ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta elektrotechnická

Katedra řídící techniky

Využití robota Lego Mindstorms
Návrh a realizace speciálních úloh

Usage of the Lego Minstorms Robots
Design and realization of the special tasks

Bakalářská práce

Studijní program: Elektrotechnika a informatika strukturovaný bakalářský
Studijní obor: Kybernetika a měření

Vedoucí práce: Ing. Martin Hlinovský Ph.D.

Tomáš Bělík

Praha 2010

Poděkování

Děkuji členům rodiny za podporu ve studiu a Ing. Martinu Hlinovskému Ph.D. vedoucímu bakalářské práce za spolupráci.

Prohlášení

Prohlašuji, že jsem svou bakalářskou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

V Praze, dne 21. 5. 2010 .....

.....
podpis

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra řídicí techniky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: **Tomáš Bělík**

Studijní program: Elektrotechnika a informatika (bakalářský), strukturovaný
Obor: Kybernetika a měření

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

Pokyny pro vypracování:

1. Seznamte se s možnostmi robota Lego Mindstorms (současný stav, HW a SW vybavení).
2. Nastudujte si možnostmi programování robota Lego Mindstorms.
3. Proveďte a realizujte návrh dvou speciálních úloh pro předmět A3B99RO Roboti - inverzní kyvadlo (v libovolném programovacím prostředí) a skládání Rubikovy kostky (též v libovolném programovacím prostředí).
4. Vytvořte webový stránky k realizovaným úlohám (popis úloh, návrh regulátoru nebo principu činnosti, vysvětlení navrženého softwaru, fotogalerii a popřípadě návod na stavbu jednotlivých modelů).

Seznam odborné literatury:

Dodá vedoucí práce

Vedoucí: Ing. Martin Hlinovský, Ph.D.

Platnost zadání: do konce zimního semestru 2010/2011

prof. Ing. Michael Šebek, DrSc.
vedoucí katedry

L.S.

doc. Ing. Boris Šimák, CSc.
děkan

V Praze dne 21. 10. 2009

Abstrakt

Práce se zabývá současným stavem hardwarového a softwarového vybavení stavebnice Lego Mindstorms, možnostmi jejího programování, realizací dvou speciálních úloh(inverzní kyvadlo a skládání Rubikovy kostky) a tvorbou webové stránky k realizovaným úlohám.

Abstraction

The thesis deals with the current state of hardware and software equipment of Lego Mindstorms, possibilities of its programming, the implementation of two special tasks(inverted pendulum and solving Rubik's cube) and creation of web pages to implemented tasks.

Obsah

1 ROBOT LEGO MINDSTORMS 8

1.1 STAVEBNICE LEGO MINDSTORMS 8
1.2 OSOBNÍ ZKUŠENOSTI Z VÝSTAVY AMPER 9
1.3 HARDWAROVÉ VYBAVENÍ 9
1.3.1 Programovatelná kostka 9
1.3.2 Servomotory 15
1.3.3 Senzory 16
1.4 SOFTWAROVÉ VYBAVENÍ 24
1.4.1 Standardní firmware lego 24
1.4.2 Firmware leJOS 25
1.4.3 Firmware nxtOSEK 25

2 PROGRAMOVÁNÍ ROBOTA LEGO 26

2.1 ÚVOD 26
2.2 PŘEHLED PROGRAMOVACÍCH JAZYKŮ 26
2.3 NXT-G 27
2.3.1 Program HelloWorld v NXT-G 27
2.4 NBC/NXC 28
2.4.1 Vývojové prostředí BricxCC 28
2.4.2 Program HelloWorld v NXC 28
2.5 LEJOS NXJ 29
2.5.1 Výhody LeJOS NXJ 29
2.5.2 Popis instalace 30
2.5.3 Program HelloWorld v NXJ 32

3 REALIZACE DVOU SPECIÁLNÍCH ÚLOH 33

3.1 ÚLOHA INVERZNÍ KYVADLO 33
3.1.1 Konstrukce 33
3.1.2 Vstupy a výstupy 34
3.1.3 Identifikace 34
3.1.4 Návrh regulátoru 37
3.1.5 Programové řešení úlohy 39
3.1.6 Program ovládající kyvadlo 43
3.1.7 Závěr k úloze inverzní kyvadlo 44
3.2 ÚLOHA RUBIKOVA KOSTKA 45
3.2.1 Hlavolam Rubikova kostka 45
3.2.2 Konstrukce 46
3.2.3 Programové řešení úlohy 48
3.2.4 Závěr k úloze Rubikova kostka 51

4 WEBOVÉ STRÁNKY K REALIZOVANÝM ÚLOHÁM 53

4.1 OBSAH 53
4.2 ÚVOD 53
4.3 ZPRACOVÁNÍ 53
4.4 ZÁVĚR K WEBOVÝM STRÁNKÁM 54

5 ZÁVĚR 55

POUŽITÁ LITERATURA 56
SEZNAM OBRÁZKŮ 59
SEZNAM TABULEK 59
OBSAH ELEKTRONICKÉ PŘÍLOHY 59

ÚVOD

Tato bakalářská práce pojednává o současných hardwarových a softwarových možnostech stavebnice Lego Mindstorms. Rozebírá problematiku programování stavebnice především pro volně dostupná programovací prostředí. Dále se zabývá realizací dvou speciálních úloh pro účely předmětu Roboti, který je vyučovaný na Katedře řídící techniky, Českého vysokého učení v Praze. První z úloh je inverzní kyvadlo, což je dvoukolový robot podobný komerčně známému dopravnímu prostředku Segway. Druhá úloha se zabývá robotem, který je schopen samostatně vyřešit hlavolam Rubikovy kostky. V poslední kapitole je popsána tvorba webových stránek k realizovaným úlohám. Tyto webové stránky budou umístěny na webu vytvořeném pro již zmiňovaný předmět Roboti v záložce s projekty.

Řešené úlohy se opírají o již dříve realizované projekty. Úloha Rubikovy kostky byla zpracována v roce 2008 Hansem Anderssonem, ale při běhu programu docházelo k častým chybám, proto musela být jeho práce upravena a vylepšena v této bakalářské práci. Úloha také byla rozšířena o možnost použití dalšího barevného senzoru, který byl vyvinut pro stavebnici Mindstorms v roce 2009. Úlohou inverzního kyvadla stabilizovaného pomocí gyroskopického senzoru se zabývalo již několik tvůrců, kteří vytvořili různé konstrukce i programové řešení robota. Namátkou jmenuji roboty StandAlone, GELWay, NXTway-GS, Marvin, NXT segway a další. Prvním impulsem k řešení této úlohy byl robot StandAlone, který byl regulovaný programem běžícím v počítači, ale řešení této úlohy se opírá o práce na robotech GELWay a NXTway-GS.

Toto téma bakalářské práce jsem si vybral, protože mě zajímala jak práce se stavebnicí Lego Mindstorms tak i řešené úlohy, které pro mě byly velkou motivací.

1 ROBOT LEGO MINDSTORMS

1.1 Stavebnice Lego Mindstorms

Je programovatelnou stavebnicí vyvíjenou společností Lego. Jde o geniální produkt, v němž se pojí vynalézavost, logika a hravost. Své si v Legu Mindstorms najde každý, kdo touží po realizaci vlastních projektů. Jednou z hlavních předností stavebnice je téměř neomezená variabilita robota. Základním prvkem stavebnice je inteligentní programovatelná kostka(kostka NXT), ke které je v základu možnost připojit až tři servomotory a čtyři senzory viz. obr 1. Současný trh nabízí širokou škálu senzorů, které jsou plně kompatibilní se stavebnicí. Kostka též obsahuje bluetooth zařízení, které jí umožňuje bezdrátovou komunikaci. Ze stavebnice je tak možno sestavit různá vozidla, chodící roboty, hlídací roboty apod. Se stavebnici je dodáván grafický programovací systém NXT-G. S jeho pomocí systémem drag&drop naprogramujete snadno a názorně jednoduché, ale i složitější úlohy pro robota. Protože NXT-G je pouze jedním z mnoha podporovaných programovacích jazyků, stává se Lego Mindstorms zajímavým produktem nejen pro děti, ale i pro jejich otce, a dokonce i velmi dobrou učební pomůckou, ze které jsou nadšení jak žáci, tak i učitelé. První verze stavebnice byla uvedena do prodeje pod názvem Robotics Invention System v roce 1998 a byla vyvinuta ve spolupráci s laboratoří MIT. Další verze byla vydaná v roce 2006 již pod názvem Lego Mindstorms NXT. Hlavním rozdílem oproti původní verzi je možnost připojení prostřednictvím bluetooth. Poslední verze Lego Mindstorms NXT 2.0 byla uvedena na trh v roce 2009 a v základu obsahuje nový barevný senzor a plně podporuje operace s desetinnou čárkou.

obr. 1 Programovatelná kostka a základní příslušenství

1.2 Osobní zkušenosti z výstavy Amper

V rámci prezentace nového předmětu A3B99RO Roboti vyučovaného na katedře řízení jsem se zúčastnil výstavy Amper 2010. Hlavní částí prezentace bylo seznámení s organizací předmětu a Robo soutěží 09. Jako názorná ukázka práce v předmětu sloužili především tři různí roboti sledující čáru. Ostatní roboti Inverzní kyvadlo, Pavouk, Rubikova kostka a robot balancující na kuličce sloužili spíš k prezentaci celé katedry řízení. Osobně mě velmi překvapil velký zájem o robota skládající Rubikovu kostku. Dle reakcí návštěvníků si myslím, že v nich prezentace vyvolala dobré pocity a zájem jak o stavebnici Lego Mindstorms, tak i o předmět, který ji využívá ve výuce.

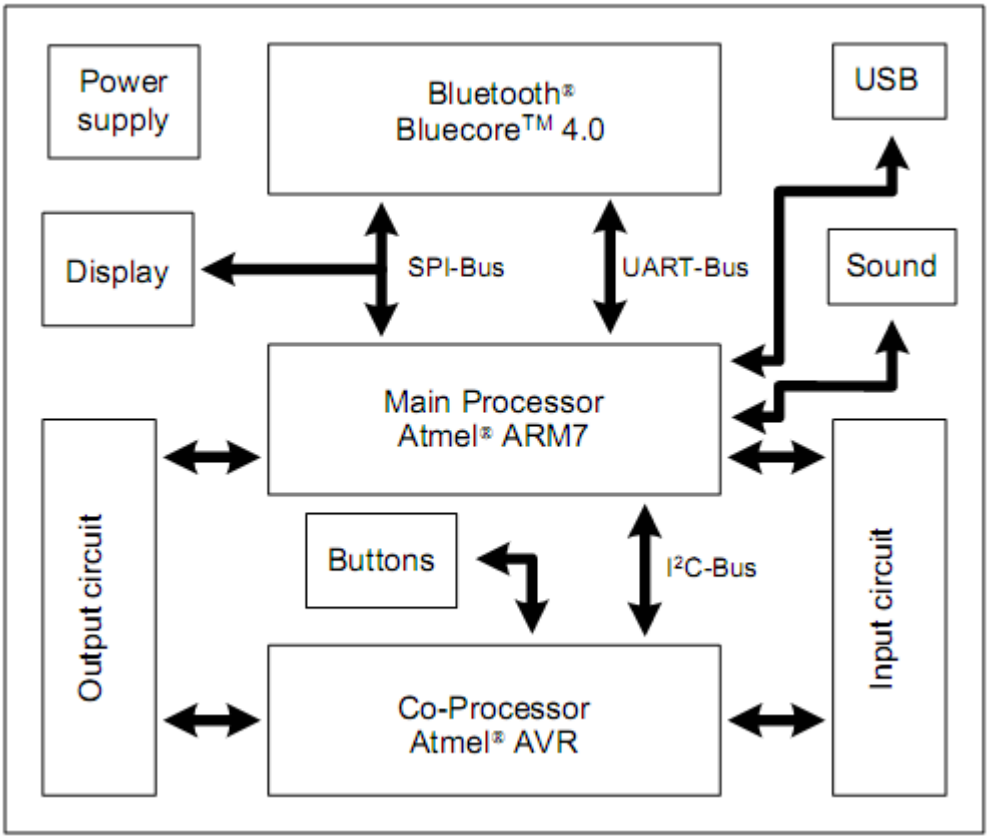
1.3 Hardwarové vybavení

Hardwarové vybavení lze rozdělit do tří kategorii:

- Programovatelná kostka
- Servomotory
- Senzory

1.3.1 Programovatelná kostka

Programovatelná kostka, která je hlavním prvkem stavebnice, je osazená 32-bit mikroprocesorem s 256KB flash pamětí. Přes výstupní porty umožňuje připojení až tří servomotorů a přes čtyři vstupní porty čtyři libovolné senzory pomocí kabelů typu RJ12. Kostku můžeme propojit s počítačem přes usb port nebo bezdrátově přes bluetooth. K ovládání kostky slouží čtyři tlačítka(enter, escape, doleva , doprava). K zobrazování dat má kostka monochromatický maticový display. K přehrávání tónů slouží zabudovaný reproduktor. Kostku lze napájet šesti AA bateriemi nebo nabíjecím akumulátorem.



obr. 2 Schéma zapojení inteligentní kostky

1.3.1.1 Mikroprocesory

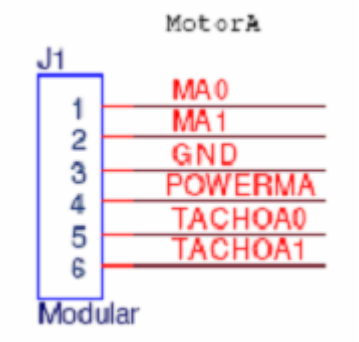
Kostka je osazena dvěma mikroprocesory. Hlavním mikroprocesorem je Atmel 32-bit ARM processor, AT91SAM7S256 s 256KB flash pamětí a 64KB RAM běžící na frekvenci 48MHz.

Druhý mikroprocesor zastává funkci koprocesoru a je jím Atmel 8-bit AVR processor, ATmega48 s 4KB flash pamětí a 512B RAM běžící na frekvenci 8Mhz.

Jak je vidět ze schématu na obr.2, mikroprocesory k vzájemné komunikaci využívají I^2C protokol. Oba pracují jak s výstupy, tak vstupy kostky. Hlavní mikroprocesor navíc ovládá zobrazovací jednotku, zvukový výstup a usb i bezdrátové bluetooth připojení.

1.3.1.2 Výstupní porty

Kostka má tři výstupní porty, přes které je možno řídit připojené servomotory. Porty mají šest pinů, z nichž dva slouží pro vysílání, dva pro příjem a dva k napájení motorů.



obr. 3 Schéma výstupního portu

MA0 a MA1 jsou řídícími PWM signály, které jsou řízeny vnitřním ovladačem motoru, který může kontinuálně dodávat 700mA na každý výstupní port. Ovladač motoru má ochranu proti přehřátí. Pokud bude do motoru dlouho dodáván vysoký výkon, ovladač automaticky hodnotu výstupu sníží.

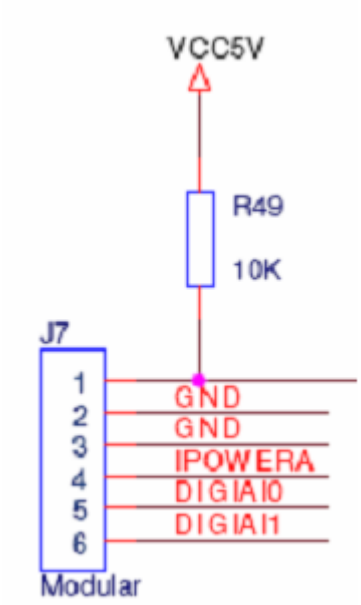
GND na pinu 3 je nulou pro napájení motorů.

POWERMA slouží k dodávání provozního napětí o velikosti 4,3V na výstup. Je vnitřně propojen se všemi výstupními i vstupními porty.

TACHOA0 a TACHOA1 jsou signály obsahující pulsy ze senzoru otáček, který je umístěn uvnitř motoru. Mezi vstupními piny(5,6) a vstupními piny hlavního mikroprocesoru ARM7 je Schmittův klopný obvod. Ze signálů lze takto počítat počty pulsů a zjistit tak, zda se motor otáčí vpřed nebo vzad.

1.3.1.3 Vstupní porty

Kostka má čtyři vstupní porty, přes které je možno připojit libovolné variace kompatibilních senzorů. Stejně jako výstupní porty mají i vstupní šest pinů viz. obr. 4.



obr. 4 Schéma vstupního portu

ANA(pin 1) je analogovým vstupním pinem, který je připojen na 10-bitový A/D převodník. Je zároveň připojen ke zdroji proudu a slouží tak k napájení aktivních senzorů. Vstupní analogové signály jsou v A/D převodníku vzorkovány vzorkovací frekvencí f_{vz}=333Hz, protože analogové senzory musí být napájeny alespoň po dobu 3ms, aby byly schopné měření.

IPOWERA má stejné vlastnosti jako pin POWERMA popsaný v kapitole 1.3.1.2.

DIGAI0 a DIGIAI1 jsou vstupně výstupní piny používány k číslicové komunikaci protokolem I^2C, který pracuje rychlostí 9600 b/s. V případě potřeby může být pin DIGAI1 přepnut pro příjem analogového signálu, což umožňuje zvýšení vzorkovací frekvence analogového senzoru. Tato možnost však není standardním firmwarem podporována.

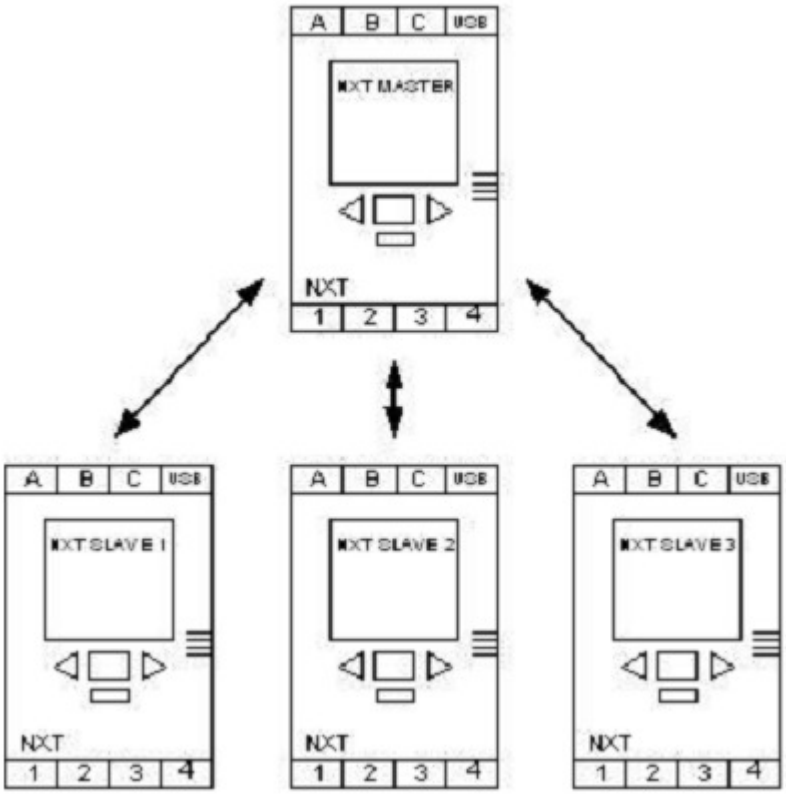
1.3.1.4 Bluetooth

Blutooth zařízení implementované v NXT umožňuje bezdrátové připojení s dalšími kostkami, počítačem, mobilním telefonem, či jiným bluetooth zařízením. Umožňuje tak bezdrátové ovládání, komunikaci či přenos souborů z nebo do kostky. Rozšiřuje tak možnosti uplatnění robotů Mindstorms.

Pro komunikaci přes bluetooth má kostka čip CSR BlueCore 4 verze 2, který jí umožňuje připojení až ke třem dalším zařízením zároveň, ale komunikovat v jeden časový okamžik může pouze s jedním zařízením. Ke komunikaci se využívá Serial Port Profi-

le(SSP), který prezentuje propojení jako bezdrátový sériový port. Kostka může být připojena pouze k zařízením, která jsou schopna používat LEGO MINDSTORMS NXT komunikační protokol a podporují SSP. Aby bluetooth neodebíralo příliš mnoho energie, byla použita technologie bluetooth třídy II s dosahem do 10m na volném prostranství.

Kostky mohou pomocí bluetooth navzájem komunikovat, a to na základě master/slave protokolu. Jedna z kostek musí být tzv. master a ostatní kostky jsou tzv. slave. Slave kostky nemohou mezi sebou komunikovat přímo. K vzájemné komunikaci musejí využít mastera, který komunikuje přímo se všemi připojenými slave kostkami. Master nemůže přijímat informace od všech slave kostek najednou, ale vždy jen od jedné z nich. Naopak je schopen vysílat data všem slavům současně. Komunikace totiž probíhá po čtyřech kanálech. Kanál 0 je využíván slavy pro komunikaci s masterem, zatím co zbylé tři kanály slouží k odesílání dat od mastera slavům.

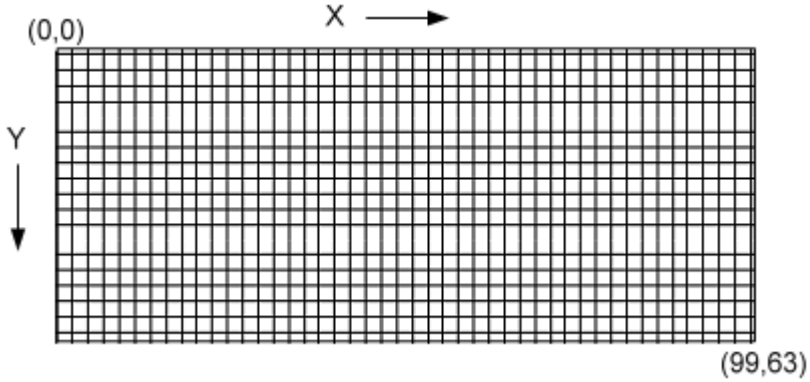


obr. 5 Schéma master/slave komunikace mezi kostkami

1.3.1.5 Display

Monochromatický maticový LCD Display o rozlišení 100x64 pixelů slouží především k zobrazení uživatelského prostředí a usnadňuje debugování programů. Zobrazovací plocha je o rozměrech 26x40.6mm. Display je ovládaný přes UltraChip 1601,který dostává instrukce přímo od hlavního mikroprocesoru prostřednictvím SPi rozhraní. Zobrazení na LCD je kontinuálně po řádcích aktualizováno. Pro kompletní aktualizaci LCD je zapotřebí 17ms.

Display je v paměti alokován jako dvourozměrné pole o rozměrech [8;100] s počátkem v levém horním rohu. Data do LCD kontroléru jsou odesílána následovně: první byte kontroluje prvních 8 pixelů vertikálně a druhý byte kontroluje prvních 8 bitů horizontálně. Pro větší názornost alokování pixelů se podívejte na následující obrázek obr. 6.



obr. 6 Alokace pixelů na display

1.3.1.6 Zvukový výstup

Kostka obsahuje diferenční zesilovací čip od společnosti SUNPLUS, který zesiluje a zlepšuje kvalitu zvuku. Čip je schopen dosáhnout maximálního zesílení o velikosti 20. Zvukový výstup je tvořen PWM signálem, který je řízen hlavním mikroprocesorem ARM7. Na výstupu jsou umístěny filtry zamezují převzorkování šumu v signálu. Zesílený a vyfiltrovaný signál jde do malého reproduktoru o průměru 21mm a impedanci 16Ω.

1.3.2 Servomotory

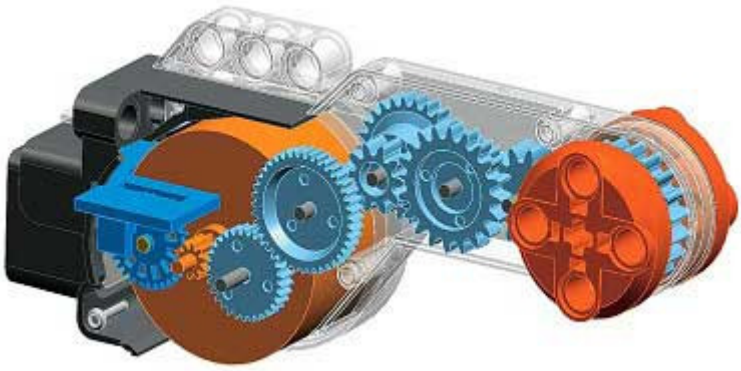
Motory dodávané se stavebnicí jsou dostatečně výkonné k tomu, aby byly schopné rozpohybovat robota nebo zvedat předměty. Pokud by nebyla síla motoru dostatečná, lze jí pomocí standardních lego součástek zpřevodovat, tak aby vyhovovala vašim požadavkům. Obsahují impulsní snímač otáček, který umožňuje měření natočení či rychlosti otáčení osy motoru.

Motory jsou napájeny 9V. To, co se ukrývá pod krytem motoru, je vidět na obr. 7. Modrá část vlevo na obrázku je impulsní senzor otáček, který snímá natočení motoru s přesností ±1°. Za jedno otočení motoru vyšle přesně 360 impulsů, protože má kostka kvadraturní modulátor, je maximální možná citlivost senzoru až 720 impulsů na otáčku. Oranžová část za snímačem otáček je hlavním pohonem motoru. Ten je zpřevodován na výstup motoru převodovým ústrojím světle modře na obrázku v poměru 48:1.

Technické parametry motoru

- Rychlost otáčení:177 ot./min
- Kroutící moment:16.7 Ncm
- Síla:2.03 W
- Účinnost:41%

Podrobnější informace o servomotorech včetně různých charakteristik naleznete ve zdroji [10].



obr. 7 Servomotor

1.3.3 Senzory

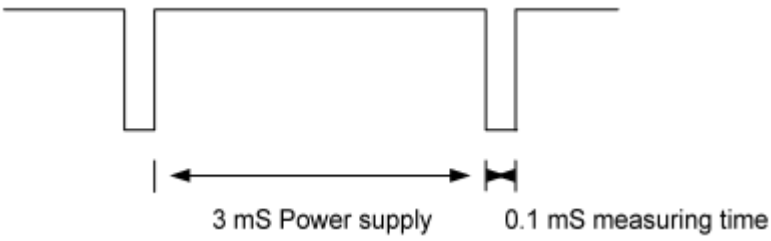
Senzory umožňují kostce reagovat na vnější podněty. V základu lze ke kostce přes výstupní porty připojit čtyři senzory. Tento počet lze rozšířit až na šestnáct použitím senzorového multiplexoru [9] od společnosti HiTechnic [10]. Multiplexor poskytuje možnost připojení až čtyř senzorů na jeden výstup inteligentní kostky. K dispozici jsou základní senzory dodávané se stavebnicí nebo nadstandardní senzory, které můžeme dokoupit přímo od společnosti Lego nebo od Hitechnic a mnoho dalších. Senzory dělíme do tři skupin aktivní, pasivní a digitální.

1.3.3.1 Aktivní senzory

Aby byla zajištěna zpětná kompatibilita se senzory vyvinutými pro původní stavebnici s RCX kostkou, je NXT kostka vybavena zdrojem proudu, který má v průměru na výstupu 18mA. Zdroj dodává senzorům potřebnou energii vždy po dobu 3 ms a poté se po dobu 0.1ms měří analogová hodnota senzoru viz. obr. 8.

Aktivními senzory jsou:

- Senzor světla
- Impulsní senzor otáček



obr. 8 Průběh na A/D převodníku při použití aktivního senzoru

1.3.3.2 Pasivní senzory

Mezi pasivní senzory řadíme senzory, které narozdíl od aktivních senzorů nepotřebují napájení zdrojem proudu. Pasivní senzory ale využívají stejný A/D převodník, proto je jejich vzorkovací frekvence též 333Hz.

Pasivními senzory jsou:

- Dotykový senzor
- Zvukový senzor
- Senzor teploty
- Senzor světla ze sady NXT

1.3.3.3 Digitální senzory

Všechny senzory, které využívají I^2C komunikaci, řadíme mezi digitální senzory, protože obsahují externí mikrokontroler, který ovládá vzorkování zařízení.

Digitálními senzory jsou:

- Ultrazvukový senzor
- HiTechnic barevný senzor

1.3.3.4 Dotykový senzor

Je ve své podstatě obyčejným tlačítkem bez aretace. Může sloužit jako detektor překážek, dorazů či jako jednoduchý ovládač robota. Umožňuje rozpoznání stlačení nebo uvolnění tlačítka senzoru. Stejnou funkci mohou zastávat i tlačítka umístěná přímo na inteligentní kostce. Ta se však nedají snadno umístit tak, aby vyhovovala potřebám úlohy. Slouží spíše k ručnímu ovládání robota.

obr. 9 Dotykový senzor

1.3.3.5 Senzor zvuku

Zvukový senzor měří úroveň hlasitosti zvuku přibližně v rozsahu 50-90dB. Hodnoty však uvádí v procentech 0-100%. Senzor umožňuje změnit prostřednictvím vážených filtrů senzitivitu tak, aby byly naměřené hodnoty podobné vnímání hlasitosti lidským uchem. Senzor tak umožňuje robotovi reagovat na zvukové podněty, například tlesknutí nebo písknutí. Podrobnější popis senzoru včetně průběhů naměřených na senzoru naleznete ve zdroji [5].

obr. 10 Senzor zvuku

1.3.3.6 Ultrazvukový senzor

Senzor umožňuje robotovi detekovat pohyb, překážky a měřit vzdálenost od objektů. Vzdálenost měří v centimetrech nebo v palcích. Je schopen měřit vzdálenost až do jeden a půl metru s přesností ±3cm záleží na velikosti a členitosti plochy měřeného objektu.

Senzor pracuje na principu sonaru a ke komunikaci s kostkou využívá protokolu I^2C. Měří dobu, za kterou se vyslaný ultrazvukový signál o frekvenci 40 kHz vrátí odražený od měřeného objektu zpět do senzoru. Ze změřené doby pomocí vlastního mikroprocesoru vypočte vzdálenost objektu. Pokud se v místnosti nachází více těchto senzorů, je možné, že mezi sebou budou interferovat a ovlivní se tak přesnost jejich měření.

obr. 11 Ultrazvukový senzor

1.3.3.7 Senzor světla

Senzor umožňuje robotovi detekovat intenzitu světla, případně rozlišovat barvy v odstínech šedi. Jednou z hlavních součástí senzoru je LED dioda, která je buď zapnutá nebo vypnutá dle zvoleného módu senzoru. V módu detekce odstínu barvy je LED dioda zapnutá a osvětluje detekovaný objekt. Část emitovaného světla se dle barvy povrchu absorbuje a zbytek světla se odrazí zpět, kde ho zachytí fototranzistor. Ze zachyceného odrazu světla se vyhodnotí odstín šedi v rozmezí 0-255. Protože senzor dokáže detekovat pouze odstíny šedi, je vhodné při práci s ním používat vzájemně kontrastní barvy např. bílou a černou. Obdobně jako mód detekce barvy pracuje i režim detekce odraženého světla, který udává hodnotu odraženého v procentech. V režimu detekce intenzity osvětlení je LED dioda zhasnutá a fototranzistor snímá pouze intenzitu světla okolí.

obr. 12 Senzor světla

1.3.3.8 Barevný senzor Lego V2.0

Senzor je dodávaný v novém setu stavebnice Lego Mindstorms 2.0. Umožňuje robotovi detekovat barvy v barevném modelu RGB. Princip je obdobný jako u senzoru světla, který byl popsán o kapitolu výše. S tím rozdílem, že senzor má RGB diodu, která svítí v barvách červená, zelená a modrá a světelný senzor, který je citlivý na všechny světelné délky. Senzor umožňuje rozpoznávání barvy ve dvou hlavních módech: RGB, který obsahuje hodnoty intenzit jednotlivých barevných složek, nebo Colorval, který udává přesné číslo barvy např. 1 pro černou a 6 pro bílou. Metodu Colorval doporučuji požívat při rozeznávání dostatečně kontrastních barev ze základní barevné škály: černá, modrá, zelená, žlutá, červená, bílá. Protože jde o senzor z druhé řady NXT, musí být na kostce nahrán firmware verze 1.26 nebo vyšší.

obr. 13 Barevný senzor Lego V2.0

1.3.3.9 Barevný senzor HiTechnic

Senzor, vyráběný společností HiTechnic, je opět principielně podobný jako předchozí senzor. Liší se použitím bílé LED diody a specielního barevně citlivého čipu, který je rozdělen do tří detekčních částí překrytých modrým, červeným a zeleným filtrem. Senzor může pracovat, jak v aktivním módu(LED dioda zapnuta), tak i v módu pasivním(LED dioda vypnuta). Také pro rozlišování barev můžete využít jeden ze dvou módů číslo barvy nebo hrubý RGB mód. V módu, který vrací přímo číslo barvy, jsou barvy číslovány od 0(černá) do 17(bílá) viz. obr. 14. Pokud by vám metoda nevyhovovala, senzor vám dá k dispozici hrubé RGB hodnoty, ze kterých sám rozhoduje barvu. Umožňuje vám tak experimentovat a navrhnout si vlastní detektor barev. Senzor je schopen vypočítat hodnotu barvy přibližně stokrát za vteřinu. Výrobce uvádí, že pro zlepšení přesnosti, je třeba ze stránek[] stáhnout konfigurační soubor pro elektrické sítě pracující na frekvenci 50Hz, protože je senzor od výrobce nakonfigurován na síť pracující na frekvenci 60Hz.

obr. 14 Barevný senzor HiTechnic s barevnou škálou

1.3.3.10 Porovnání barevných senzorů

Srovnával jsem barevný senzor Lego V2.0 a barevný senzor od výrobce HiTechnic(HT). K porovnání jsem využil inteligentní NXT kostku s firmwarem verze 1.28. Na výstupní port jsem připojil servomotor, který sloužil k otáčení palety barev. Na vstupní port jsem připojil vždy pouze jeden ze senzorů. Veškerá měření probíhala za běžného denního světla. Se senzory jsem provedl celkem dva různé pokusy. První z pokusů byl zaměřen na schopnost senzoru vypočíst správnou hodnotu barvy v závislosti na vzdálenosti senzoru od snímaného objektu. V druhém pokusu jsem se zaměřil na stálost hrubých hodnot senzoru při detekci jedné barvy.

V pokusech jsem použil základní barvy. Všechny barvy včetně jejich reprezentačních čísel pro jednotlivé senzory jsou zaznamenány v tab. 1.

tab.1 Barvy s číselným kódem

Barvy	Modrá	Červená	Zelená	Černá	Žlutá	Bílá
Lego	2	5	3	1	4	6
HT	2	9	4	0	6	17

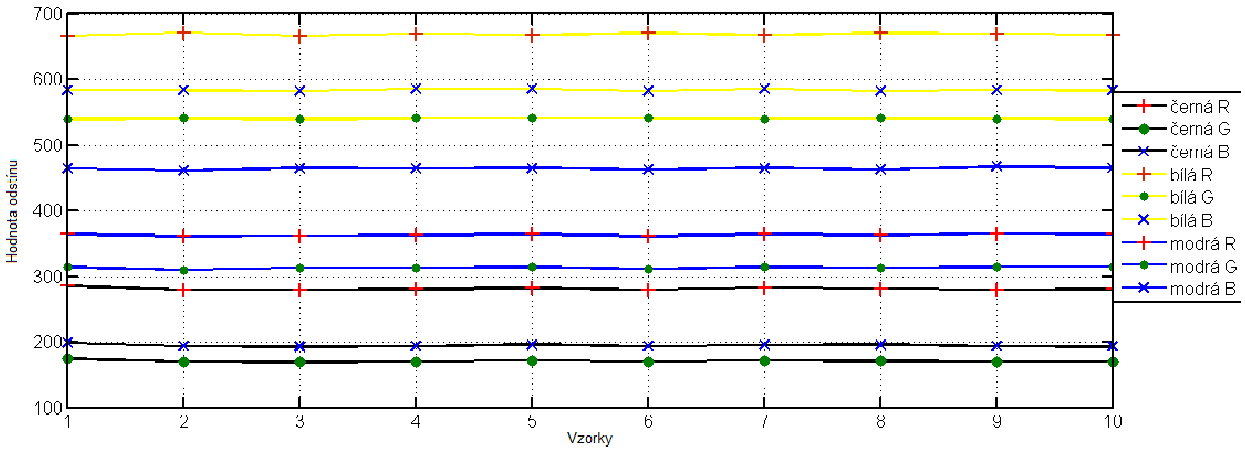
V prvním pokusu jsem postupně zvyšoval vzdálenost senzoru a objektu od 0,5cm do 3cm a zaznamenával hodnoty ze senzoru tab. 2. Z tabulky je vidět, že Lego senzor je schopen precizně pracovat do vzdálenosti okolo 1cm, pro větší vzdálenosti vyhodnotil všechny měřené barvy jako černé. HT senzor byl schopen správně detekovat ve všech měřených vzdálenostech většinu barev. Problémy měl pouze s barvou černou a bílou. Pro krátké vzdálenosti 0.5-1cm identifikoval bílou barvu jako světle žlutou, ale pro větší vzdálenost senzoru již bílou rozpoznal správně. Naopak černou barvu správně rozpoznal pouze na nejkratší vzdálenost. Při aplikaci senzoru např. v robotovi sledujícím čer-

nou čáru na bílém podkladu, by mohlo docházet k potížím způsobeným právě špatnou identifikací černé a bílé barvy.

tab.2 Naměřená data k prvnímu pokusu

Senzor	Barvy/Vzdálenost[cm]	Modrá	Červená	Zelená	Černá	Žlutá	Bílá
Lego	0.5	2	5	3	1	4	6
	1	2	5	3	1	4	6
	2	1	1	1	1	1	1
	3	1	1	1	1	1	1
HiTechnic	0.5	2	9	4	0	6	12
	1	2	9	4	12	6	12
	2	2	9	4	17	6	17
	3	2	9	4	17	6	17

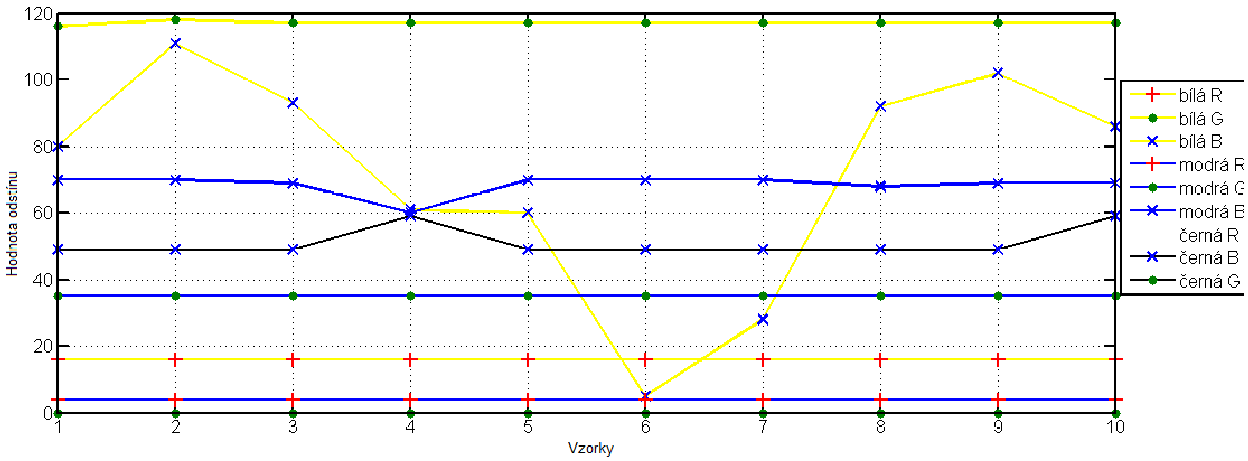
V druhém pokusu jsem se zabýval stálostí hodnot jednotlivých složek snímané barvy. Z možné palety barev jsem vybral pouze barvy tři: černou, bílou a modrou. Senzor jsem umístil do vzdálenosti 0.5cm. Každou z barev jsem nechal desetkrát změřit a hodnoty zaznamenat. Naměřené hodnoty jsem pro názornost vynesl do grafů obr.15 a obr.16 .



obr. 15 Naměřené hodnoty senzorem Lego

Z průběhů v grafu je vidět, že senzor Lego měří složky barev téměř konstantně s drobnou odchylkou ±2. Z toho lze usoudit, že pro tuto vzdálenost je senzor velmi přesný.

V obr. 16 jsou průběhy zaznamenané senzorem HT. Složky zelené a červené jsou detekovány velmi přesně většinou s nulovou odchylkou, ale modrá složka má odchylky velké až ±50. Při použití tohoto senzoru v módu hrubých hodnot proto doporučuji vyřadit modrou složku a spolehnout se na zbylé dvě barevné složky.

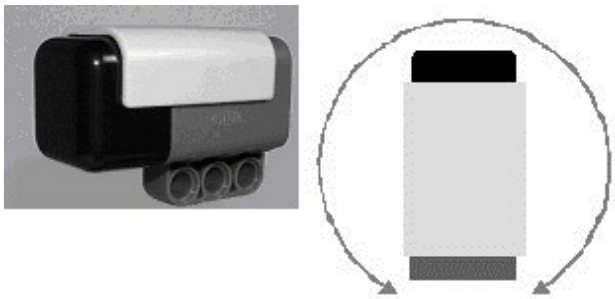


obr. 16 Naměřené hodnoty senzorem HT

Další měření a testování těchto senzorů za různých světelných podmínek můžete nalézt ve zdroji [4].

1.3.3.11 Gyroskopický senzor

Je dalším senzorem od výrobce HiTechnic. Senzor umožňuje snímat v jedné ose rychlost a směr natočení. Hodnoty udává ve stupních za vteřinu. Senzor je schopen měřit až 360° za sekundu. Může být využit ke stavbě robota, který je schopen balancovat, nebo v dalších úlohách, kde je měření rychlosti otáčení zapotřebí. Protože senzor využívá analogový vstup, je frekvence snímání přibližně 300Hz.



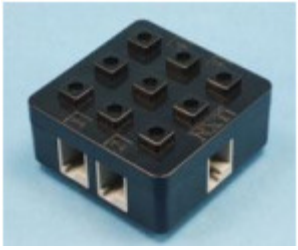
obr. 17 Gyroskop a náznak osy snímání rychlosti otáčení

1.3.3.12 Kompas

Je digitálním senzorem schopným měřit magnetické pole země a z něj vypočíst odchylku od severního pólu ve stupních. Umožňuje tak zpřesnit navigaci robota v prostoru. Senzor měří v rozsahu 0-359° a až stokrát za sekundu.

1.3.3.13 Senzorový multiplexor

Není sice standardním senzorem, ale připojuje se na vstupní port kostky, proto ho řadím mezi senzory. Multiplexor umožňuje na jeden standardní vstup kostky připojit až 4 senzory. Ke komunikaci s kostkou využívá I^2C protokolu. K multiplexoru je externě připojena devítivoltová baterie, která zajišťuje potřebné napětí pro připojené senzory. Programovou podporu má prozatím pouze v Labview 2009 a RobotC. Více podrobností včetně výpisu podporovaných senzorů a možnostech programování se dočtete na stránkách výrobce[10] nebo v tomto dokumentu[9].



obr. 18 Senzorový multiplexor

1.4 Softwarové vybavení

Standardní firmware inteligentní kostky dodávaný společností Lego je open-source neboli systém se zpřístupněným zdrojovým kódem. Umožňuje tak dalším vývojářům vytvořit například vlastní kompilátor a nebo alternativní firmware k inteligentní kostce. Lego samotné vyvíjí stále nové verze svého původního firmwaru. V současnosti je poslední verze firmwaru 1.29. Mezi nejznámější alternativní firmwary patří leJOS a nxtOSEK.

1.4.1 Standardní firmware lego

Společnost lego, pro usnadnění vývoje nových firmwarů a kompilátorů, zpřístupnila zdrojové kódy a k nim i potřebnou dokumentaci. Firmware je napsán v ANSI C, který je podobný jazyku C. Skládá se z několika modulů a virtuálního stroje, který spouští program uložený v rxe souboru. Soubor obsahuje vše potřebné k běhu programu. Takže

jeden soubor rxe reprezentuje jeden spustitelný program. Pro podrobnější popis firmwaru a běhu programu doporučuji Software Developer Kit [11], který je dostupný na stránkách Lego [26].

1.4.1.1 Základní Datové typy

Podporované nejnovějším firmwarem verze 1.29.

Integer – čísla se znaménkem nebo bez znaménka o délce 8, 16 a 32bit

Float – čísla s plovoucí desetinnou čárkou o délce 32bit

Array – je polem jednoho nebo více prvků jedné datové struktury

Boolean – má pouze dva stavy TRUE nebo FALSE

String – je speciálním druhem pole, ve kterém je uložen textový řetězec

1.4.2 Firmware leJOS

Je dalším open-source firmwarem ke kostce. Jeho hlavní součástí je malý virtuální Java stroj, který podporuje programování v jazyce NXJ(Not exactly Java), který je odvozen od Javy. Firmware umožňuje objektové programování, preemptivní běh vláken, více rozměrná pole, synchronizaci atd. K dispozici jsou i různé nástroje, které usnadňují debugování programu, či umožňují bezdrátové ovládání kostky.

1.4.3 Firmware nxtOSEK

NxtOSEK firmware obsahuje ovladač vstupů a výstupů z firmwaru leJOS a speciální platformy pro vestavěné real-time systémy. Umožňuje programování v jazyce ANSI C nebo C++ a generování programu ze simulace sestavené v simulinku, který je součástí Matlabu. K oběma jazykům je dostupné rozhraní pro programování obsahující řadu senzorů jak od Lega, tak i od ostatních výrobců.

2 Programování robota Lego

2.1 Úvod

Za řadu let se k NXT vyvinulo mnoho programovacích jazyků a prostředí. Od jednoduchých grafických až po komplexní textové. Díky široké nabídce si může tvůrce robota vybrat programovací jazyk, který nejlépe vyhovuje jeho potřebám a znalostem. Většina jazyků pracuje se základním firmwarem. Některé však mají firmware vlastní a je tak zapotřebí ho na kostku nainstalovat.

2.2 Přehled programovacích jazyků

V přehledu naleznete pouze ty nejpoužívanější a nejzajímavější programovací prostředí. Také jejich vlastnosti jsem omezil pouze na ty nejdůležitější. Rozsáhlejší přehled softwaru a řídících prostředí pro NXT kostku je k dispozici ve zdroji[4].

tab.3 Porovnání dostupných softwarů

Vlastnosti	NXT-G	RoboLab	NBC	NXC	RobotC	LabVIEW	leJOS NXJ	leJOS OSEK
Typ	Grafický	Grafický	Asembler	Not exactly C	C	Grafický	Java	ANSI C
Firmware	Standardní	Standardní	Standardní	Standardní	Standardní	Standardní	Vlastní	Vlastní
Platformy	Win, Mac OS	Win, Mac OS	Win, Mac OS, Linux	Win, Mac OS, Linux	Win, Linux	Win, Mac OS	Win, Mac OS, Linux	Win
Vlákna	Ano	Ano	Ano	Ano	Ano	Ano	Ano	Ano
Bluetooth kostka/PC	Ano	Ne	Ano	Ano	Ano	Ano	Ano	Ano
bluetooth kostka/kostk	Ano	Ne	Ano	Ano	Ano	Ano	Ano	Ne
Licence	Placená ve standardním setu	Placená v Education setu	Freeware	Freeware	Placená	Placená	Open-source	Open-source

V podrobnějším popisu jsem se zaměřil především na softwary s licencí freeware nebo open-source, a to zejména z dostupnosti širší skupině lidí.

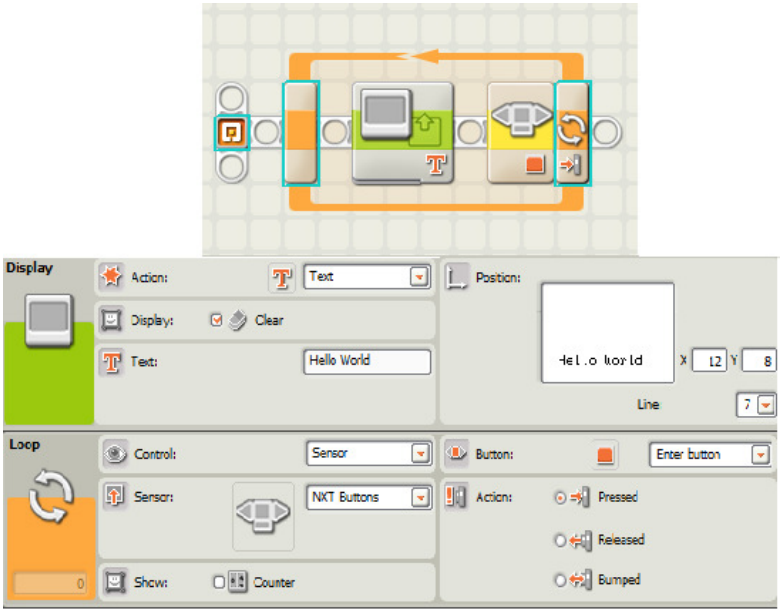
2.3 NXT-G

Je grafickým programovacím softwarem dodávaným ke standardní stavebnici. Vznikl za spolupráce společnosti Lego a National Instruments. Je založen na programu LabVIEW, který používá barevné čáry k datovému propojení jednotlivých bloků. Na podobném principu funguje i NXT-G, kde skládáním jednotlivých nastavitelných bloků za sebe vyváříte program. V blocích můžete měnit jejich vlastnosti a parametry. Tvorba programu v tomto softwaru je víceméně intuitivní, proto k jeho užívání není třeba znalostí programování. Hodí se spíš pro tvorbu jednodušších programů a pochopení tak základních principů programování.

Software dále umožňuje propojení s kostkou jak přes usb, tak i prostřednictvím bluetooth. Po připojení lze na kostce procházet a měnit položky uložené v paměti, nahrávat a spouštět programy, či přehrát verzi firmwaru.

2.3.1 Program HelloWorld v NXT-G

Program se skládá ze dvou bloků displaye a smyčky. Display, který zajišťuje zobrazování požadovaného textu, je umístěn ve smyčce, která se ukončí stisknutím tlačítka enter. V obrázku je vidět jak celé schéma programu, tak i nastavení jednotlivých bloků.



obr. 19 Program HelloWorld v NXT-G

2.4 NBC/NXC

NBC(Next Byte Codes) je jednoduchý textový programovací jazyk podobný assembleru, oproti tomu NXC(Not Exactly C) je programovacím jazykem vyšší úrovně podobný jazyku C. Ke kompilaci programu napsaném v NXC se využívá kompilátor pro NBC rozšířený o soubory zdrojového kódu k NXC. Jazyky pracují se standardním firmwarem Lego a píší se ve vývojovém prostředí BricxCC. Pokud máte základní programátorské znalosti a preferujete psaní, je pro vás NBC/NXC vhodným nástrojem.

2.4.1 Vývojové prostředí BricxCC

Hlavní předností tohoto programu je jeho volná dostupnost, takže si jeho aktuální verzi můžete stáhnout přímo ze stránky vývojáře. Instalace obsahuje vše potřebné k napsání vašeho prvního NQC/NXC programu včetně průvodců v programování k oběma jazykům. Program umožňuje připojení, ovládání a procházení kostky NXT prostřednictvím usb nebo bluetooth.

Poznámka: Pro kompilování programů na firmware verze 1.2x je potřeba v nastavení přepnout kompatibilitu kompilátoru.

Edit/Preferences/Compiler/NBC/NXC/NXT compatible 2.0 firmware

2.4.2 Program HelloWorld v NXC

```
task main()
{
  TextOut(10,LCD_LINE2,"Hello World!");
  while(ButtonPressed(BTNCENTER,false)==0);
}
```

Každý program psaný v NXC musí obsahovat task main, který je spouštěn na začátku běhu programu. Program může obsahovat více tasků, přičemž každý task představuje jedno vlákno programu. Program není třeba nijak ukončovat, protože se ukončí sám po vykonání poslední instrukce.

2.5 LeJOS NXJ

Je programovacím jazykem založeným na Javě a fungujícím na stejnojmenném firmwaru. Pro práci s robotem obsahuje mnoho užitečných tříd, které jsou velmi dobře popsány v rozhraní pro programování aplikace(API) [14]. Součástí prostředí jsou různé nástroje pro ovládání, nahrávání a debugování programů. Pokud by vám dostupné nástroje nevyhovovaly, můžete si vytvořit vlastní. Jazyk nemá vlastní vývojové prostředí, ale využívá volně dostupné profesionální prostředí Netbeans nebo Eclipse.

Protože je Java mezi programátory rozšířená, jde o jeden z nejpoužívanějších nástrojů k programování kostky NXT. Nejste-li zdatní Java programátoři, oceníte tutoriál a řadu vzorových aplikací, které jsou umístěné na webové stránce vývojáře[13].

2.5.1 Výhody LeJOS NXJ

- Podpora objektového programování
- Open-source projekt s mnoha developery
- Podpora platform Windows, Linux, Mac OS X
- Rychlost běhu programu
- Přesné řízení servomotorů
- Mnoho matematických funkcí, vícerozměrná pole
- Preemptivní běh vláken
- Širokou komunitu s komunitním fórem
- Další výhody naleznete v tomto zdroji[13]

2.5.2 Popis instalace

LeJOS NXJ může být nainstalován na většinu komerčně známých operačních systémů, ale já jsem se v mém popisu zaměřil na nejrozšířenější Windows. Anglicky psaný postup instalace jak pro Windows, tak i pro ostatní operační systémy je popsaný ve zdroji[15].

2.5.2.1 Předpoklady k instalaci

1. K instalaci budete potřebovat ovladač ke kostce NXT, který je dodáván se standardním Lego softwarem. Pokud již máte tento software nainstalovaný, máte nainstalovaný i potřebný ovladač, v opačném případě software nainstalujte.
2. Také budete potřebovat Java Development Kit (JDK) verze 1.5 nebo 1.6, který je dostupný na stránce http://java.sun.com/.

2.5.2.2 Stažení a instalace

1. LeJOS NXJ software stáhnete z webové stránky http://lejos.sourceforge.net/nxj-downloads.php
2. Spustíte grafického průvodce instalací, který vás krok za krokem provede instalací.
3. Ihned po ukončení instalátoru se spustí nástroj na instalaci nového firmwaru do vaší kostky.
4. Pro úspěšné ukončení instalace je třeba zkontrolovat proměnné prostředí ve vašem PC. Ve Windows 7 můžete tyto proměnné nastavit tak, že půjdete do Ovládací panely/ Systém / Upřesnit nastavení systému / Upřesnit / Proměnné prostředí. Jednotlivé proměnné by měly obsahovat hodnoty dle tab. 4. Pokud se ve vaší cestě vyskytují mezery, dejte ji do uvozovek.

tab.4 Nastavení proměnných prostředí

Variable	Value	Example
NXJ_HOME	The folder you installed leJOS NXJ into	C:\lejosbeta85\lejos_nxt
JAVA_HOME	The folder where you installed the JDK	C:\Program Files\Java\jdk1.6.0_16
PATH	Add the bin folders for the JDK and leJOS	C:\Program Files\Java\jdk1.6.0_16\bin;C:\lejosbeta85\lejos_nxt\bin;

2.5.2.3 Instalace vývojového prostředí

LeJOS NXJ lze používat v kombinaci s NetBeans nebo Eclipse, v obou případech jde o plnohodnotné vývojové prostředí, které nabízí řadu možností a usnadňuje tak psaní programů. Protože jsem měl již z dřívějška zkušenosti s programem NetBeans, zvolil jsem ho i nyní.

1. NetBeans IDE je volně ke stažení v několika verzích dle podporovaných technologií na stránce. (http://netbeans.org/downloads/index.html) Pro naše potřeby nám stačí verze Java SE. Stažený balíček NetBeans nainstalujeme dle průvodce instalací.

2. K tomu, aby bylo možné vytvářet nové NXJ projekty, je třeba nainstalovat do NetBeans plugin. To provedete tak že v Netbeans půjdete do Tools/Plugins/Downloaded/ a kliknete na Add Plugins. Poté vyhledáte složku, do které jste nainstalovali leJOS NXJ projekty a v ní vyhledáte NXJPlugin/build/nxjplugin.nbm. Potvrdíte “ok“ a plugin se sám automaticky nainstaluje.

3. Pro vytvoření nového projektu nyní stačí kliknout na “New Project“, zvolit kategorii samples a projekt “NXJ project“. Projekt pojmenovat, zvolit jeho umístění a kliknout na “finish“. Pokud budete chtít přejmenovat hlavní třídu je potřeba přepsat i odkazy umístěné v souboru build.xml/uploadandrun.

4. V takto připraveném prostředí už můžete kompilovat NXJ projekty a nahrávat je přímo prostřednictvím usb nebo bluetooth do kostky NXT.

Pokud budete mít problémy s vytvářením či psaním vlastních projektů, můžete si přečíst průvodce programováním ve zdroji[13] nebo požádat o radu na fóru <http://lejos.sourceforge.net/forum/>.

2.5.3 Program HelloWorld v NXJ

```
import lejos.nxt.*;

public class HelloWorld
{
  public static void main (String[] aArg)
  throws Exception
  {
     LCD.drawString("Hello World",3,4);
     while (!Button.ESCAPE.isPressed()) {}
  }
}
```

Program napsaný v NXJ může obsahovat i několik tříd, ale alespoň jedna ze tříd musí mít metodu main, která se vykoná první po spuštění programu. Program se sám ukončí po vykonání všech instrukcí.

3 Realizace dvou speciálních úloh

3.1 Úloha Inverzní kyvadlo

V úloze inverzního kyvadla jsem se zabýval dvoukolovým robotem, který je schopen balancovat ve vzpřímené poloze. Principielně je robot podobný komerčně známému dopravnímu prostředku Segway. Prvním impulsem k tvorbě této úlohy byl robot StandAlone [19], který je řízený přes usb kabel programem běžícím v počítači. Základní myšlenkou tedy bylo vytvořit robota, který by byl na rozdíl od robota StandAlone soběstačný a byl by schopen pohybovat se všemi směry. V kapitole popíšu konstrukci robota. Dále identifikaci stavového popisu a návrh regulátoru. V závěru kapitoly je podrobný popis programu zaměřující se především na nejdůležitější části.

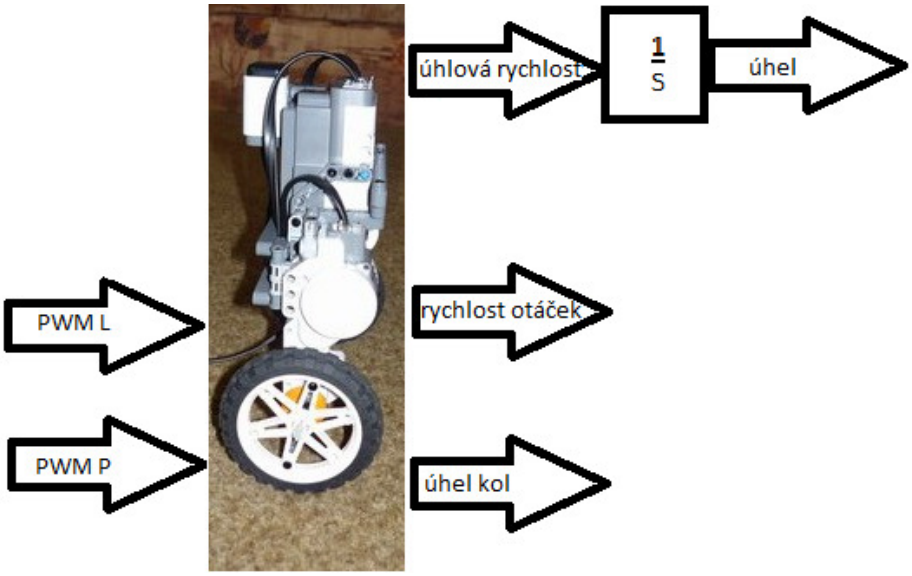
3.1.1 Konstrukce

Z konstrukčního hlediska je důležité, aby těžiště robota bylo umístěno nad osou otáčení kol a aby konstrukce byla schopna unést celou váhu robota ve vzpřímené poloze. Konstrukce mnou používaného robota byla inspirována robotem NXTway-GS, který byl sestrojen Yorihisou Yamamotem [24]. Z konstrukce jsem odebral ultrazvukový senzor, který sloužil k detekci překážek a rampu, která sloužila k umístění dalších doplňkových senzoru. Robot používá ke stabilizaci dva servomotory připojené do výstupů B a C a gyroskop připojený na vstupu 2. Gyroskopický senzor musí být na těle robota umístěn tak, aby byla jeho osa snímání v ose náklonu robota. Finální konstrukce je na obr. 20. Návod k sestavení robota je v elektronické příloze.

obr. 20 Inverzní kyvadlo

3.1.2 Vstupy a výstupy

Inverzní kyvadlo má celkem dva vstupy a čtyři výstupy, které jsou ukázány na obr. 21. Vstupy jsou PWM signály ovládající servomotory. Výstupem je průměrná hodnota úhlu natočení $\theta_{p,l}$[°] a rychlosti otáčení $\dot{\theta}_{p,l}$[°/s] kol. Oba tyto výstupy je možné měřit přímo z impulsního snímače otáček umístěného v motorech. Poslední dvě výstupní veličiny jsou úhel ψ [°] a úhlová rychlost $\dot{\psi}$[°/s] náklonu robota. Protože gyroskopický senzor měří pouze úhlovou rychlost, je třeba úhel náklonu dopočítat integrálem z úhlové rychlosti $\dot{\psi}$.



obr. 21 Vstupy a výstupy

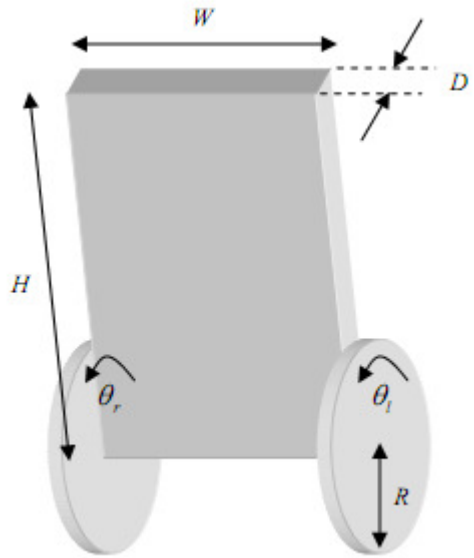
3.1.3 Identifikace

K tomu, aby bylo možné navrhnout k inverznímu kyvadlu regulátor, bylo zapotřebí provést identifikaci stavového popisu. Protože z robota nešlo snadno získat potřebné průběhy, rozhodl jsem se provést identifikaci formou fyzikálního popisu systému. Naštěstí dokumentace k NTXway-GS obsahovala podrobný popis fyzikálního modelu. To mi značně usnadnilo práci s pochopením a identifikací stavových rovnic modelu. Prvním krokem v identifikaci modelu bylo zjištění jeho konstantních parametrů, které jsou zaznamenány v tabulce tab.5. V identifikaci popisuji jen to nejdůležitější z postupu, podrobný popis identifikace včetně výpočtů je ve zdroji [24].

Hodnoty J_m, f_m a f_w byly zvoleny odhadem, neboť je náročné je změřit. Převod motoru byl pro zjednodušení výpočtu zvolen roven jedné. Konstanty včetně proměnných systému jsou zobrazeny na obr. 22 a obr. 23. Oba dva obrázky jsem převzal z dokumentace k NXTway-GS.

tab.5 Konstanty modelu

Konstanta	Velikost	Veličina	Popis
H	0.144	M	výška těla robota
W	0.14	M	šířka těla robota
D	0.04	M	hloubka těla robota
L	H/2	M	poloha těžiště
M	0.6	Kg	hmotnost
m	0.03	Kg	hmotnost kol
R	0.04	M	poloměr kol
J_w	$mR^2/2$	Kgm^2	moment setrvačnosti kol
g	9.81	m/s^2	gravitační zrychlení
J_ψ	$ML^2/3$	Kgm^2	moment setrvačnosti těla
$J\phi$	$M(W^2+D^2)/12$	Kgm^2	moment setrvačnosti odklonu
J_m	10^{-5}	Kgm^2	moment setrvačnosti motoru
R_m	6.69	Ω	odpor motoru
K_b	0.468	Vs/rad	převodní konstanta motoru
K_t	0.317	Nm/A	konstanta kroutícího momentu motoru
N	1	-	převod motoru
f_m	0.0022	-	tření mezi tělem a motorem
f_w	0	-	tření s povrchem



obr. 22 Ilustrativní znázornění konstant a proměnných

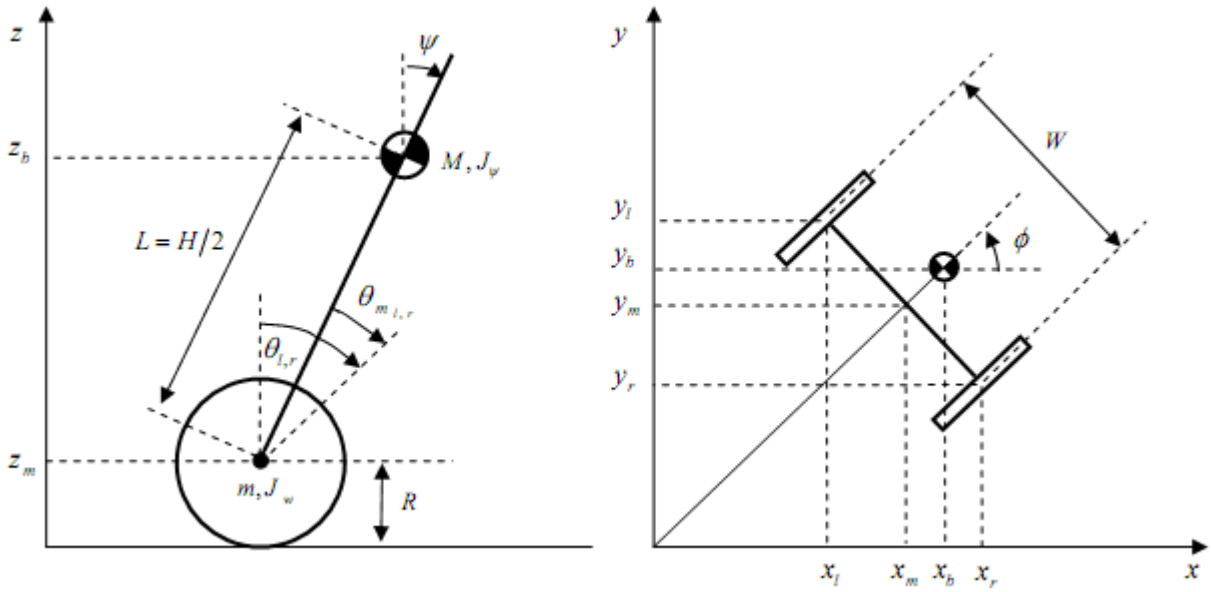
3.1.3.1 Pohybové rovnice

Lagrangeovou metodou založenou na souřadném systému v obr. 23 získám pohybové rovnice kyvadla, které po linearizaci vypadají následovně.

$$F_\theta = [(2m+M)R^2 + 2J_\psi + 2n^2 J_m]\ddot{\theta} + (MLR - 2n^2 J_m)\ddot{\psi} \tag{1}$$

$$F_\psi = (MLR - 2n^2 J_m)\ddot{\theta} + (ML^2 + J_\psi + 2n^2 J_m)\ddot{\psi} \tag{2}$$

$$F_\phi = [\frac{1}{2} mW^2 + J_\phi + \frac{W^2}{2R^2}(J_\omega + n^2 J_m)] \tag{3}$$



obr. 23 Inverzní kyvadlo v souřadném systému

3.1.3.2 Stavový popis

Dosazením konstant z tab. 5 do pohybových rovnic (1), (2) a (3) získaných Langrangerovou metodou a provedením úprav získám stavový popis systému, který má obecný tvar (4) a popisuje chování systému. Takováto forma zápisu systému je zejména vhodná k provedení analýz a simulování modelu. Protože jde o systém linearizovaný, nestabilní a z robota mi nešlo snadno získat odpovídající průběhy k porovnání, neprovedl jsem již ověření identifikace a plně jsem se spolehl na identifikaci provedenou Yorhisou Yamamotem.

$$\dot{x} = Ax + Bu \qquad (4)$$
$$y = Cx + Du$$

$$\begin{bmatrix} \dot{\theta} \\ \dot{\psi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -194.23 & -85.91 & 85.91 \\ 0 & 165.75 & 41.41 & -41.41 \end{bmatrix} \begin{bmatrix} \theta \\ \psi \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 83.5 & 83.5 \\ -40.25 & -40.25 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \qquad (5)$$
$$\begin{bmatrix} \theta \\ \psi \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \theta \\ \psi \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}$$

3.1.4 Návrh regulátoru

Po regulátoru požadujeme, aby byl dostatečně rychlý a zajišťoval dostatečně robustní řízení systému. Mezi základní regulátory patří PID regulátor, proto jsem se v návrhu regulátoru nejprve zaměřil právě na něj. Bohužel se ukázalo, že samotný PID regulátor nevyhovuje požadavkům, proto jsem se dále zaměřil na regulaci stavovou zpětnou vazbou. Pro její návrh jsem po diskusi s panem Ing. Františkem Vaňkem zvolil metodu lineární kvadratické regulace, která je popsána v následující kapitole.

3.1.4.1 Lineární kvadratická regulace

Lineární kvadratická regulace(LQR) se používá k automatizovanému nalezení stavové zpětné vazby. Metoda k výpočtu používá matici Q, která obsahuje váhové faktory zpětné vazby a matici R, která je váhovým skalárním faktorem.

Zpětná vazba je dána vzorcem,

$$u = -Kx \qquad (6)$$

kde K se vypočítá ze vzorce

$$K = R^{=1} B^T P(t) \qquad (7)$$

a P vypočítáme řešením Riccatiovi diferenciální rovnice.

$$A^T P(t) + P(t)A - P(t)BR^{=1}B^T P(t) + Q = -\dot{P}(t) \qquad (8)$$

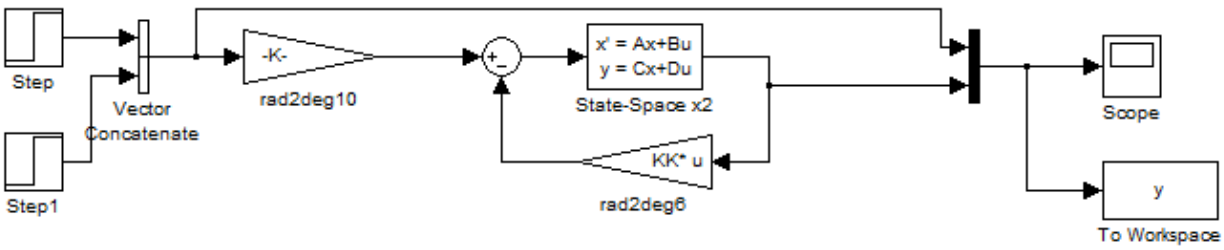
Pro návrh regulátoru jsem použil stejné matice Q a R jako byly použity v dokumentaci k NXTway-GS.

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 6\cdot 10^5 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 4\cdot 10^2 \end{bmatrix} \qquad R = \begin{bmatrix} 1\cdot 10^3 & 0 \\ 0 & 1\cdot 10^3 \end{bmatrix} \tag{9}$$

Použitím funkce lqr v programu matlab jsem získal potřebné váhové koeficienty tab. 6 pro stavovou zpětnou vazbu. Pátý integrální koeficient není potřebný, z důvodu použití jiného typu zpětné vazby než bylo použito u NXTway-GS. Pro oba vstupy jsou koeficienty stejné, proto v tab. 6 zmiňuji koeficienty pouze pro jeden vstup. Zapojení stavové zpětné vazby je vidět na obr. 24

tab.6 Koeficienty stavové zpětné vazby

Proměnná	K	hodnota
úhel kol	k1	-0,837
úhel	k2	-34,92
rychlost otáček	k3	-1,297
úhlová rychlost	k4	-2,933

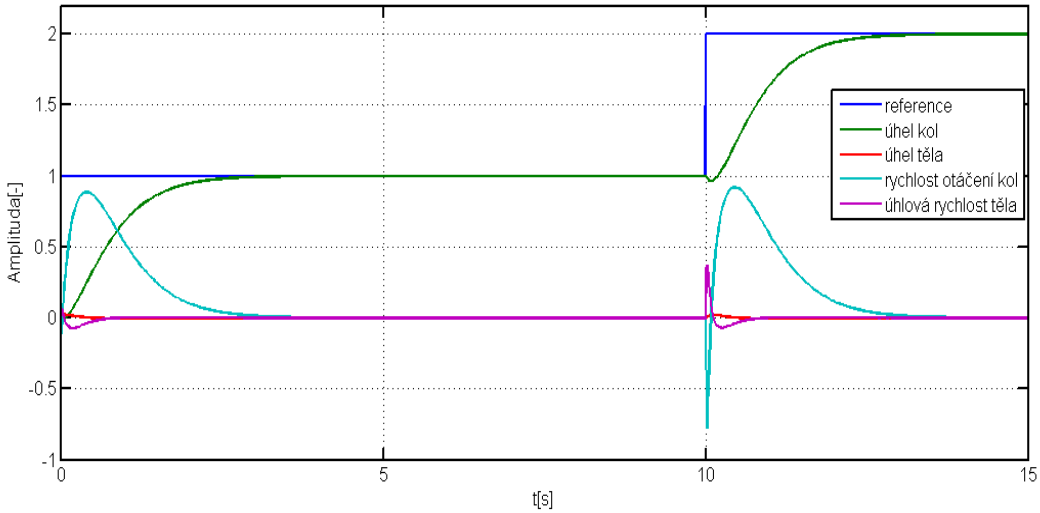


obr. 24 Schéma zapojení stavové zpětné vazby v simulinku

Na obr. 25 je vidět přechodová charakteristika systému se zavedenou stavovou zpětnou vazbou. Z průběhů vidíme, že doba ustálení se pohybuje okolo tří vteřin. Z důvodu větší stability by tato hodnota měla být ještě nižší. Na druhou stranu má regulátor nulovou regulační odchylku bez překmitu. Pro vylepšení jeho vlastností je ještě k regulaci použit PID regulátor z projektu GELway. Jeho parametry jsou v tab. 7. Výsledné parametry regulátoru se nakonec musely odladit až při implementaci programu.

tab.7 Parametry PID regulátoru

Parametry regulátoru	Hodnoty
P	1.2
I	0.25
D	0.1



obr. 25 Průběh s LQR stavovou zpětnou vazbou

3.1.5 Programové řešení úlohy

V této kapitole je popsán postup implementace řídícího systému, který byl navržen v předchozí kapitole. Protože jsem se rozhodl použít program psaný v NXJ k robotovi GELway, bylo zapotřebí zjistit, zda je schopen robota řídit a stabilizovat. Program jsem řádně prostudoval a provedl řadu úprav tak, aby vyhovoval mým potřebám.

3.1.5.1 Implementace programu GELway

Z rozsáhlé dokumentace a videí umístěných na webu youtube jsem byl přesvědčen, že program dokáže robota stabilizovat, dokonce i umožňuje ovládání prostřednictvím bluetooth.

Pro jeho úspěšnou implementaci bylo zapotřebí nainstalovat LeJOS NXJ software a vývojové prostředí NetBeans rozšířené o LeJOS plugin. Podrobný popis instalace je v kapitole 2.5.2. Po úspěšném vytvoření nového projektu a nahrání všech tříd potřebných k běhu programu jsem musel odstranit řadu drobných chyb, které byly způsobeny především jinou verzí softwaru LeJOS NXJ. Většinou bylo zapotřebí změnit importované balíky a některé funkce. Protože projekt GELway používá oproti mému projektu navíc EOPD senzor k detekci vzpřímené polohy a ultrazvukový senzor k detekci překážek, musel jsem je z programu odstranit. To vyžadovalo smazání celé třídy DetectObstacle a provedení změn ve třídách BalanceControler a GELway. Teprve poté bylo možné program zkompilovat a nahrát do kostky s firmwarem NXJ. Ihned po spuštění programu jsem byl vyzván k připojení bluetooth zařízení. Bohužel program do počítače distribuovaný se softwarem GELway nebyl funkční, proto jsem musel odstranit i připojení k bluetooth. Po jeho odstranění se již program rozběhl a byl plně schopný inverzní kyvadlo stabilizovat. Později jsem byl schopen odstranit i potíže s během programu na PC a zajistil jsem tak možnost ovládání inverzního kyvadla prostřednictvím bluetooth. Kompletní programy jsou k dispozici v elektronické příloze.

3.1.5.2 Rozbor jednotlivých tříd programu

V rozboru popíšu podrobně pouze nejdůležitější třídy, u ostatních pouze zmíním jejich funkci. Kompletní program je k nalezení na elektronické příloze.

Třída GELway

Třída GELway je hlavní třídou programu a spouští se ihned po spuštění programu. Hned v úvodu se v ní inicializuje třída CtrlParam, ve které jsou metody sloužící k ovládání robota. Nejdůležitější metodou v třídě je metoda loadNormal, ve které se inicializuje regulátor, bezdrátové připojení, ovládání robota a v poslední řadě i arbitrátor, který je důležitý pro běh celého programu.

```
if(connect)
{
  LCD.clear();
  LCD.drawString("Cekam...", 2, 1);
  BTConnection conn = Bluetooth.waitForConnection();
  br = new BluetoothReader(conn);
  br.start();
}
```

Dle toho, zda byla vybrána možnost připojení bluetooth, program čeká na připojení. Pokud se naváže spojení vytvoří se instance třídy BluetoothReader v níž se vytvoří input a outputstream. Poté se spustí vlákno, které obstarává čtení instrukcí.

```
BalanceController bc = new BalanceController(ctrl,connect);
bc.start();
```

Inicializuje se a spustí se vlákno třídy BalanceController, v níž je implementován regulátor kyvadla.

```
if(connect)
{
    b1 = (Behavior)new GELwayDriver(ctrl, br);
}
else
{
  b1 = new GELwayFollower(ctrl);
}
Behavior[] bArray = {b1};
Arbitrator arby = new Arbitrator(bArray);
arby.start();
```

Na základě toho, zda je kyvadlo připojeno k bluetooth zařízení, se inicializuje chování robota, které je vloženo do Arbitratoru, který po spuštění běží neustále a rozhoduje, které chování systému bude aktivováno. Protože jsem chování robota omezil vždy pouze na jednu možnost, je arbitrátor téměř zbytečný, ale pokud by někdo chtěl rozšířit vlastnosti kyvadla arbitrátor mu to značně usnadní.

Třída BalanceController

V této třídě je hlavním prvkem neustále běžící vlákno obsahující stavovou zpětnou vazbu a PID regulátor, který udržuje kyvadlo ve vzpřímené poloze.

```
double Psi = gyro.getAngle();
double PsiDot = gyro.getAngleVelocity();
double Phi = motors.getAngle() - ctrl.tiltAngle();
double PhiDot = motors.getAngleVelocity();
// Proportional Error
double error = Psi * K_psi + Phi * K_phi + PsiDot * K_psidot + PhiDot
       * K_phidot;
```

Ze senzorů jsou v každém cyklu načítána data, která jsou pak násobena příslušným váhovým koeficientem stavové zpětné vazby. Tento součet se následně rozdělí do tří složek proporční, integrační a derivační.

```
double pw = (error * Kp + deriv_error * Kd + int_error * Ki);
motors.setPower(pw + ctrl.leftMotorOffset(), pw + ctrl.rightMotorOffset());
```

Jednotlivé složky jsou vynásobeny konstantami PID regulátoru a z jejich součtu je vypočítán akční zásah motoru. Podrobný popis k PID regulátoru je ve zdroji [21]. Jeden cyklus trvá přibližně 10ms, což je dostatečně krátká doba k tomu, aby byl regulátor schopen robota stabilizovat. Pokud by se tato doba změnila, dojde ke zhoršení stability kyvadla.

Třída BluetoothReader

Třída umožňuje dálkové ovládání inverzního kyvadla prostřednictvím bluetooth. Vlákno v ní běžící zajišťuje přijímání příkazů z připojeného bluetooth zařízení. Přijímané příkazy jsou v kódové podobě zmačknutého tlačítka na ovládacím zařízení.

Třída CtrlParam

V této třídě jsou umístěny důležité parametry, které umožňují ovládat robota. Jsou zde offsety jednotlivých motorů, ale i offset náklonu robota, který umožňuje jízdu vpřed i vzad. Změna směru jízdy je tedy zajištěna zavedením úmyslné chyby do systému, kdy si program myslí, že došlo k náklonu robota, i když k němu ve skutečnosti nedošlo. Na tuto skutečnost pak patřičně reaguje regulátor, který rozpohybuje robota vpřed či vzad.

Třída GELwayDriver

Hlavní součástí třídy je rozhodovací strom(switch), který vyhodnocuje přijaté příkazy z připojeného bluetooth zařízení. Na základě příkazů pak spouští metody ze třídy MotorDirection.

Třída GyroscopeSensor

V této třídě se inicializuje gyroskopický senzor a vypočítával se v ní pomocí průměrování offset gyroskopu. Protože firmware LeJOS NXJ verze 0.85 dělá tento výpočet samostatně, byl výpočet offsetu ze třídy odebrán.

Třída MotorController

Třída dodává potřebné napětí na motory k tomu aby stabilizovaly kyvadlo. Hodnota napětí je vypočítávána regulátorem ve třídě BalanceController. Třída také udává hodnoty úhlu a rychlosti otáčení motorů.

Třída MotorDirection

Třída prostřednictvím offsetů ze třídy CtrlParam ovládá směr pohybu kyvadla k tomu slouží metody: forward, backward, move, left, right a stop.

3.1.6 Program ovládající kyvadlo

Jde o počítačový program vyvinutý Stevenem Witzandem, který ho využil k ovládání jeho robota GELway. Pracuje na jednoduchém principu čtení kláves a odesíláním jejich kódové hodnoty prostřednictvím bluetooth streamu. Avšak i pro jeho správnou funkci

jsem musel zaměnit některé funkce, které nebyly kompatibilní s novou verzí softwaru LeJOS NXJ. K dosavadnímu programu jsem taktéž přidal funkci reset, která zastaví běh motorů kyvadla stiskem klávesy enter. Spustitelný Program včetně projektu je k dispozici v elektronické příloze.

Program je schopen ovládat robota na volném prostranství až do vzdálenosti deseti metrů. V zarušeném prostředí může dojít ke ztrátě komunikace, pak je nutné restartovat program v inverzním kyvadle a navázat novou komunikaci.

3.1.7 Závěr k úloze inverzní kyvadlo

Byla úspěšně provedena realizace a implementace řízení úlohy inverzní kyvadlo. Realizované řízení je dostatečně robustní, aby dokázalo udržet kyvadlo ve vzpřímené poloze i při vnějších rušivých vlivech. Kyvadlo je s ním dokonce schopné na přídavné rampě uvézt závaží až o hmotnosti půl kila. Dále se podařilo zprovoznit bezdrátové ovládání kyvadla z počítače.

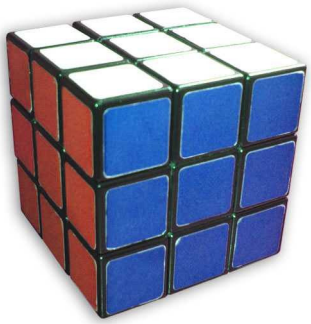
3.2 Úloha Rubikova kostka

V této kapitole jsem se zabýval robotem ze stavebnice Lego Mindstorms, který je schopen samostatně složit libovolně zamíchanou Rubikovu kostku. Robot a program, ze kterého jsem v úloze vycházel byl vytvořen Hansem Anderssonem. Protože tato verze byla značně chybová, musel jsem na ní provést v rámci možností optimalizaci. V kapitole popíšu konstrukci a některá má vylepšení původní konstrukce robota. Dále popíšu program robota včetně implementace tří různých barevných senzorů.

3.2.1 Hlavolam Rubikova kostka

Jde o mechanický hlavolam, který byl vynalezen v 70. letech 20. století Ernem Rubikem. Během několika let se stala dosud nejprodávanějším hlavolamem na světě. Nejběžnějším typem kostky je varianta 3x3x3 obr. 25, jejíž povrch tvoří padesát čtyři kostiček, které jsou v šesti různých barevných odstínech. Kostka se skládá z dvaceti šesti dílů z toho je osm rohů, šest středů a dvanáct hran. Protože se středy nepohybují, určují barvu vrstvy. Kostka může být zamíchána ve více než 43 trilionech(10^{18}) kombinací, přičemž možné řešení je pouze jedno. Ke skládání hlavolamu bylo vymyšleno několik různých postupů, které se liší především počtem použitých algoritmů a dobou potřebnou k jeho složení. Mezi nejznámější metody patří „vrstva za vrstvou“(layer by layer), „rohy první“(corner first) a Jessica Fridrich’s. Počítačovou simulací bylo zjištěno, že libovolně zamíchaná kostka může být vyřešena dvaadvaceti tahy nebo méně. Člověk udělá k vyřešení kostky v průměru padesát pět tahů.

Ve skládání kostky se pořádají i celosvětové soutěže zvané Speedcubing, kde se jedná především o to složit kostku v co nejkratším čase. Skládají se kostky jak standardní velikosti 3x3x3, tak i 2x2x2, 4x4x4 a 5x5x5. Mezi další disciplíny patří skládání poslepu, jednou rukou nebo složení hlavolamu co nejméně tahy. Současný rekord ve skládání kostky 3x3x3 drží s časem 7.08s Erik Akkersdijk.



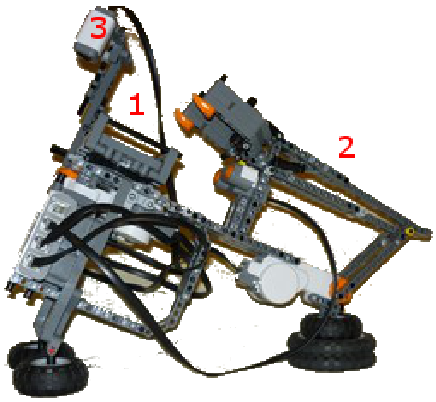
obr. 26 Rubikova kostka

3.2.2 Konstrukce

Jak již bylo řečeno hlavním tvůrcem konstrukce obr. 27 je Hans Andersson. Návod k sestavení základní konstrukce je ve zdroji [25] nebo v elektronické příloze. Robot se skládá ze tří hlavních částí. První je kolébka pro horizontální otáčení kostky na obr. 27 označena číslem 1. Druhou částí je rameno, které otáčí kostku vertikálně, na obrázku je označeno číslem 2. Poslední částí je rameno s barevným senzorem označeno číslem 3.

Kolébka je ovládána motorem připojeným k výstupu A a otáčí se o 360°. Otáčecí rameno je ovládáno motorem připojeným k výstupu B a rameno barevného senzoru je ovládáno motorem z výstupu C.

Ke správnému chodu má robot tři senzory. Senzor dotyku připojený na vstup 1. Ten zajišťuje detekci dorazové polohy ramena barevného senzoru. K výstupu 2 je připojen barevný senzor, který slouží ke skenování barev z kostky. Posledním senzorem je ultrazvukový senzor připojený k portu 3. Ten detekuje umístění Rubikovy kostky do kolébky.

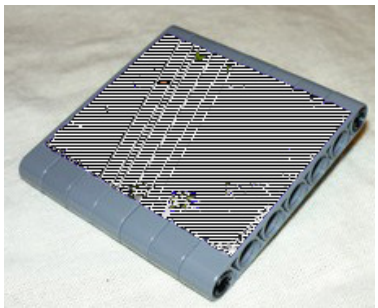


obr. 27 Konstrukce robota

3.2.2.1 Úpravy konstrukce

Protože docházelo k častému vypadávání kostky při jejím vertikálním otáčení, byl jsem nucen udělat drobné změny v dosavadní konstrukci. První ze dvou úprav se týká kolébky pro kostku a druhá celkového náklonu robota.

Aby nedocházelo z zadrhávání kostky při otáčení, Hans Andersson doporučuje na dno kolébky vložit tvrdý lesklý papír např. z pohledu o rozměrech 70x70mm, aby kostka lépe klouzala. Mně se toto vylepšení neosvědčilo, proto jsem ze dna kolébky vyndal téměř všechny součástky a nahradil je deskou vytvořenou ze součástek lega a přelepenou vhodnou lesklou nálepkou obr. 28. Návod k jejímu sestavení je v elektronické příloze.



obr. 28 Deska do kolébky

I přes předchozí opatření docházelo k nedotáčení kostky nebo k jejímu strhnutí ramenem zpět. Zdálo se mi, že je to pravděpodobně způsobeno náklonem robota, proto jsem metodou pokus omyl podkládal zadní část robota a měnil tak úhel jeho náklonu. Nakonec jsem úhel změnil z původních 25° na 17° a snížil tak chybovost otáčení kostky. Výsledná konstrukce podložení je na obr. 29. Návod k jeho sestavení je též v elektronické příloze.

obr. 29 Podložení zadní části robota

Zjistil jsem, že síla motorů je úměrná nabití baterií robota, což se nejvíce projeví u motoru B, který otáčí kostku. Při téměř vybité baterii motor nebyl schopen kostkou otočit. Jelikož je program optimalizován na plně nabitý akumulátor, doporučuji používat akumulátor včetně připojeného adaptéru do sítě.

Jednou za čas doporučuji promazat silikonovým olejem, jak Rubikovu kostku, tak i všechny části robota, které s ní přijdou do styku.

3.2.3 Programové řešení úlohy

Programů k řešení úlohy je několik, protože je robot schopen pracovat se třemi různými senzory. Všechny programy jsou napsané v jazyce NXC. Senzor světla od Lega(senzor Lego) je v programovém balíku společně s barevným senzorem od HiTechnic(senzor HT). Programový balík pro tyto senzory funguje na firmwaru verze 1.05. Pro barevný senzor Lego V2.0(senzor Lego V2) jsem vytvořil samostatný programový balík, který je založen na balíku předchozím a funguje s firmwarem verze 1.28. Liší se především odlišnými instrukcemi pro čtení hodnot z barevných senzorů. Každý balíček obsahuje dva programy, kalibrační program sloužící k naučení hodnot barev a polohy ramena se senzorem vůči středu kostky a program, který Rubikovu kostku dle naučených barevných hodnot kostku skládá.

Jak jsem již napsal, výše původní program byl napsán Hansem Anderssonem, který ho publikoval a dal volně ke stažení na webové stránce [25]. Programový balík by měl být schopen bezproblémově vyřešit hlavolam Rubikovy kostky jak se senzorem Lego, tak i se senzorem HT. Stažené programy jsem nahrál do NXT kostky pomocí softwaru BricxCC. Provedl jsem kalibraci na složené Rubikově kostce. Po úspěšné kalibraci jsem spustil hlavní program, který by měl v první fázi oskenovat barevná políčka kostky a přiradit jim pozici na kostce. V druhé fázi rozliší jednotlivé barvy políček a ve třetí vypočítá tři různá řešení podle toho, z jaké strany kostky začne hlavolam řešit. Ze tří řešení vybere to, co má nejméně tahů a použije ho v poslední fázi k vyřešení hlavolamu.

Při úvodním testování na složené kostce jsem si všiml hned několika problémů, které bylo třeba odstranit. Prvního nedostatku co jsem si všiml, byla špatná detekce kostky ultrazvukovým senzorem. Kvůli tomuto problému často docházelo k bezdůvodnému zastavení běhu programu. Další problém se vyskytl při výpočtu řeše-

ní, kdy program měl u složené kostky vypočíst všechna řešení rovna nule. Místo toho docházelo k jeho zaseknutí, zacyklení nebo k výpočtu nenulových řešení, která nevedla ke správnému vyřešení hlavolamu. Hans Andersson upozorňuje, že by se tento problém měl týkat špatné vzdálenosti nebo nevhodného osvětlení, ale ať jsem měnil podmínky jakkoli, nevedlo to k úspěšnému řešení. Optimalizace programu tedy byla nevyhnutelná.

3.2.3.1 Optimalizace programů

Prvním krokem bylo odstranění problému s detekcí Rubikovy kostky v kolébce. Detekování kostky se provádí ultrazvukovým senzorem v metodách WaitForCube a CheckCube. Jednoduchým testem pomocí prostředku view v inteligentní kostce jsem zjistil, že hodnoty pro detekci vzdálenosti nastavené v metodách jsou nedostačující, proto jsem je zvýšil z 9cm na 13cm. Tímto krokem jsem odstranil problém s úvodní detekcí Rubikovy kostky, ale i tak nadále docházelo k chybám při běhu programu. Chyba byla už jen v metodě CheckCube, která měla za úkol zjišťovat přítomnost kostky v kolébce po jejím vertikálním otočení. Protože metoda zastavovala program, i když bylo vše v pořádku a naopak nezastavila program, když došlo k vypadnutí kostky, musel jsem ji z běhu programu odstranit úplně.

Po odstranění problémů s detekcí kostky došlo na řešení problému nejzásadnějšího, a to problému s rozlišováním barev. Protože senzor HT nemá narozdíl od senzoru Lego potíže s rozlišováním barev, zvolil jsem ho pro odstranění tohoto problému. V prvním kroku jsem se rozhodl porovnat hodnoty barev získané kalibrací s hodnotami snímanými při běhu hlavního programu. Zjistil jsem, že se tyto hodnoty při stejných světelných podmínkách liší. Nejvíce se lišila hodnota modré barevné složky, ale tento jev byl již zaznamenán v kapitole 1.3.3.10. Protože senzor snímal každou hodnotu barevného políčka pouze jednou, rozhodl jsem se zavést průměrování jak při kalibraci, tak i při běhu hlavního programu. Nastavil jsem snímání barevných políček tak, aby bylo z každého políčka vzato deset vzorků a z nich byl vypočítán průměr. Tento krok zvýšil stálost červené a zelené barevné složky, ale u modré stále docházelo k velkým odchylkám, proto jsem jí nakonec z algoritmu pro rozlišování barev odstranil. I přes to, že program k rozlišování používá pouze dvě barevné složky ze tří, je rozlišování dle naučených vzorků velmi přesné. Po připojení senzoru Lego se ukázalo, že tyto kroky odstranily

problémy i s tímto senzorem. Stejné kroky jsem provedl i v programovém balíčku pro senzor Lego V2 s výjimkou odstranění modré barevné složky.

Poslední problém spočíval ve špatném vertikálním otáčení kostky. Ten jsem částečně vyřešil optimalizací síly motoru, který pohání otáčecí rameno.

3.2.3.2 Rozbor programu kalibrace

V úvodu programu se inicializují senzory robota. Poté je uživatel vyzván k umístění ramena na střed kostky pro zkalibrování polohy senzoru vůči středu kostky. Po zkalibrování polohy začne skenování barev z jednotlivých políček.

```
for(int i=0;i<10;i++)
{
while(!ReadSensorHTRawColor(IN_2, r, g, b));
tempr+=r;
tempg+=g;
tempb+=b;
Wait(10);
}
```

Ve smyčce funkce ReadSensorHTRawColor načte desetkrát hrubé hodnoty každého z políček. Pro senzor Lego V2 je použita funkce ReadSensorColorEx. Součet těchto hodnot je vydělen počtem vzorků a použit k výpočtu hodnot barvy strany.

```
DeleteFile("tiltedtwister.clr")
CreateFile("tiltedtwister.clr",2+6*3*2,handle);
Write(handle,tachoCenter);
for(int face=0;face<6;face++)
{
  ClearScreen();
  Wait(200);
  int value;
  value=red[face]/9;
  Write(handle,value);
  value=green[face]/9;
  Write(handle,value);
  value=blue[face]/9;
  Write(handle,value);
}
CloseFile(handle);
```

Průměrná hodnota vzorků barvy je uložena do souboru tiltedtwister.clr, pro pozdější použití hlavním programem.

3.2.3.3 Rozbor hlavního programu

Skenování políček kostky probíhá podobně jako v kalibračním programu s tím rozdílem, že je zaznamenávána navíc poloha jednotlivých políček na kostce. Po ukončení skenování se provede rozlišování jednotlivých barev v metodách ResolveColors.

```
rgb centerColor,cubieColor;
centerColor = refColor[center];
SensorColorGet(cubie,CENTER,cubieColor);
unsigned int cost = (centerColor.red - cubieColor.red) *
(centerColor.red - cubieColor.red) + (centerColor.green - cubieColor.green)
* (centerColor.green - cubieColor.green);
```

Hodnoty z každého políčka jsou porovnány zjednodušenou metodou nearest neighbor se vzorkem barev načtených ze souboru tiltedtwister.clr. Vzorek, který bude mít nejnižší hodnotu z výpočtů, je barvou odpovídající políčku. Takto program rozliší všechna políčka na kostce.

Na rozpoznaná políčka použije algoritmy pro vyhledání tří různých řešení dle toho jakou stranu kostky bude robot považovat za vrchní. Použité algoritmy jsou z metody corner first. Vyhledané řešení není nejoptimálnější, ale i tak vede ke správnému řešení. Pro vyhledání optimálního řešení by robot potřeboval sofistikovanější program, který se však již nevejde do paměti inteligentní kostky NXT a robot by musel být řízen z počítače. Z vypočtených řešení je vybráno to nejrychlejší a následně jsou provedeny tahy potřebné k vyřešení hlavolamu. V průběhu řešení je na display NXT zobrazován počet zbývajících tahů do úspěšného vyřešení. Po složení hlavolamu se program ukončí.

3.2.4 Závěr k úloze Rubikova kostka

Optimalizací se podařilo docílit bezchybné detekce barev senzorem HT a Lego V2. Pro senzor Lego je zapotřebí přelepit některé barvy na kostce, pak již funguje též. Taktéž se mi podařilo odstranit potíže se zasekáváním běhu programu, které byly způsobeny špatným vyhodnocováním stavu ultrazvukového senzoru. Úpravou konstrukce a optimalizací dynamiky otáčecího ramena se mi podařilo snížit riziko vypadnutí kostky z kolébky. Nepodařilo se mi však odstranit problém s občasným přetáčením kolébky,

které způsobí to, že otáčecí rameno narazí na kostku a neotočí ji. Tato chyba se projevovala zejména při použití firmwaru verze 1.28. U firmwaru 1.05 se za použití stejných příkazů tato chyba neprojevila. Z toho usuzuji, že jde o chybu ve firmwaru a bude pravděpodobně odstraněna jeho novou verzí.

4 Webové stránky k realizovaným úlohám

4.1 Obsah

Obsahem této kapitoly je popis obsahu a tvorby dvou webových stránek vytvořených pro úlohy popsané v kapitole 3. Webové stránky jsou určeny pro web Roboti [27], který je umístěn na adrese http://support.dce.felk.cvut.cz/roboti/.

4.2 Úvod

Web Roboti byl vytvořen v roce 2009 jako součást bakalářské práce Využití robota LEGO MINDSTORMS při výuce [1], které je autorem Pavel Trojánek. Stránky slouží pro účely předmětu A3B99RO Roboti a také pro informační a organizační potřeby Robo soutěže. Mé stránky doplňují web o základní informace o mnou realizovaných úlohách. Obsahují základní popis úlohy, její konstrukci a základní princip funkce robota. Dále jsou na stránkách umístěny odkazy vedoucí k funkčním programům pro dané úlohy. K programům je na stránkách umístěn i popis jejich instalace do inteligentní kostky. Ve spodu obou stránek jsou umístěny důležité odkazy k dané úloze.

4.3 Zpracování

Webové stránky jsou rozděleny do čtyř částí které jsou definované kaskádovým stylem. Části jsou hlavička, menu, patička a hlavní okno. Protože chci na webu prezentovat pouze určitý obsah a né kompletně nové stránky, vytvořím php soubor pouze pro hlavní okno. Ostatní části se nahrávají automaticky použitím php skriptu. Protože je web prezentován jak v české podobě tak i v anglické, bylo nutné pro každou úlohu vytvořit dva soubory, které se liší pouze koncovkou v názvu _cs nebo _en a jazykem textu. Adresářová struktura webu je jednoduchá všechny soubory stránek jsou umístěny v kořenovém adresáři, obrázky jsou v adrasáři img a ostatní soubory v adresáři file. Protože je tvorba webových stránek podrobně popsána v dokumentu [1], nebudu se zde více k tvorbě webových stránek rozepisovat. Vytvořené stránky je možno libovolně aktualizovat, tak aby co nejvíce vyhovovaly potřebám předmětu. Vytvořené soubory pro hlavní okno naleznete v elektronické příloze.

4.4 Závěr k webovým stránkám

Vytvořené stránky slouží především k tomu, aby si jejich návštěvník byl s jejich pomocí schopen úhohu zkonstruovat, zprovoznit a pochopit základní principy úlohy.

5 Závěr

V práci se mi podařilo splnit všechny pokyny k vypracování. V úvodní části práce jsem popsal hardwarové a softwarové možnosti stavebnice. Pozornost jsem věnoval jak inteligentní kostce NXT, tak i ostatním aktivním částem stavebnice, aby si bylo možno udělat obrázek o jejich využitelnosti. Protože je kompatibilních senzorů hodně vybral jsem pouze ty, které mají podle mě nejširší uplatnění a nebo jsou použity v této práci. Dokonce jsem provedl srovnání dvou barevných senzorů, dle kterého se čtenář může rozhodnout, který ze senzorů bude pro realizaci jeho úlohy vhodnější.

V druhé části jsem analyzoval možnosti programování stavebnice a protože je možností opravdu mnoho a nejsou všechny volně dostupné, zaměřil jsem se pouze na ty volně dostupné a nejrozšířenější v komunitě Lego Mindstorms. Ke všem rozebíraným programovacím jazykům jsem vytvořil i základní program Helloword, aby si každý mohl udělat alespoň částečnou představu o programování v daném jazyce.

Ve třetí kapitole popisuji realizaci dvou speciálních úloh. Podařilo se mi realizovat funkční a dostatečně robustní inverzní kyvadlo, které se dá dálkově ovládat prostřednictvím počítače. V úloze Rubikovy kostky se mi povedlo implementovat v úloze všechny tři barevné senzory. Dále jsem optimalizoval detekci a rozpoznávání barev, podstatně jsem snížil riziko vypadnutí kostky z kolébky, ale neeliminoval jsem ho.

K realizovaným úlohám jsem vytvořil webové stránky, které informují, obsahují popis úloh, návod na konstrukci a stručný popis principu funkce.

Použitá literatura

Bakalářské a diplomové práce

[1] TROJÁNEK, Pavel. Využití robota LEGO MINDSTORMS při výuce. Praha, 2009. 94 s. Bakalářská práce. ČVUT, FEL, Katedra řídicí techniky. Dostupné z WWW: <http://support.dce.felk.cvut.cz/e-kurzy/file.php/24/BP_Pavel_Trojanek.pdf>.

[2] PROCHÁZKA, Michal. Řízení robota MINDSTORMS NXT pomocí PC. Zlín, 2008. 57 s. Diplomová práce. Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky. Dostupné z WWW: <https://www.stag.utb.cz/apps/stag/dipfile/index.php?download=9407>.

Hardwarové a softwarové vybavení

[3] LEGO Mindstorms NXT Hardware Developer Kit. [s.l.] : [s.n.], ©2006. 25 s. Dostupné z WWW: <http://mindstorms.lego.com/en-us/support/files/default.aspx>.

[4] HURBAIN, Philippe. Philo's Home Page [online]. 10 january 2000, Last updated 25 February 2010 [cit. 2010-05-13]. Dostupné z WWW: <http://www.philohome.com/>.

[5] Robotics Introduction [online]. 27/02/2009, last update 15/03/2009 [cit. 2010-05-13]. NXT sound sensor. Dostupné z WWW: <http://www.convict.lu/htm/rob/NXT_sound_sensor.htm>.

[6] LEGO Engineering [online]. ©2010 [cit. 2010-05-13]. NXT Sensors. Dostupné z WWW: <http://www.legoengineering.com/nxt-sensors-2.html>.

[7] ETH - TIK - Computer Engineering and Networks Laboratory [online]. July 2006 [cit. 2010-05-13]. SA Lego Mindstorms NXT. Dostupné z WWW: <http://www.tik.ee.ethz.ch/mindstorms/sa_nxt/index.php?page=tests_us>.

[8] LEGO Mindstorms NXT User Guide. [s.l.] : [s.n.], ©2006. 66 s.

[9] SOLDAAT, Xander. HiTechnic Sensor Multiplexer : Programmer's Guide [online]. [s.l.] : [s.n.], November 25, 2009 [cit. 2010-05-13]. Dostupné z WWW: <http://www.breigh.com/xander/HTSMUX-Tutorial-LATEST.pdf>.

[10] HiTechnic Products [online]. ©2001 [cit. 2010-05-13]. Dostupné z WWW: <http://www.hitechnic.com/>.

[11] LEGO® MINDSTORMS® NXT Executable File Specification [online]. [s.l.] : [s.n.], 2009 [cit. 2010-05-13]. Dostupné z WWW: <http://mindstorms.lego.com/en-us/support/files/default.aspx>.

[12] CHIKAMASA, Takashi. NxtOSEK [online]. ©2007 [cit. 2010-05-13]. Dostupné z WWW: <http://lejos-osek.sourceforge.net/>.

Programovací jazyky

[13] LeJOS, Java for Lego Mindstorms [online]. 2006 [cit. 2010-05-13]. Dostupné z WWW: <http://lejos.sourceforge.net/>.

[14] LeJOS, Java for Lego Mindstorms [online]. 2006 [cit. 2010-05-13]. Overview (leJOS NXJ API documentation). Dostupné z WWW: <http://lejos.sourceforge.net/nxt/nxj/api/index.html>.

[15] LeJOS, Java for Lego Mindstorms [online]. 2006 [cit. 2010-05-13]. The leJOS NXJ Tutorial. Dostupné z WWW: <http://lejos.sourceforge.net/nxt/nxj/tutorial/index.htm>.

[16] HASSENPLUG, Steve. Team Hassenplug [online]. c2007, updated Feb 29th, 2008 [cit. 2010-05-13]. NXT Programming Software. Dostupné z WWW: <http://www.teamhassenplug.org/NXT/NXTSoftware.html>.

[17] Bricx Command Center 3.3 [online]. 2001 [cit. 2010-05-13]. Dostupné z WWW: <http://bricxcc.sourceforge.net/>.

[18] HANSEN, John. Not eXactly C : Programmer. [s.l.] : [s.n.], October 10, 2007. 118 s.

Inverzní kyvadlo

[19] WIDMER, Lukas; BRUMANN, Reto. NXT Standalone [online]. 2007 [cit. 2010-05-12]. Dostupné z WWW: <http://www.brumann-art-production.ch/NXT_Standalone/>.

[20] WATANABE, Ryo. NXTway-G [online]. March 2007. 2006 [cit. 2010-05-12]. Dostupné z WWW: <http://web.mac.com/ryo_watanabe/iWeb/Ryo's%20Holiday/NXTway-G.html>.

[21] SLUKA, Jim. Jim Sluka's Page [online]. 2009 [cit. 2010-05-12]. PID Controller For Lego Mindstorms Robots. Dostupné z WWW: <http://www.inpharmix.com/jps/PID_Controller_For_Lego_Mindstorms_Robots.html>.

[22] Techbricks [online]. 13. březen 2009 [cit. 2010-05-13]. Self-balancing Segway NXTway-Gyro bot & wireless PS2 controls | My NXT projects. Dostupné z WWW: <http://www.techbricks.nl/My-NXT-projects/nxt-self-balancing-segway-nxtway-robot.html>.

[23] WITZAND, Steven. Coordinated LEGO Segways. [s.l.], November 2009. 180 s. Bakalářská práce. School of Information Technology & Electrical Engineering, The University of New South Wales, Australian Defence Force Academy. Dostupné z WWW: <http://code.google.com/p/gelway/>.

[24] YAMAMOTO, Yorihisa . NXTway-GS Model-Based Design : control of self-balancing two-wheeled robot built with LEGO Mindstorms NXT [online]. [s.l.] : [s.n.], February 2008, updated May 2009 [cit. 2010-05-13]. Dostupné z WWW: <http://www.mathworks.com/matlabcentral/fileexchange/19147>.

Rubikova kostka

[25] ANDERSSON, Hans. Tilted Twister [online]. 2008 [cit. 2010-05-13]. Dostupné z WWW: <http://tiltedtwister.com/>.

Všeobecné

[26] Nxtasy.org | Building a LEGO Mindstorms NXT Community Brick by Brick [online]. June 2006, May 2010 [cit. 2010-05-13]. Dostupné z WWW: <http://nxtasy.org/>.

[27] TROJÁNEK, Pavel. ROBOTI | ČVUT - Katedra řídící techniky [online]. 2009 [cit. 2010-05-13]. Dostupné z WWW: <http://support.dce.felk.cvut.cz/roboti/index.php?id=news>.

[28] LEGO.com MINDSTORMS [online]. 2010 [cit. 2010-05-13]. Dostupné z WWW: <http://mindstorms.lego.com/en-us/default.aspx>.

Wikipedia

[29] Lego Mindstorms NXT. In Wikipedia : the free encyclopedia [online]. St. Petersburg (Florida) : Wikipedia Foundation, 17 July 2006, last modified on 13 May 2010 [cit. 2010-05-24]. Dostupné z WWW: <http://en.wikipedia.org/wiki/Lego_Mindstorms_NXT>.

[30] Lego Mindstorms NXT 2.0. In Wikipedia : the free encyclopedia [online]. St. Petersburg (Florida) : Wikipedia Foundation, 9 July 2009, last modified on 24 May 2010 [cit. 2010-05-24]. Dostupné z WWW: <http://en.wikipedia.org/wiki/Lego_Mindstorms_NXT_2.0>.

[31] Rubics cube. In Wikipedia : the free encyclopedia [online]. St. Petersburg (Florida) : Wikipedia Foundation, 1 November 2001, last modified on 21 May 2010 [cit. 2010-05-24]. Dostupné z WWW: <http://en.wikipedia.org/wiki/Rubics_cube>.

[32] Linear-quadratic regulator. In Wikipedia : the free encyclopedia [online]. St. Petersburg (Florida) : Wikipedia Foundation, 30 May 2006, last modified on 5 March 2010 [cit. 2010-05-24]. Dostupné z WWW: <http://en.wikipedia.org/wiki/Linear-quadratic_regulator>.

[33] Lagrangeovská formulace mechaniky. In Wikipedia : the free encyclopedia [online]. St. Petersburg (Florida) : Wikipedia Foundation, 24. 11. 2007, last modified on 9. 2. 2010 [cit. 2010-05-24]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/Lagrangeovská_formulace_mechaniky>.

Seznam obrázků

obr. 1 Programovatelná kostka a základní příslušenství ... 8
obr. 2 Schéma zapojení inteligentní kostky ... 10
obr. 3 Schéma výstupního portu ... 11
obr. 4 Schéma vstupního portu ... 12
obr. 5 Schéma master/slave komunikace mezi kostkami ... 13
obr. 6 Alokace pixelů na display ... 14
obr. 7 Servomotor ... 15
obr. 8 Průběh na A/D převodníku při použití aktivního senzoru ... 16
obr. 9 Dotykový senzor ... 18
obr. 10 Senzor zvuku ... 18
obr. 11 Ultrazvukový senzor ... 19
obr. 12 Senzor světla ... 19
obr. 13 Barevný senzor Lego V2.0 ... 20
obr. 14 Barevný senzor HiTechnic s barevnou škálou ... 21
obr. 15 Naměřené hodnoty senzorem Lego ... 22
obr. 16 Naměřené hodnoty senzorem HT ... 23
obr. 17 Gyroskop a náznak osy snímání rychlosti otáčení ... 23
obr. 18 Senzorový multiplexor ... 24
obr. 19 Program HelloWorld v NXT-G ... 27
obr. 20 Inverzní kyvadlo ... 33
obr. 21 Vstupy a výstupy ... 34
obr. 22 Ilustrativní znázornění konstant a proměnných ... 35
obr. 23 Inverzní kyvadlo v souřadném systému ... 36
obr. 24 Schéma zapojení stavové zpětné vazby v simulinku ... 38
obr. 25 Průběh s LQR stavovou zpětnou vazbou ... 39
obr. 26 Rubikova kostka ... 46
obr. 27 Konstrukce robota ... 46
obr. 28 Deska do kolébky ... 47
obr. 29 Podložení zadní části robota ... 47

Seznam tabulek

tab.1 Barvy s číselným kódem ... 21
tab.2 Naměřená data k prvnímu pokusu ... 22
tab.3 Porovnání dostupných softwarů ... 26
tab.4 Nastavení proměnných prostředí ... 31
tab.5 Konstanty modelu ... 35
tab.6 Koeficienty stavové zpětné vazby ... 38
tab.7 Parametry PID regulátoru ... 39

Obsah elektronické přílohy

- Programy včetně zdrojových kódů
- Manuály k sestavení úloh
- Firmwary k inteligentní kostce
- Pdf verze dokumentu