2. Fotografie soutěžní plochy

Ke kostrukci robota lze využít pouze dílů z přidělených sestav (v našem případě se jedná o Základní soupravu LEGO® MINDSTORMS® Education 45544 EV3 a soupravu doplňkových dílů 45560 EV3). Dílky musí držet pohromadě pouze pomocí standardních

spojovacích prvků LEGO. Robot nesmí mít půdorys větší než 28cm (délka) x 28cm (šířka). Není dovoleno používat kluzný podvozek. Za kluzný podvozek se považuje takový, jehož libovolná část se po pojezdové ploše neodvaluje, ale klouže.

Robot musí černou čáru sledovat po celou dobu z jedné strany, zprava či zleva. Robot se nesmí od sledované čáry odchýlit o více jak 200 mm a nesmí si zkrátit cestu. Cílem je, aby robot projel celou dráhu (tzn. odstartoval, putoval celou dobu podél černé čáry, přes most, a znovu se vrátil na startovní pozici, kde jízda končí) do 90 vteřin.

3.1 Konstrukce robota

Robot se bude skládat z jednoduché konstrukce obsahující dvě kolečka připevněná k velkým servomotorům, jednoho volně se otáčejícího kolečka v zadní části, barevného senzoru pro sledování čáry a samozřejmě inteligentní EV3 kostky.

Při konstrukci robota budeme vycházet ze zkušenosti, kterou jsme získali při řešení podobné úlohy v rámci předmětu A3B99RO Roboti. Tato zkušenost nám říká, že pokud umístíme barevný senzor do osy otáčení (tj. mezi pohonné kolečka), pohyb robota se bude řídit složitěji, než pokud senzor umístíme před osu otáčení. Naproti tomu je ale jasné, že když bude senzor před osou otáčení (nebo i za), bude se při přejíždění zlomových ploch mostu senzor oddalovat a přibližovat k podložce, a tím se značně zkreslí senzorem snímaná hodnota.

Vzhledem ke zmíněnému rozporu se nejprve pokusíme sestavit robota se senzorem mezi koly (v ose otáčení). Pokud poté nebudeme schopni navrhnout regulátor, díky kterému by byl robot schopen zvládnout celou trasu v zadaném čase, konstrukci předěláme a senzor umístíme od osy dále.

Obrázek 3.3. Podvozek robota

Budeme se snažit vytvořit konstrukci co možná nejjednodušší a nejlehčí, protože při pohybu do mostu by každý gram navíc mohl způsobit pomalejší pohyb. Nejdříve sestavíme robustní podvozek (obrázek 3.3) a poté na něj připevníme kostku, spojíme ji se senzorem a motory. Velikost koleček volíme střední.

3.2 Návrh regulátoru

Našim cílem je navrhnout PID regulátor, který bude schopen zajistit, aby se robot podél černé čáry pohyboval plynule, dostatečně rychle ale hlavně bezpečně, tzn. aby se od čáry neodchýlil.

Pro řízení robota slouží program *LineFollowingRobot* (příloha A.3). Třída *MyBrick* obsahuje metodu *calibrate()*, která zajistí kalibraci referenční hodnoty pro sledování podle aktuálně naměřených hodnot černé čáry a bílého podkladu barevným senzorem v režimu měření intenzity odraženého světla. Jako referenční hodnotu volíme průměrnou hodnotu z naměřené intenzity odraženého světla od čáry a od bílého podkladu. Budeme tedy chtít sledovat přesně rozmezí mezi čárou a podkladem.

Metoda *countValue(double value)* vrací výstupní hodnotu PID regulátoru. Metoda *run()* potom cyklicky čte hodnoty z barevného senzoru a nastavuje potřebné hodnoty výkonu motorům podle výstupu PID regulátoru. K počátečnímu výkonu jednoho motoru je přičten výstup regulátoru s kladným znaménkem, zatímco k výkonu druhého motoru je přičten se znaménkem záporným. Právě volba tohoto znaménka určuje, z jaké strany bude robot čáru sledovat. V našem případě sledujeme čáru zprava.

Máme tedy k dispozici čtyři proměnné, jejichž hodnoty budeme hledat. Jedná se o tři složky PID regulátoru (proporcionální, integrační a derivační) a počáteční výkon motorů, který bude nastaven, pokud bude výstup regulátoru nulový, tedy pokud pojede robot přesně po hranici čáry.

Při návrhu PID regulátoru budeme opět postupovat heuristickou metodou. Nejprve budeme zvyšovat proporcionální složku regulátoru, dokud nedosáhneme dostatečné rychlosti pohybu robota. Nesmíme ale přesáhnout hraniční hodnotu, při které už bude robot nekontrolovaně kmitat, až se od černé čáry odchýlí. Integrální složka PID regulátoru slouží ke snížení ustálené odchylky. Pokud tedy nebude robot sledovat hranici černé čáry, lze to napravit zvýšením integrální složky. Zvýšením derivační složky potom lze docílit snížení překmitu a tím i plynulejšího pohybu.

Obrázek 3.4. Robot sledující černou čáru

Dodržováním výše uvedeného postupu docházíme k závěru, že tímto způsobem nejsme schopni navrhnout PID regulátor pro robota se senzorem v ose otáčení tak, abychom splnili stanovený cíl, projet dráhu během minuty a půl. Potvrdil se tedy náš

původní předpoklad, že nedokážeme uregulovat pohyb robota se senzorem mezi koly. Posuneme tedy barevný senzor více dopředu a navrhneme regulátor znovu.

Konstrukce robota po posunutí senzoru před osu otáčení je zobrazen na obrázku 3.4. Při návrhu regulátoru jsme se dostali k PID regulátoru s parametry $k_P = 1.4$, $k_I = 0$ a $k_D = 28$. Nejlepších výsedků dosahujeme, pokud volíme počáteční výkon motorů rovný 60% maximálního výkonu. Pro dosažení co možná nejvyšší rychlosti po rovných úsecích dráhy ještě nastavme maximální výkon na oba motory, pokud je odchylka od referenční hodnoty malá.

Potvrdil se nám i druhý předpoklad, že pokud umístíme barevný senzor před osu otáčení robota, nastane problém při najetí robota na zlomová místa u mostu. Senzor se na nich oddálí nebo naopak přiblíží k podložce a hodnoty, které naměří, jsou potom zkresleny. Tato skutečnost se projevuje mírným rozkmitáním robota při průjezdu přes most.

I přesto ale dokáže robot s uvedeným regulátorem i s počáteční kalibrací ujet celou trasu v časovém limitu.

Kapitola 4
Tvorba webových stránek

V této části se budeme zabývat tvorbou webových stránek k realizovaným projektům. Webové stránky budou součástí webu Roboti [15], který vytvořil Pavel Trojánek v rámci své bakalářské práce [3] v roce 2009. Tento web slouží k internetové prezentaci předmětu A3B99RO Roboti a Robosoutěže pořádané katedrou řídící techniky, katedrou měření a katedrou kybernetiky na FEL ČVUT v Praze.

Stránky mají čtyři části. Hlavičku, menu, patičku hlavní část. Naše stránky budou umístěny v záložce *Projekty* spolu s projekty LEGO vzducholoď, Inverzní kyvadlo a Rubikova kostka, které vznikly během prací [2] a [1]. Protože naším úkolem je vytvořit pouze jednotlivé stránky, omezíme se na psaní php skriptu pro hlavní část stránek. Budeme vycházet ze skriptů stránek již vzniklých, abychom dodrželi určitou koncepci webu.

Již vytvořené stránky dodržují rozdělení textu do částí Úvod, Konstrukce, Popis funkce programu, Instalace programu a Odkazy. Budeme se toho tedy držet také. Web je primárně určen pro mladší studenty s cílem vzbudit v nich zájem o robotiku a techniku obecně. Nemá tedy smysl v textu zbytečně podrobně a technicky popisovat řešení projektů. Pokusíme se text doplnit obrázky a videi pro zpestření obsahu.

Vytvořené skripty jsou přiloženy v příloze A.4. Není nutné se jimi v této práci nějak více zabývat, proces tvorby stránek je dostatečně podrobně popsán v [3].

Závěr

Během práce došlo nejprve k seznámení se s hardwarovým vybavením, které jsme měli k dispozici pro řešení dvou projektů. Porovnali jsme sadu LEGO Mindstorms Education EV3 se staršími verzemi a zjistili, co nabízí nového. Dále jsme probrali různé možnosti programování robotické stavebnice pomocí různých programovacích prostředích.

Ve druhé části jsme se věnovali realizaci Lego pinball machine. Postupně jsme navrhnuli a zkonstruovali celý automat a vytvořili k němu i obslužné programy. Nejzajímavější částí této úlohy byl návrh regulátoru pro řízení pohybu dvou flipperů. Nejlepších výsledků jsme dosáhli použitím PD regulátoru. Nakonec jsme se dostali k funkčnímu automatu, který ale nabízí ještě několik možností k vylepšení. V některých částech hrací plochy může docházet k zaseknutí kuličky, což by mohlo být vyřešeno jemným upravením konstrukce daných částí.

V další části práce jsme řešili úlohu sledování černé čáry na bílém podkladu s mostem a otočkou. Jako klíčovou částí se ukázala poloha barevného senzoru na robotickém vozítku. Pohyb jsme řídili opět pomocí PD regulátoru, který jsme navrhli heuristickou metodou. Námi navržený robot již projíždí dráhu s maximální rychlostí, ale mohlo by ještě dojít ke zrychlení, pokud bychom použili větší kolečka. Protože jsme ale dosáhli výsledného času, který splňuje limit, už jsme se do této úpravy nemuseli pouštět.

V poslední části jsme vytvořili webové stránky, kde prezentujeme navržená řešení úloh. Při jejich tvorbě jsme vycházeli z již existujících stránek.

Literatura

[1] KIRSCHNER, Filip. *Model vzducholodi s využitím stavebnic LEGO Mindstorms Education EV3*. Praha, 2014. Katedra řídící techniky, Elektrotechnická fakulta, České vysoké učení technické v Praze. Vedoucí práce Ing. Martin Hlinovský Ph.D.

[2] BĚLÍK, Tomáš. *Využití Lega Mindstorms*. Praha, 2010. Bakalářská práce. ČVUT FEL Praha.

[3] TROJÁNEK, Petr. *Využití robota LEGO MINDSTORMS při výuce*. Praha, 2009. Bakalářská práce. ČVUT FEL Praha.

[4] Ziegler-Nichols Tuning of PID Regulators. FRANKLIN, POWELL a EMAMI-NAEINI. *Feedback Control of Dynamics Systems*. 5. Prentice Hall, USA, 2005, s. 221-224.

[5] LEGO GROUP. *Lego Mindstorms EV3 User Guide*. 2013.

[6] *Lego Mindstorms NXT User Guide*. LEGO Group, 2006.

[7] KELLY, James F. *LEGO mindstorms NXT-G programming guide*. 2nd ed. New York: Distributed to the book trade worldwide by Springer-Verlag, c2010. ISBN 978-143-0229-766.

[8] *LEGO® MINDSTORMS® Education* [online]. the LEGO group, 2016 [cit. 2016-05-22]. Dostupné z:
`education.lego.com`

[9] FLYING PENGUINS. Lego EV3 vs NXT-G Math Blocks. In: *Youtube* [online]. 2013 [cit. 2016-04-04]. Dostupné z:
`https://www.youtube.com/watch?v=V16jV1dBmjU`

[10] GLOOMYANDY. PAN Configuration. In: *LeJOS News: NEWS, IN DEPTH ARTICLES, HOW TO'S & FEATURED PROJECTS* [online]. [cit. 2016-04-27]. Dostupné z:
`https://lejosnews.wordpress.com/2015/02/11/pan-configuration/`

[11] Pinball: History. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2016-04-12]. Dostupné z:
`https://en.wikipedia.org/wiki/Pinball`

[12] THE ECLIPSE FOUNDATION. *Eclipse* [online]. 2016 [cit. 2016-04-04]. Dostupné z:
`https://eclipse.org/`

[13] *LeJOS EV3 API* [online]. leJOS - developers, 2013 [cit. 2016-04-05]. Dostupné z:
`http://www.lejos.org/ev3/docs/`

[14] *LeJOS EV3 wiki* [online]. leJOS - developers, 2016 [cit. 2016-04-04]. Dostupné z:
`https://sourceforge.net/p/lejos/wiki/Home/`

[15] *Roboti* [online]. Praha: Katedra řídící techniky, ČVUT FEL v Praze, 2009 [cit. 2016-05-10]. Dostupné z:
`http://www.robosoutez.cz`

[16] JAVA for LEGO MINDSTORMS EV3. *Oracle* [online]. Oracle America, Inc., 2016 [cit. 2016-04-04]. Dostupné z:
http://www.oracle.com/technetwork/java/embedded/downloads/avaseemeddedev3-1982511.html

[17] LEGO - Pinball Machine V4 *Mindstorms EV3*. *YOUTUBE* [online]. Brickfreak79, 2014 [cit. 2016-04-07]. Dostupné z:
https://www.youtube.com/watch?v=vPTTAanSZ9s

[18] LEGO MINDSTORMS Education EV3 pro základní, střední, a vysoké školy. *EDUXE s.r.o.* [online]. Velké Pavlovice: EDUXE s.r.o., 2016 [cit. 2016-04-04]. Dostupné z:
http://www.eduxe.cz/les/robotika-ev3/

[19] Lego pinball machine. *YOUTUBE* [online]. Superlegosam, 2013 [cit. 2016-04-07]. Dostupné z:
https://www.youtube.com/watch?v=t_bSWFrsbEM

Příloha A
Obsah přiloženého CD

A.1 PinballMain

Adresář PinballMain obsahuje program pro řízení hlavní kostky pinballu a jeho zdrojové kódy.

A.2 PinballController

Adresář PinballController obsahuje program pro obsluhu řídící kostky pinballu a jeho zdrojové kódy.

A.3 LineFollowingRobot

Adresář LineFollowingRobot obsahuje program pro obsluhu robota sledujícího černou čáru a jeho zdrojové kódy.

A.4 Web

Adresář Web obsahuje zdrojové kódy php skriptů webových stránek, složku s obrázky a složku se soubory ke stažení ze stránek.

A.5 Video

Tento adresář obsahuje videa demonstrující funkčnost realizací obou projektů.

A.6 BP-Bauer.pdf

Soubor BP-Bauer.pdf je elektronická verze tohoto dokumentu.