



TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií ■

Automaticky řízený model vozu s přísavným systémem

Ročníkový projekt

Studijní program: B2646 – Informační technologie
Studijní obor: 1802R007 – Informační technologie

Autor práce: **Daniel Varnuška**
Vedoucí práce: Ing. Jan Koprnický, Ph.D.





TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Mechatronics, Informatics
and Interdisciplinary Studies ■

Automatically controlled vehicle model with suction system

Project report

Study programme: B2646 – Information technology
Study branch: 1802R007 – Information technology

Author: **Daniel Varnuška**
Supervisor: Ing. Jan Koprnický, Ph.D.



Tento list nahradte
originálem zadání.

Prohlášení

Byl jsem seznámen s tím, že na můj ročníkový projekt se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasaahuje do mých autorských práv užitím mého ročníkového projektu pro vnitřní potřebu TUL.

Užiji-li ročníkový projekt nebo poskytnu-li licenci k jeho využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Ročníkový projekt jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mého ročníkového projektu a konzultantem.

Současně čestně prohlašuji, že tištěná verze práce se shoduje s elektronickou verzí, vloženou do IS STAG.

Datum:

Podpis:

ZADÁNÍ BAKALÁŘSKÉHO PROJEKTU

Jméno a příjmení: **Daniel Varnuška**
Název práce: **Automaticky řízený model vozu s přísavným systémem**
Zadávající katedra: **Ústav mechatroniky a technické informatiky**
Vedoucí práce: **Ing. Jan Koprnický Ph.D.**
Rozsah práce: **15–20 stran**

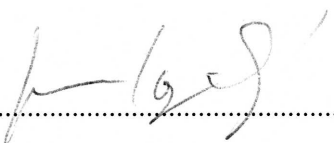
Zásady pro vypracování:

1. Provedte řešení tématu pohybu vozů nebo modelů vozů po vertikálních plochách.
2. Zprovozněte model vozu s přísavným systémem řízený pomocí Arduino Fio.
3. Navrhnete úpravu modelu doplněním vhodných snímačů k jeho autonomnímu řízení.
4. Pro sepsání závěrečné zprávy projektu použijte sázecí systém LaTeX a šablonu tulthesis.

Seznam odborné literatury:

- [1] NAJMAN, Petr. Automaticky řízený model vozu s přísavným systémem. Liberec, 2015. Bakalářská práce. Fakulta mechatroniky, informatiky a mezioborových studií Technické univerzity v Liberci. Vedoucí baka
- [2] Arduino Fio © 2016 Arduino [online]. [cit. 2015-10-09]. Dostupné z:
<https://www.arduino.cc/en/Main/ArduinoBoardFio>
- [3] SATRAPA, Pavel. LaTeX pro pragmatiky [online]. TUL a CESNET, 2011 [cit. 2016-10-09]. Dostupné z:
<http://www.nti.tul.cz/~satrapa/docs/latex/>
- [4] SATRAPA, Pavel. Balík tul pro LaTeX [online]. Liberec, 2015 [cit. 2016-10-09]. Dostupné z:
<http://www.nti.tul.cz/~satrapa/vyuka/latex-tul/>

V Liberci dne 10.10.2016

..... 

Abstrakt

Tato práce popisuje analýzu a postup pro přidání akcelerometru do automaticky řízeného modelu auta s přísavným systémem, který umožňuje modelu jezdit po vertikálních plochách, tak aby přísavný ventilátor byl ovládán dle náklonu modelu. Práce navazuje na bakalářskou práci pana Petra Najmana, který sestavil z zakoupeného modelu na dálkové ovládání model, opatřený řídicí jednotkou Arduino Fio, optickým senzorem CNY70 a dvěma akumulátory. Model byl vybaven o snímač zrychlení ADXL337. Na zprovoznění modelu bylo vyzkoušeno několik algoritmů a byla testována závislost výkonu přísavného ventilátoru na akcelerometru tak, aby ventilátor dodával dostatečný podtlak pod model podvozku, aby se stále udržel, ale zároveň v klidu nevybíjel akumulátory. Algoritmy byly napsány v prostředí Arduino IDE, jazykem vycházejícím z jazyka C/C++.

Abstract

English abstract

Poděkování

Vám poděkování a lásku vám.

Obsah

Seznam zkratek	8
1 Úvod	9
2 Autonomní vozidla	10
2.1 Senzory	10
2.2 Plánování	11
2.3 Roboti s podobnou konstrukcí	12
3 Model vozu	14
3.1 Řídicí deska Arduino Fio	15
3.2 Akumulátory	15
3.3 Akcelerometr ADXL337	16
4 Návod pro použití modelu	19
5 Algoritmy pro pohyb	20
5.1 Jízda podle čáry	20
5.2 Jízda ve válci	20
6 Testování modelu	25
6.1 Testování zapojení akcelerometru ADXL337	25
6.2 Testování přísavného ventilátoru	25
7 Závěr	30
Literatura	30

Seznam zkratek

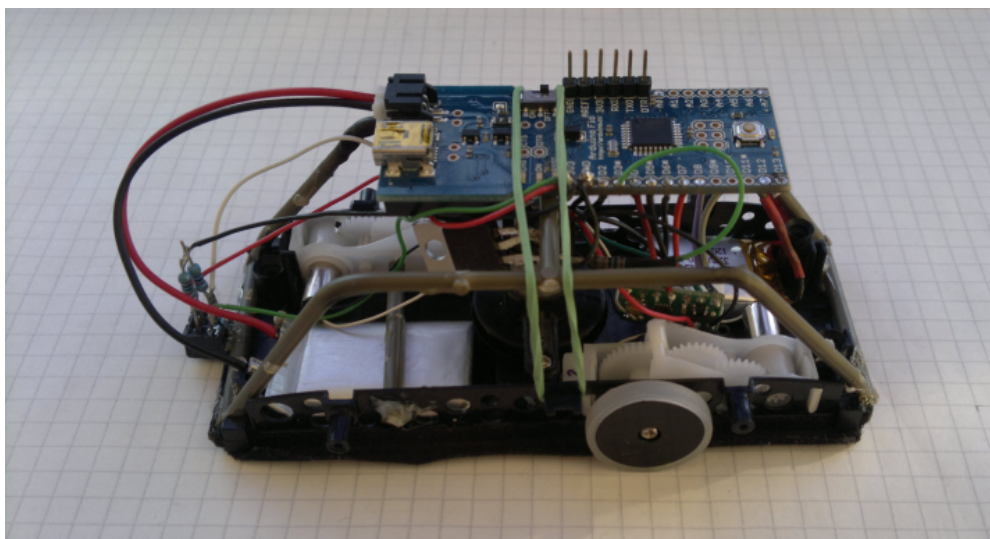
GND	Ground (uzemění)
GPS	Global Positioning System (globální polohový systém)
IDE	Integrated Development Environment (vývojové prostředí)
IR	Infračervené záření
Li-Pol	Lithium polymerový akumulátor
NP	Nedeterministicky polynomiální problém
ON	Online
PC	Personal Computer (osobní počítač)
PWM	Pulse Width Modulation (pulzně šířková modulace)
USB	Universal Serial Bus (univerzální sériová sběrnice)

1 Úvod

Cílem tohoto projektu je pokračování v práci s modelem auta sestrojeném v rámci bakalářské práce [2]. Vozidlo má na podvozku dva stejnosměrné motory pro pohyb a jeden stejnosměrný motor pro přísavný systém, který vytváří podtlak na ploše podvozku modelu. V modelu je již zakomponovaná řídicí jednotka Arduino Fio, dva akumulátory a čidlo odrazivosti. Model můžeme vidět na obrázku 1.1.

U modelu budou zkontrolovány všechny jeho funkční prvky, jelikož byl rok odstaven bez provozu. Model bude doplněn o snímač zrychlení. Dále jsou navrženy algoritmy pro ovládání modelu v různých situacích, jako sledování čáry a jízda ve válci. Model, který mi byl svěřen již umí jezdit ve všech horizontálních i vertikálních plochách, jelikož je vybaven přísavným systémem, který vytváří dostatečný podtlak, aby se udržel na ploše i vzhůru nohama. S ním zvládá auto jezdit po stropě. V práci je postupně zpracovaná problematika závislosti výkonu ventilátoru na pozici snímače zrychlení pro prodloužení operační doby modelu.

Struktura práce začíná teorií autonomních robotů a vozidel. Následuje seznámení se s modelem a jeho změnami. Návod pro nového uživatele, který by s modelem chtěl pracovat. Zde budou vysvětleny všechny otázky a potřeby uživatele. Vysvětlení algoritmizace pro dané problémy. Samotné testování modelu. Závěrem bude shrnutí projektu a jeho vyhodnocení.



Obrázek 1.1: Původní model bakalářského projektu [2]

2 Autonomní vozidla

Autonomní robot je robot, který se chová nebo provádí úkoly samovolně, což je velmi žádoucí ve vesmírném letectví, údržbě domácností (např. úklid), čištění odpadních vod, doručování zboží a služeb, válečných konfliktech, záchranářských pracích, inspekční činnosti (radiace, vysoké teploty) atd.

Jedním z důležitých odvětví výzkumu robotiky je umožnit robotovi překonat prostředí, ať už je to země, voda, vzduch, podzemí či vesmír.

Plně autonomní robot může:

- získávat informace o svém okolí,
- pracovat na určitém časovém intervalu bez lidského zásahu,
- pohybovat se celý nebo částí sebe v operačním prostoru bez lidské pomoci,
- vyhnout se situacím, které poškozují lidi, majetek nebo sami sebe pokud nejsou navrženi pro tyto situace.

Autonomní robot může také získat nebo se naučit nové znalosti například přizpůsobením nových prostředků pro plnění svých úkolů nebo aklimatizováním se měnícímu prostředí. Stejně jako ostatní stroje, autonomní roboti stále vyžadují pravidelnou údržbu. [3]

Autonomní vozidlo musí plánovat své akce, vnímat okolí, provádět jeho plány a přizpůsobit se prostředí. Takovýto komplexní systém vyžaduje správu vícero pod-systémů, spolupracujících tak, aby dosáhly běžných cílů.

Začínáje předpokladem, že nejzásadnější problém, kterému může mobilní robot čelit, je nerozhodnost. Můžeme z toho vyvodit, že robot musí mít následující základní schopnosti:

- získání informace o svém okolí pomocí senzorů,
- plánování své činnosti dle přijatých dat ze senzorů.

2.1 Senzory

Robot musí být schopen určit jeho vztah vůči okolí pomocí snímačů. Dostupný je široký výběr způsobů snímání: odometrie; ultrazvukové, infračervené a laserové snímání vzdálenosti; monokulární, binokulární a trinokulární vidění apod. Obtížnost je v překladač dat, jaké informace vlastně získáváme. Jak je máme zpracovat a jak

je máme zakomponovat do kódu, aby plnily svou úlohu, a robot se poté pohyboval nebo konal činnost podle našich plánů. To je rozhodování se podle toho, co nám senzory řeknou o vnějším světě.

Jak uvádí Štěpán [5] základní rozdělení senzorů spočívá na:

- *Aktivní*, které k provedení měření potřebují vyslat energii a detekovat její návrat (optický senzor CNY70, radar, sonar, IR).
- *Pasivní* opírající se o vhodnou fyzikální vlastnost prostředí (snímač zrychlení, kamera, kompas).
- *Distribuované*, do kterých spadá GPS.

V [8] je uváděno, že hlavní problém ve vnímání je ve vytvoření spojení mezi výstupním signálem a vlastnostmi třírozměrného světa. Pro interní senzory, jako jsou společné snímače polohy, toto spojení je pevné a známé. Pro externí senzory je spojení v nejlepším slabé.

V podstatě všechny přístupy k interpretování sensorových informací postupují prvně identifikováním sady rysů v datech, které se potenciálně shodují se subjekty ve světě.

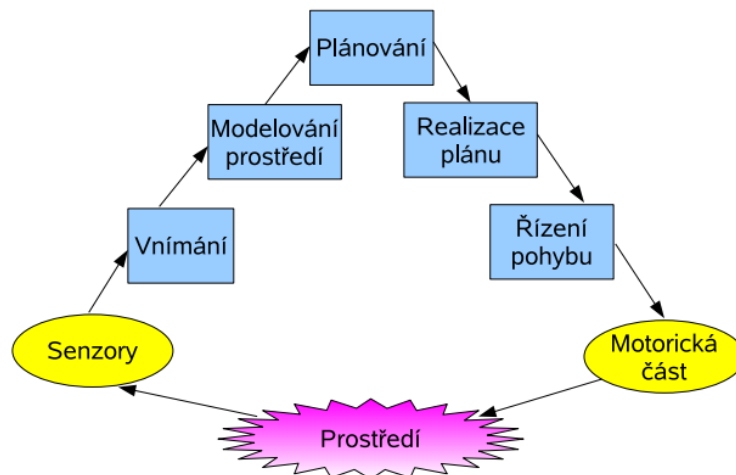
2.2 Plánování

Robot musí být schopen se rozhodnout, jaké akce jsou zapotřebí k dosažení cíle v daném prostředí. Toto může ovlivnit rozhodnutí jakou cestou se vydat a jaké senzory využít.

Uvažování robota se může lišit dle prostředí, využitých snímačů a pro jaký účel byl stvořen. Zde však nastává mnoho proměnných a mnoho způsobů jakými se robot má řídit. V tento moment přichází na řadu algoritmy pro ovládání modelu. Vozidlo se poté pomocí dat získaných ze senzorů snaží dojít ke svému cíli.

Podle Petra Štěpána [4] je několik druhů architektur mobilních robotů:

- *Reaktivní architektura*: Každý modul plánuje akci na základě sensorické informace. Výhodou reaktivních systémů je, že cíle nemusí být explicitně vyjádřeny. Způsob určení výsledné akce záleží na druhu reaktivního systému.
- *Funkční dekompozice*: Do této skupiny spadá náš robot. Funkční dekompozice je klasická metoda analýzy shora dolů. Řízení je rozděleno na množinu akcí a událostí, které na sebe navazují. Akce jsou:
 - Vnímání ovládání senzorů, čtení dat.
 - Modelování prostředí na základě dat přijatých ze senzorů.
 - Plánování akcí robotu, aby byl splněn cíl zadaný operátorem.
 - Realizace plánu a ověření, zda je plán realizovatelný.
 - Řízení pohybu robotu na nízké úrovni po naplánované trajektorii.



Obrázek 2.1: Funkční dekompozice [4]

- *Plánování na vyšší úrovni*: Metody řízení a plánování akcí robotu založené na deklarativní reprezentaci znalostí o světě.
- *Neuronové sítě a genetické algoritmy*: Pro řešení úloh, které je velmi složité přesně formulovat, nebo pokud se jedná o úlohy patřící do skupiny NP-úplných problémů, se často používají metody z oblasti soft-computingu.
- *Hybridní přístup*: Hybridní přístup je kombinací výše uvedených přístupů. V praxi se téměř vždy jedná o kombinaci více přístupů, např. kombinace reaktivního systému se systémem funkční dekompozice.

2.3 Roboti s podobnou konstrukcí

Mezi roboty s podobnou problematikou pohybu po vertikálních plochách můžeme zařadit servisního robota ROBOTUL Vertical Climber 02, který je určen pro mytí, čištění stěn a pro realizaci inspekčních činností, jako je kontrola stavu pláště v místě konstrukčního upevnění skla a kontrola neporušenosti pláště tlakových nádob z nerez oceli. Může být aplikován jako mobilní plošina pro nesení aparatury k zajištění technologických funkcí, např. odprašování povrchů, mytí apod. Jeho kompaktní rozměry $1120 \times 1120 \times 300$ mm a váha 48 kg mu umožňují nést závaží o váze až 20 kg. Umyje až 80 m^2 plochy za hodinu případně provede diagnostiku plochy 90 m^2 / hod [11].

Vertikální pohyb, pohyb po stropě.

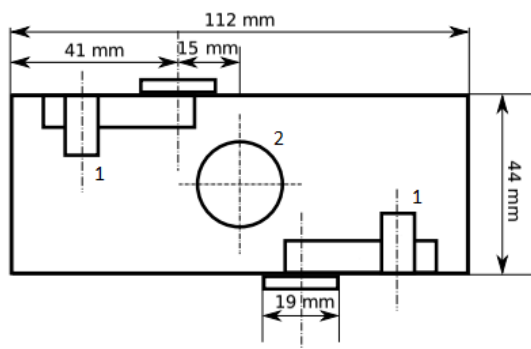


Obrázek 2.2: ROBOTUL Vertical Climber 02 [11]

3 Model vozu

Po obdržení modelu auta jsem se seznámil se všemi jeho prvky. Detailní pročtení práce pana Najmana [2] bylo velkým přínosem.

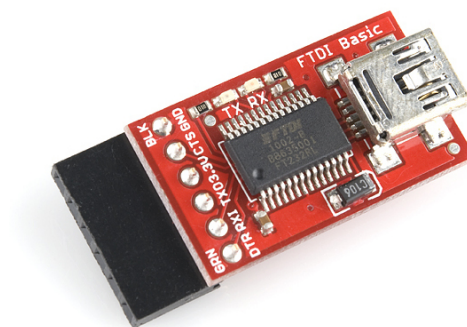
Hlavní částí modelu je podvozek z vozidla na původní dálkové ovládání. V podvozku jsou zabudované tři stejnosměrné motory. Jeden umístěn uprostřed, který obstarává podtlak pod vozidlem. Další dva jsou pro pohyb vozidla. Výkon motorů je přenesen skrz převodové ústrojí na kola. Vůz je tvořen pouze dvěma koly pro ovládání, které nejsou v jedné ose viz obrázek 3.1. Kola jsou pogumované, aby se zabránilo skluzu vozidla po vertikálních plochách. Každý z motorů je ovládán samostatně. Na podvozku je přidělaná základní konstrukce. Na konstrukci je přichycena řídicí deska Arduino Fio a dva akumulátory. Jeden akumulátor napájí desku a druhý dva motory a přísavný ventilátor. V přední části vozidla je umístěný optický senzor CNY70, který rozpoznává černou a bílou barvu na vzdálenost pár jednotek centimetrů.



Obrázek 3.1: Rozložení podvozku – 1 stejnosměrné motory, 2 ventilátor [2]

Pro řídicí desku Arduino Fio jsem na stránkách výrobce [1] získal Arduino IDE, pomocí kterého se do desky přes port USB nahrává algoritmus, podle kterého stroj pracuje. Poté jsem navrhl prostý program pro ovládání vozidla. Účelem bylo zjištění, jak co pracuje. Kód sestával pouze z jedné podmínky a to když čidlo detekuje černou barvu, tak se auto zastaví.

Dle návodu na internetových stránkách Arduina [7] byl model připojen k počítači. Pro úspěšné propojení bylo zapotřebí využít FTDI adaptér, který navazoval komunikaci, a správně nastavit všechno nastavení v Arduino IDE. Poté se kód zkompiloval a nahrál. Model se dal do pohybu.



Obrázek 3.2: FTDI deska pro propojení přes USB s počítačem [13]

3.1 Řídicí deska Arduino Fio

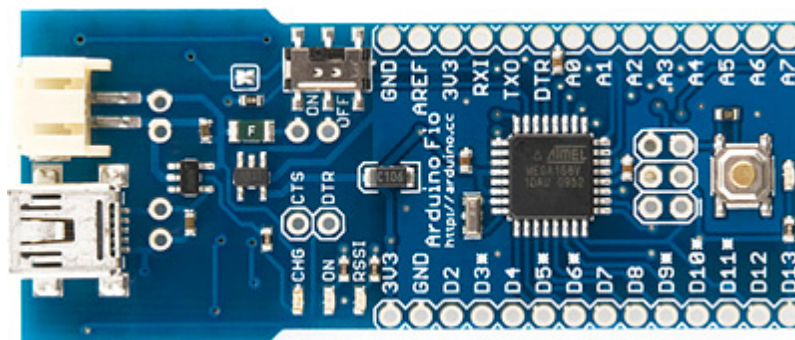
Vůz je vybaven řídicí deskou Arduino Fio. Jejím základem je 8bitový mikrokontrolér založený na architektuře RISC ATmega328P od firmy Atmel. Tato řídicí jednotka, s rozměry $27,9 \times 66 \text{ mm}$ a hmotností 9 g, má parametry:

- 14 digitálních výstupů (z toho 6 umožňuje PWM),
- 8 analogových pinů,
- frekvence 8 MHz,
- paměť typu flash o velikosti 32 KB,
- operační napětí 3,3 V,
- vstupní napětí 3,35–12V,
- vstupní napětí pro nabíjení 3,7–7V.

Arduino Fio může být napájeno dvěma způsoby, první z nich je pomocí rozhraní USB, druhá varianta je napájení z baterie o napětí 3,7 V. Je-li k Arduino připojena baterie, tak zmíněné USB funguje jako port pro její nabíjení. K nahrání programu do Arduino slouží rozšiřující plošný spoj, FTDI Basic Breakout adaptor, obsahující port USB (viz obrázek 3.2). Vždy, když se nahraje nový algoritmus, tento plošný spoj vyvolá přerušení a automaticky resetuje řídicí jednotku, nový program pak běží okamžitě bez zásahu uživatele [2].

3.2 Akumulátory

Model je vybaven výše zmíněnými dvěma akumulátory. Jsou to 3,7 V Li-Pol akumulátory s kapacitami 220 mAh a 350 mAh. Akumulátor s vyšší kapacitou je připojen k řídicí desce, druhý pohání motorické ústrojí s ventilátorem. V bakalářské práci [2]



Obrázek 3.3: Řídicí deska Arduino Fio [15]

je uvedeno, že napájení je rozděleno z důvodu prudkého kolísání napětí, které mělo za následek resetování řídicí desky. Proto byly zavedeny dva akumulátory.

Při testování vozidla se zapojenými akumulátory nastal problém. Když byl do řídicí desky nahrán program a byl odpojen USB kabel z FTDI adaptéru, tak po následném zapnutí desky přes akumulátory se vůz nepohyboval. Došel jsem k závěru, že jsou vybité akumulátory.

Každý akumulátor byl zapojen do řídicí jednotky zvlášť se zároveň připojeným USB konektorem k počítači. Deska červenou diodou s označením !!charge!! informovala, že se nabíjí akumulátor. USB konektor v počítači poskytuje napětí 5 V. Proud v USB 2.0 je rozdělen v jednotkách po 100 mA, resp. 150 mA v USB 3.0. Maximální zatížení portu je 5 jednotek, což je 500 mA, resp. 750 mA [16]. K nabití jsem využil USB 3.0 a nabíjel přibližně hodinu.

Plně nabité jsem je zapojil zpět. Zapnul spínač. Nic se nedělo. S akumulátory bylo něco v nepořádku. Do modelu jsem přivedl USB kabel přes FTDI adaptér. V tu chvíli jsem si uvědomil, že problém nastal u akumulátoru pro desku, protože motory fungují na samostatné jednotce, a proto model jezdil. Tento závěr jsem potvrdil tím, že po připojení prvního akumulátoru a zapnutí, deska nepracovala, kdežto s druhým se rozsvítily diody.

Starý akumulátor byl odstraněn a byl pořízen nový stejného typu s kapacitou 450 mAh viz obrázek 3.4. Nový akumulátor měl jiný konektor pro připojení, takže byl přepájen ze starého, aby se dal připojit do modelu. Nyní byl již vůz samostatný bez potřebného externího připojení.

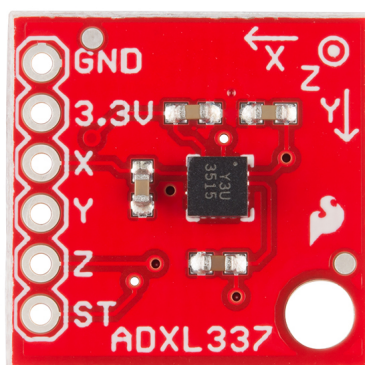
3.3 Akcelerometr ADXL337

V bakalářské práci [2] Najman zmiňuje možnost rozšíření modelu o akcelerometr.

Akcelerometr je elektromechanické zařízení, které měří zrychlení sil. Tyto síly mohou být statické jako tíhová síla, nebo dynamické – způsobeny pohybem, nárazem nebo vibrováním akcelerometru [6]. Pomocí tohoto zařízení můžeme určit v jakém úhlu je vozidlo k zemskému povrchu, a tím korigovat výkon ventilátoru a dosáhnout delší provozní doby.

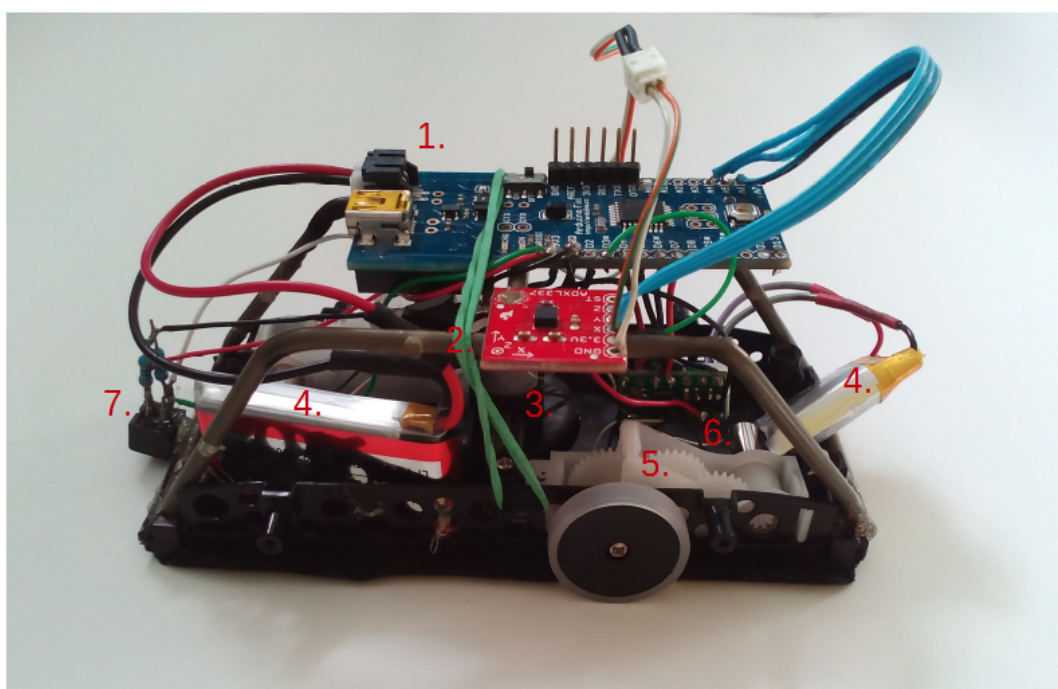


Obrázek 3.4: Nově pořízený Li-Pol akumulátor 3,7 V 450mAh [17]



Obrázek 3.5: Akcelerometr ADXL337 [18]

Do vozidla byl přidán nový modul – ADXL337 piezoelektrický snímač zrychlení [18]. Modul je schopný měřit ve třech osách. Každá osa má vlastní analogový výstupní signál, který je přiveden do desky Arduina na porty A4 až A6. Akcelerometr je přidělán tak, když je vozidlo v klidu na vodorovné ploše, aby byl na minimální hodnotě osy Z, resp. maximální jelikož když působí tíhová síla směrem dolů, tak je piezoelektrický materiál stlačen a dává vyšší hodnoty, než když je vzhůru nohama [19]. Pro naše využití je nejdůležitější osa Z. Osa Z nám určí pod jakým náklonem auto setrvává k zemskému povrchu. Akcelerometr využijeme k tomu, aby auto při jízdě po rovině nevyužívalo ventilátor a šetřila se energie v akumulátoru. Když při naklánění roviny úhel zkosení dosáhne určitého stupně, tak se spustí přísavný motor a vždy dodá dostatečný podtlak, aby se vozidlo udrželo při manipulaci s deskou. Tímto dosáhneme delší provozní doby modelu.



Obrázek 3.6: Finální stav vozidla – 1. Řídicí deska Arduino Fio, 2. Akcelerometr ADXL337, 3. Ventilátor, 4. Akumulátory (vlevo 450 mAh, vpravo 220 mAh), 5. Převodové ústrojí, 6. Stejnoseměrný motor, 7. Senzor CNY70

4 Návod pro použití modelu

V této kapitole je uvedeno, jakým způsobem máme postupovat, abychom úspěšně propojili model vozu s počítačem a nahráli program do paměti desky a následně, aby auto jelo podle nahraného algoritmu.

1. Na piny umístěné na desce se připojí rozhraní FTDI tak, aby byl GND zastrčen do správného vstupu FTDI.
2. FTDI se připojí k počítači pomocí USB kabelu, kde na jednom konci je standardní koncovka a na druhém mini USB.
3. Spustí se Arduino IDE. V záložce *Nástroje* se zvolí vývojová deska *Arduino Fio*, programátor *Atmel STK500 development board* a jako port *COM3* (může se lišit).
4. Nyní by měli být počítač a model spárovaný. Vybereme si příslušný kód, který chceme nahrát do desky a klikne se na šipku *Nahrát*. Program zkontroluje kód a nahraje ho do desky.
5. Odpojí se USB kabel a FTDI rozhraní. Nastaví se spínač na desce do polohy ON a model začne uskutečňovat zadanou práci.
6. Když model nebude reagovat je pravděpodobné, že jsou vybité akumulátory. USB kabel se připojí přímo do desky Arduina. Tento port je pouze pro nabití akumulátorů. Poté, co se nabije první akumulátor, tak se odpojí a připojí druhý, který je umístěn v zadní části vozu. Poté se model uvede do stavu, ve kterém byl před nabitím.

5 Algoritmy pro pohyb

5.1 Jízda podle čáry

Algoritmus si uloží do proměnné `sensorValue` datového typu `integer` hodnotu z pinu čidla A0 pomocí funkce pro čtení analogových vstupů `analogRead(pin)`, která je volaná na začátku cyklu funkce `loop()`. Poté následuje podmínka, ve které se využívá proměnné `start` typu `boolean`. Po spuštění algoritmu je této proměnné přiřazena hodnota `false`, tudíž je první podmínka po zapnutí modelu vždy pravdivá. V této podmínce se uskutečňuje pohyb vpřed. Nachází se zde i další podmínka, kde když čidlo detekovalo černou barvu, tak se hodnota proměnné `start` změní na `true` a tímto končí první dvě podmínky. Tím že se již nemůže vyvolat podmínka, kde hodnota v proměnné `start` se rovnala `false`, tak algoritmus přechází do druhé části.

V této části se již uskutečňuje následování černé čáry. Disponujeme dalšími dvěma proměnnými typu `boolean` s počátečními hodnotami `false` a to `temp` a `once`. `temp` slouží ke změně směru jízdy a `once` k vyvolání podmínky pro změnu směru. Při detekci černé se vozidlo pohybuje nejdříve na pravou stranu. Při kolizi s bílou barvou (hranou) se vozidlo začne pohybovat na opačnou stranu a povolí se podmínka pro změnu směru jízdy na černé barvě. Jakmile se dostane opět na černou plochu, aktivuje se změna v proměnné `temp`, aby se vozidlo přes černou barvu pohybovalo ve stejném směru jako na předchozím bílém povrchu.

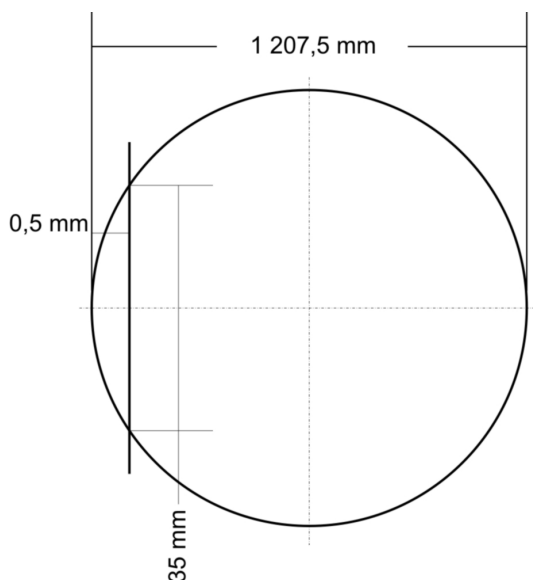
Je nutné, aby se vozidlo i na černé ploše stále stáčelo. Kdyby se pohybovalo pouze rovně, tak nastává problém, když je vyznačená trasa zakřivená. Když po bílé barvě detekuje v zatáčce černou, tak by se rozjelo kupředu opět na stejnou bílou plochu, na které bylo před momentem a změnil by se směr jízdy a vozidlo by se začalo vracet zpět, místo toho aby pokračovalo kupředu. Zde se může i zacyklit a nepokračovat v jízdě podle čáry.

5.2 Jízda ve válci

Cílem jízdy ve (na) válci bylo aby ho model uvnitř (vně) projel po celé kružnici s regulovaným přísavným ventilátorem podle akcelerometru. Testování nejdříve probíhalo na zakřiveném kartónu. Prvotní testy jízdy ve válci byly neúspěšné, vozidlo nebylo koly v kontaktu s povrchem.

Při měření proporcí modelu bylo zjištěno, že osy kol jsou od sebe vzdáleny 35 mm a světlá výška podvozku je stěží 1 mm. S těmito daty jsem musel sestavit takovou kružnici, aby její sečna měla na kolmici se středem od sečny ke kružnici

vzdálenost 0,5 mm a přímka mezi body protnutí byla dlouhá 35 mm. K zpracování tohoto problému jsem použil grafický program, kde jsem sestrojil přímku dlouhou 35 mm. Na střed této přímky jsem umístil kolmici. Na kolmici se měl vyskytovat střed kružnice. Sestrojil prvotní kružnici, kterou jsem postupně zvětšoval, dokud se krajní body přímky nedotýkaly kružnice. Poté jsem změřil průměr kružnice. Měřil 1 207,5 mm. To znamená, že bychom museli mít válec s průměrem 1,21 m, aby se mohl model vně pohybovat. Válec s takovým průměrem jsme nebyli schopni vyrobit a testování bylo ukončeno. Kdyby jsme chtěli pohyb uvnitř válce, tak by to byl již nemyslitelný průměr jelikož délka podvozku činí 142 mm, což je víc jak 4krát větší.



Obrázek 5.1: Kružnice pro pohyb vně válce

```

0  //log vstup pro pin MODE
   #define modePin 2

   //log vstup pro motor 1
   #define motorPin1 4
5
   //log vstup pro motor 2
   #define motorPin2 7

   //piny pro rizeni motoru
10  #define powerPin1 5
   #define powerPin2 6

   //pin pro senzor
   #define sensor A0
15

   //pin pro meric zrychleni
   #define axisX A4
   #define axisY A5
   #define axisZ A6
20

   //promenne
   int sensorValue = 0;
   boolean start = false;
   boolean temp = false;
25  boolean once = false;

   void setup() {
       pinMode(sensor, INPUT);

30     pinMode(axisX, INPUT);
       pinMode(axisY, INPUT);
       pinMode(axisZ, INPUT);

       pinMode(modePin, OUTPUT);
35
       pinMode(motorPin1, OUTPUT);
       pinMode(motorPin2, OUTPUT);

       pinMode(powerPin1, OUTPUT);
40     pinMode(powerPin2, OUTPUT);

       digitalWrite(modePin, HIGH);
   }

```

Zdrojový kód 5.1: Jízda podle čáry

```

45 void loop(){
    //nacteni hodnoty ze senzoru
    sensorValue = analogRead(sensor);

    //podminka pro pohyb vpred nez se narazi na cernou barvu
50 if (!start){
    forward(50);
    if (sensorValue < 300){
        start = !start;
    }
55 }
    else {
        if (sensorValue < 300){
            if (!temp){
                if (once){
50                 temp = !temp;
                    once = false;
                }
                else{
55                 rightTurn(50);
                }
            }
            if (temp){
                if (once){
50                 temp = !temp;
                    once = false;
                }
                else {
55                 leftTurn(50);
                }
            }
75        }
    }
    else{
        if (temp){
            once = true;
80            rightTurn(50);
        }
        if (!temp){
            once = true;
85            leftTurn(50);
        }
    }
}
}

```

Zdrojový kód 5.2: Jízda podle čáry

```

90
    void forward(int power){
        digitalWrite(motorPin1, HIGH);
        digitalWrite(motorPin2, LOW);
        analogWrite(powerPin1, power);
95    analogWrite(powerPin2, power-5);
    }

    void leftTurn(int power){
        digitalWrite(motorPin1, HIGH);
100    analogWrite(powerPin1, power);
        analogWrite(powerPin2, 0);
    }

    void rightTurn(int power){
105    digitalWrite(motorPin2, LOW);
        analogWrite(powerPin2, power);
        analogWrite(powerPin1, 0);
    }

```

Zdrojový kód 5.3: Jízda podle čáry

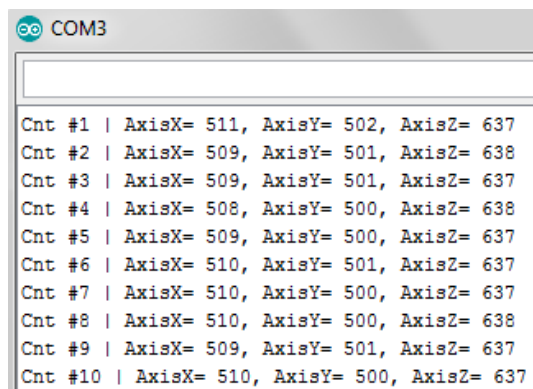
6 Testování modelu

6.1 Testování zapojení akcelerometru ADXL337

Poté, co byl přidán do modelu akcelerometr, bylo zapotřebí ho inicializovat. Stejně jako pro senzor CNY70 v bakalářské práci [2] byly definované nové vstupní proměnné.

A4, A5 a A6 značí analogové vstupy do Arduina. „axisX“ je pojmenování proměnné. K získání hodnot z akcelerometru využijeme funkci **analogRead(pin)**. Model se připojí přes rozhraní FTDI. Nahraje se do něj kód [0lst:adx1]. A model se spustí s připojeným kabelem. Na počítači se v Arduino IDE otevře sériový monitor a na něm se začne objevovat textový výstup viz obrázek 6.1.

Z měření na pozicích podle obrázku 6.2 jsme získali tabulku 6.1. Náš zájem se směřuje k ose Z. V klidovém stavu v první pozici nabývá akcelerometr hodnotu 637, na druhé 530 a na třetí 426. Z těchto dat můžeme usoudit, že jeden stupeň úhlu se rovná jedné v akcelerometru.



	AxisX	AxisY	AxisZ
Cnt #1	511	502	637
Cnt #2	509	501	638
Cnt #3	509	501	637
Cnt #4	508	500	638
Cnt #5	509	500	637
Cnt #6	510	501	637
Cnt #7	510	500	637
Cnt #8	510	500	638
Cnt #9	509	501	637
Cnt #10	510	500	637

Obrázek 6.1: Sériový monitor s klidovým výstupem

6.2 Testování přísavného ventilátoru

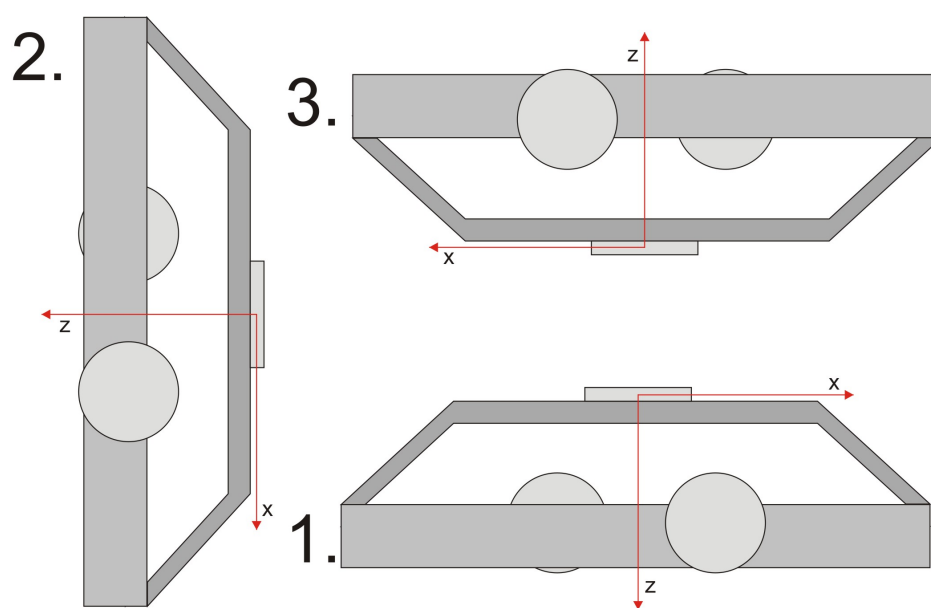
Cílem testování bylo, aby přísavný ventilátor udržel vozidlo v jakémkoli úhlu s tím, aby jeho výkon byl závislý na hodnotách akcelerometru.

Prvním krokem bylo zjištění v jaké pozici se model stále udrží na nakloněné novině bez použití ventilátoru. Tím byla krajní hodnota 625 osy Z. Dokud akcele-

Tabulka 6.1: Tabulka naměřených hodnot

	1			2			3		
Číslo měření	x	y	z	x	y	z	x	y	z
1	511	502	637	612	511	530	505	495	426
2	509	501	638	610	510	531	505	496	425
3	509	501	637	611	511	530	504	498	426
4	508	500	638	610	511	531	505	497	425
5	509	500	637	611	511	531	504	498	426
6	510	501	637	611	511	530	503	498	426
7	510	500	637	611	510	530	504	499	428
8	510	500	638	611	510	531	505	497	429
9	509	501	637	610	512	531	503	500	428
10	510	500	637	612	511	530	504	499	428

rometr snímal hodnoty vyšší jak 625, tak se ventilátor nespustil. Jako druhá krajní hodnota pro maximální výkon ventilátoru je 540, když jsou snímány hodnoty nižší tak to nijak neovlivňuje výkon ventilátoru. K získání dostatečného přtlaku mezi hodnotami 625 a 540 mi posloužil násobící koeficient 3,5, kterým se násobí 670 od kterého se odečte snímaná hodnota akcelerometrem.



Obrázek 6.2: Pozice modelu pro měření hodnot akcelerometru

```

    //pin pro meric zrychleni
2  #define axisX A4
    #define axisY A5
    #define axisZ A6

    //promenne
7  int X, Y, Z;
    int cnt = 0;

    void setup() {
        pinMode(axisX, INPUT);
12     pinMode(axisY, INPUT);
        pinMode(axisZ, INPUT);

        digitalWrite(modePin, HIGH);
        Serial.begin(9600);
17 }

    void loop(){
        cnt++;
        delay(5000);
22     X = analogRead(axisX);
        Y = analogRead(axisY);
        Z = analogRead(axisZ);

        Serial.print(String("") + "Cnt #" + cnt + " | AxisX= " +
                                X + ", AxisY= " + Y + ",
                                AxisZ= " + Z + "\n");
27 }

```

Zdrojový kód 6.1: Testování akcelerometru

```

//log vstup pro pin MODE
#define modePin 2
3
//log pro rizeni ventilatoru
#define ventPin 3

//pin pro meric zrychleni
8 #define axisZ A6

//promenne
int start = 0;
int sensorValue = 0;
13
void setup() {
    pinMode(axisZ, INPUT);

    pinMode(modePin, OUTPUT);
18    pinMode(ventPin, OUTPUT);

    digitalWrite(modePin, HIGH);
    Serial.begin(9600);
23 }

void loop(){
    sensorValue = analogRead(axisZ);
    //Serial.print(String("") + sensorValue + "\n");
28    if (sensorValue < 540){
        analogWrite(ventPin, 255);
    }
    else if (sensorValue < 625){
        start = (670-sensorValue)*3.5;
33    analogWrite(ventPin, start);
    }
    else {
        analogWrite(ventPin, 0);
    }
38 }

```

Zdrojový kód 6.2: Závislost výkonu ventilátoru na akcelerometru

7 Závěr

Literatura

- [1] Arduino [online]. 2017 [cit. 2017-05-17]. Dostupné z: <https://www.arduino.cc>
- [2] NAJMAN, Petr. Automaticky řízený model vozu s přísavným systémem [online]. Liberec, 2015. Bakalářská práce. Technická univerzita v Liberci. Vedoucí práce Jan Koprnický.
- [3] Autonomous robot. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2017 [cit. 2017-05-17]. Dostupné z: https://en.wikipedia.org/w/index.php?title=Autonomous_robot&oldid=780231059
- [4] ŠTĚPÁN, Petr. Mobilní robotika: Úvod do mobilní robotiky [online]. Praha: České vysoké učení v Praze, 2010 [cit. 2017-04-30]. Dostupné z: https://cw.fel.cvut.cz/wiki/_media/mkr/lessons/lesson1.pdf
- [5] ŠTĚPÁN, Petr. Mobilní robotika: Senzory a Plánování [online]. Praha: České vysoké učení v Praze, 2011 [cit. 2017-04-30]. Dostupné z: https://cw.fel.cvut.cz/wiki/_media/courses/a3m33mkr/c-space.pdf
- [6] Small, Low Power, 3-Axis ± 3 g Accelerometer [online]. Rev. 0. Nordwood: Analog Devices, 2010 [cit. 2017-05-10]. Dostupné z: <http://www.analog.com/media/en/technical-documentation/data-sheets/ADXL337.pdf>
- [7] Getting Started with the Arduino Fio // Retired. Arduino [online]. [cit. 2017-05-03]. Dostupné z: <https://www.arduino.cc/en/Main/ArduinoBoardFioProgramming>
- [8] COX, Ingeman J., Gordon T. WILFONG a T. LOZANO-PEREZ. Autonomous Robot Vehicles. Ilustrované vydání. New York: Springer New York, 2012. ISBN 1461389984, 9781461389989.
- [9] SATRAPA, Pavel. LaTeX pro pragmatiky [online]. TUL a CESNET, 2011 [cit. 2017-05-17]. Dostupné z: <http://nti.tul.cz/satrapa/docs/latex/>
- [10] SATRAPA, Pavel. Balik tul pro LaTeX [online]. Liberec, 2015 [cit. 2017-05-17]. Dostupné z: <http://nti.tul.cz/satrapa/vyuka/latex-tul/>

- [11] Servisní robot pohybující se po hladké vertikální ploše. In: B&R [online]. Austria, 2014 [cit. 2017-05-17]. Dostupné z: <https://www.br-automation.com/cs/spolecnost/tiskove-zpravy/servisni-robot-pohybujici-se-po-hladke-vertikalni-plose/>
- [12] Matematická sazba. Katedra informatiky [online]. Ostrava [cit. 2017-05-17]. Dostupné z: <http://www.cs.vsb.cz/benes/vyuka/latex/math.htm>
- [13] FTDI Basic Breakout - 3.3V. In: SparkFun [online]. Niwot, Colorado [cit. 2017-05-17]. Dostupné z: <https://www.sparkfun.com/products/retired/8772>
- [14] CARLISLE, David, Scott PAKIN a Alexander HOLT. The Great, Big List of LATEX Symbols [online]. 2001 [cit. 2017-05-18]. Dostupné z: https://www.rpi.edu/dept/arc/training/latex/LaTeX_symbols.pdf
- [15] Arduino Fio. In: Arduino [online]. [cit. 2017-05-18]. Dostupné z: <https://www.arduino.cc/en/Main/ArduinoBoardFio>
- [16] REICHARD, Doktoro. What is the power output of a USB port? In: Superuser [online]. [cit. 2017-05-19]. Dostupné z: <https://superuser.com/questions/690074/what-is-the-power-output-of-a-usb-port>
- [17] UDI U28-1 Kestrel: Baterie LiPo 450mAh 3,7V (pro vysílač). In: PECKA - MODELÁŘ [online]. [cit. 2017-05-19]. Dostupné z: <http://www.peckamodel.cz/produkt/nahradni-dily-a-prislusenstvi/multikoptery/udi-u28-1-kestrel/u28-1-16-kvadrocoptera-u28-1-baterie-lipo-450mah-3-7v-do-vysilace>
- [18] SparkFun Triple Axis Accelerometer Breakout - ADXL337. In: SparkFun [online]. Niwot, Colorado [cit. 2017-05-19]. Dostupné z: <https://www.sparkfun.com/products/12786>
- [19] VOJÁČEK, Antonín. Principy akcelerometrů - 1. díl - Piezoelektrické. In: Automatizace.HW.cz [online]. 2007 [cit. 2017-05-19]. Dostupné z: <http://automatizace.hw.cz/clanek/2007011401>