VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ZAŘÍZENÍ S MODELÁŘSKÝMI RC SERVY

SYSTEM WITH RC MODEL SERVOS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **ANDREJ BARNA**
AUTHOR

VEDOUCÍ PRÁCE **Prof. Dr. Ing. PAVEL ZEMČÍK**
SUPERVISOR

BRNO 2017

Zadání bakalářské práce/19799/2016/xbarna01

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2016/2017

Zadání bakalářské práce

Řešitel: **Barna Andrej**
Obor: Informační technologie
Téma: **Zařízení s modelářskými RC servy**
System with RC Model Servos
Kategorie: Uživatelská rozhraní

Pokyny:

1. Prostudujte způsob řízení modelářských serv a možnosti jejich ovládání počítačem.
2. navrhněte jednoduchý strojek sestrojený z modelářských serv a způsob jeho ovládání (například jezdící model automobilu)
3. Popište a diskutujte možnosti ovládání modelářskými servy.
4. Navržený systém implementujte a demonstrujte.
5. Diskutujte dosažené výsledky a možnosti pokračování práce.

Literatura:

- Dle pokynů vedoucího

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3 zadání

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Zemčík Pavel, prof. Dr. Ing.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2016
Datum odevzdání: 17. května 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

Účelom tejto práce je skúmanie modelárskych servomotorov a možností ich ovládania prostredníctvom počítača. S využitím získaných znalostí je pre demonštráciu následne navrhnuté a skonštruované jednoduché vozidlo riadené servomotormi, ktoré sú ovládané doskou Arduino. Taktiež je vytvorená aplikácia na počítač, ktorá je schopná toto vozidlo ovládať prostredníctvom rozhrania Bluetooth.

Abstract

The goal of this thesis is the studying of RC servos and possibilities of their control using a computer. Using the collected knowledge, a simple vehicle driven by servos and controlled by the Arduino board is designed and constructed for a demonstration. Also, a computer application is created, which is capable of controlling the constructed vehicle via the Bluetooth interface.

Kľúčové slová

servo, model, RC vozidlo, Arduino, Bluetooth

Keywords

servo, model, RC vehicle, Arduino, Bluetooth

Citácia

BARNA, Andrej. *Zařízení s modelářskými RC servy.* Brno, 2017. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Zemčík Pavel.

Zařízení s modelářskými RC servy

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána Prof. Dr. Ing. Pavla Zemčíka. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

.....
Andrej Barna
16. mája 2017

Poďakovanie

Chcel by som poďakovať vedúcemu práce, Prof. Dr. Ing. Pavlovi Zemčíkovi, za poskytnutie slobody a priestoru pre sebarealizáciu pri vypracovávaní práce a za pomoc pri jej korektúre.

Obsah

Kapitola 1

Úvod

Diaľkovo ovládané modely sa stávajú stále populárnejšími, pričom to je z dosť veľkej časti aj vďaka stálemu klesaniu cien elektronických súčiastok. Problémom bežne zakúpených modelov je však to, že sú často modifikovateľné buď len vo veľmi obmedzenej miere, alebo vôbec. Ak je teda záujem toto vozidlo upraviť podľa vlastných potrieb, tak je to obtiažne. Tento problém sa dá obísť konštruovaním vlastného modelu, avšak to vyžaduje ako určitú technickú zdatnosť, tak často aj schopnosť tvorby programu, ktorý by umožňoval ovládanie tohto vozidla. V tejto práci vytvorím takýto model vozidla a s ním aj program, ktorý by mal byť použiteľný aj pre iné projekty podobného charakteru a tým by mal uľahčiť tvorbu diaľkovo ovládaných vozidiel.

Obsahom tejto práce je teda tvorba zariadenia, ktoré bude možné ovládať počítačom a ktoré bude poháňané prostredníctvom modelárskych servomotorov. Ako zariadenie, ktoré bude spomenutými servomotormi ovládané, som si zvolil model vozidla. Rozhodol som sa, že vytvorená aplikácia pre riadenie tohto zariadenia bude umožňovať, aby bolo jednoducho možné meniť parametre pripojených servomotorov a aby bola použíteľná pre viacero druhov konštrukcií s modelárskych servomotormi aj bez zásahu do zdrojového kódu programu. Vďaka tomu by teda vytvorená aplikácia mala byť znovupoužiteľná pre rozličné projekty.

V ďalšej kapitole zhrniem teóriu ako serv, ich konštrukcie a parametrov, tak aj modulácií a využitia pulznej šírkovej modulácie pre použitie serv. Taktiež budú poskytnuté informácie o minipočítačoch používaných ako riadiace platformy pre servá, a o možnostiach prenosu dát z počítača na riadiacu platformu. V tretej kapitole bude uvedená analýza požiadaviek na vytvorené vozidlo ovládané servami a tiež požiadavky, ktoré sú kladené na program pre riadiacu platformu a na aplikáciu pre osobný počítač. Štvrtá kapitola bude obsahovať informácie o voľbe súčiastok na konštrukciu vozidla a stručný popis konštrukcie. Ďalej bude obsahovať popis vytvorenej aplikácie pre počítač a informácie o implementácii programu pre riadiacu platformu. Piata kapitola obsahuje krátke zhrnutie výsledkov skúšania ako mnou vytvoreného vozidla, tak aj aplikácie.

Kapitola 2

Zhrnutie súčasného stavu serv a možností ich ovládania počítačom

Servomotory (skrátene servá) sú mechanizmy, ktorých účelom je vykonávanie mechanickej činnosti v závislosti od riadiaceho (elektrického) signálu. Jedná sa o servomechanizmy – automatické zariadenia schopné na základe zápornej spätnej väzby korigovať svoju činnosť.

Servá majú širokú škálu využitia práve v modelárstve, kde sa dajú použiť u rôznych typov modelov. Napríklad u pohyblivých modelov na diaľkové ovládanie (automobily, lietadlá, lode) sa používajú pre riadenie alebo na modelárskych vlakových tratiach sa používajú na ovládanie výhybiek. Servá majú aj svoje využitie v robotike, kde sa používajú napríklad pre pohyb kĺbov robota. [9, 11, 15]

Každopádne, servá potrebujú pre funkčnosť riadiace signály, bez ktorých sú nepoužiteľné. Z toho dôvodu sa servá pripájajú k riadiacim platformám, ktoré sú im schopné tieto riadiace signály generovať. Keďže však je interakcia používateľa žiadaná nie priamo s riadiacou platformou, ale s počítačom, tak je tiež vyžadované aby bolo možné prenášať informácie o vstupoch používateľa z počítača na riadiacu platformu.

2.1 Konštrukcia serv

Tradičné servomotory sa skladajú z niekoľkých základných prvkov, ktorými sú jednosmerný motor, prevodovka a riadiaca elektronika, ktoré sú uložené v puzdre. Pod riadiacou elektronikou sa rozumie monostabilný klopný obvod, sčítačka, môstkový spínač a potenciometer pripojený na prevodovku, ktorý slúži ako spätná väzba. Taktiež je z puzdra vyvedený trojžilový kábel, kde je na čierny vodič pripojená zem, na červenom káble je zdroj napájania a žltý kábel slúži pre prenos riadiacich signálov. Farebné označenie týchto vodičov sa môže u niektorých výrobcov líšiť. U kvalitnejších serv môžu byť pri výstupnom hriadeli uložené v puzdre až dve guličkové ložiská pre zvýšenie životnosti serva. [9, 11, 17]

Na obrázku 2.1 je možné vidieť rozobraté servo, na obrázku je hore vľavo puzdro s nasadenými prevodmi, pod ním je na doske elektromotor s riadiacou elektronikou a pripojeným káblom. Ďalej je na obrázku možné vidieť horný a spodný kryt, upevňujúce šrúbky a páku, ktorá sa pripevňuje na výstupný hriadeľ.

Obr. 2.1: Servo rozložené na diely[1]

2.2 Princíp fungovania serv

Po riadiacom vodiči prichádzajú kladné impulzy s pravidelnou frekvenciou 50 Hz (teda každých 20 ms). Šírka týchto impulzov určuje polohu, do ktorej sa má výstupný hriadeľ serva natočiť. Najčastejšie používanou neutrálnou hodnotou (kedy má byť výstupný hriadeľ v stredovej polohe) je šírka pulzu 1,5 ms. Význam týchto signálov je neskôr podrobnejšie popísaný v kapitole o ovládaní serv.

Prijatý impulz spustí monostabilný klopný obvod. Ten následne vygeneruje impulz, ktorý zodpovedá aktuálnemu natočeniu výstupného hriadeľa. To je možné vďaka potenciometru, ktorý je pripojený na prevodovku a poskytuje spätnú väzbu o aktuálnom natočení. Impulz vygenerovaný monostabilným klopným obvodom má opačnú polaritu než mal vstupný riadiaci impulz. Následne sa sčíta tento vygenerovaný impulz so vstupným riadiacim impulzom v sčítačke a výstupom je impulz, na základe ktorého je možné jednoznačne určiť ktorým smerom sa má výstupný hriadeľ pootočiť. Výsledný impulz je následne zosilnený cez môstkový spínač a použije sa pre otáčanie výstupným hriadeľom. Ak je tento impulz nulový, tak aktuálne natočenie serva zodpovedá riadiacemu impulzu, inak v závisosti od polarity tohto impulzu sa má výstupný hriadeľ natočiť doprava, alebo doľava. V prípade, že sa zmenilo natočenie výstupného hriadeľa, tak sa tiež zmení sila impulzu z monostabilného klopného obvodu – ktorý sa opäť porovnáva s riadiacim impulzom a mení sa natočenie dokedy rozdiel riadiaceho a spätnoväzobného impulzu nie je rovný nule. [4, 7, 9, 15, 17]

[1] Zdroj: *https://upload.wikimedia.org/wikipedia/commons/e/ec/Exploded_Servo.jpg*

2.3 Parametre serv

- Veľkosť – Rozmery serva, rozdelenie je popísané v nasledujúcej sekcii.
- Hmotnosť – Je úzko spätá s veľkosťou, avšak treba započítať aj ďalšie faktory, akými sú napríklad typ motoru v serve alebo typ prevodovky.
- Ťah – Udáva, akú hmotnosť dokáže servo potiahnuť na ramene určitej dĺžky. Obvykle sa využíva ako jednotka kilogram na centimeter, teda tento parameter udáva koľko kilogramov je možné utiahnuť na ramene s dĺžkou 1 cm. To, že je servo schopné utiahnuť určitú hmotnosť neznamená, že je spôsobilé dlhodobo manipulovať s danou hmotnosťou – niektoré miniatúrne servá síce dokážu potiahnuť hmotnosť porovnateľnú s väčšími servami, avšak pri dlhodobej záťaži by sa tieto servá mohli poškodiť. Ťah bežných serv sa pohybuje v rozmedzí 0,5–10 kg/cm. [11, 15, 17]
- Typ riadiacej elektroniky – Elektronika v serve je buď riadená analógovo, alebo digitálne. Hlavný rozdiel medzi týmito typmi serv je v tom, že u digitálnych serv sa využíva mikroprocesor na rozdiel od analógových serv, kde sa používajú elektrické obvody. Digitálne servá sú v porovnaní s analógovými servami vďaka tomu presnejšie a silnejšie. Taktiež digitálne servá dokážu pracovať s vyššou frekvenciou impulzov a je možné v nich nastaviť niektoré parametre. Jedná sa o nastavenie koncového bodu a stredu, smeru otáčania, šírky pásma necitlivosti, polohy bez signálu a rýchlosti. U kvalitnejších digitálnych serv je možné nastaviť aj citlivosť a ochranu proti preťaženiu. Tieto výhody digitálnych serv sú však za cenu vyšších nákladov pri kúpe a taktiež digitálne servá majú vyššiu spotrebu. [11, 17]
- Nápajacie napätie a prúd – Tradičné napájacie napätie je 5 V, ale väčšina serv dokáže pracovať aj na 4,8 V alebo 6 V napätí. Tieto napätia sú podporované preto, lebo sú násobkami napätia NiCd alebo NiMH článkov – 1,2 V. Väčšie servá sú obvykle schopné zvládať aj vyššie napätia, ako napríklad 8,4 V. Napájací prúd závisí od veľkosti serva (respektíve jeho motoru) a typu riadiacej elektroniky. Obvyklý napájací prúd je v rozmedzí 0,1–2 A. [4, 7, 11, 15, 17]
- Operačná rýchlosť – Hodnota, ktorá udáva za aký čas sa vystaví rameno o určitý uhol, obvykle sa používa 45° alebo 60° uhol. Udáva sa v sekundách za daný uhol, napríklad 0,14 s/45°. [15]
- Operačný uhol – Udáva o aký maximálny uhol je servo schopné natočiť rameno. Bežne sa jedná o 180° uhol alebo 90° uhol.
- Kontrolné impulzy – Určujú dĺžku impulzu pre natočenie ramena do stredovej pozície (obvykle 1,5 ms) a minimálnu a maximálnu dĺžku pulzu (pre natočenie ramena do krajných prípustných pozícií) – obvykle sa používa interval impulzov buď 1–2 ms alebo 1,25–1,75 ms.
- Typ motoru – V súčastnosti sa vo väčšine lacných a štandardných serv používa tzv. trojpólový elektromotor, ktorý je jedným z najčastejšie používaných elektromotorov. V silnejších servách sa používajú aj päťpólové elektromotory, ktoré majú v porovnaní s trojpólovými rýchlejšiu akceleráciu a vyšší ťah. Tieto motory obsahujú kovové jadro omotané drôtom a obložené permanentnými magnetmi, čo má značný vplyv na hmotnosť týchto elektromotorov. Vzhľadom na túto vyššiu hmotnosť jadra motoru

je potrebné vynaložiť viac energie, aby sa roztočilo a tiež je kvôli hmotnosti jadra akcelerácia pomalšia. Taktiež po ukončení prísunu energie do motoru trvá spomalenie viac času vzhľadom na zotrvačnosť roztočeného jadra. Tento problém riešia takzvané bezjadrové (coreless) motory, ktoré majú toto ťažké kovové jadro nahradené ľahkou drôtenou konštrukciou, ktorá rotuje okolo vonkajšej strany magnetov. Vďaka zníženej hmotnosti rotujúcej časti motoru má motor rýchlejšiu akceleráciu a deceleráciu. Bezjadrové motory majú teda lepší čas odozvy, vyšší ťah a plynulejšie ovládanie. Ďalším typom motorov sú takzvané bezkomutátorové (brushless) motory, ktoré neobsahujú komutátory (mechanické rotačné usmerňovače prepínajúce smer prúdu vedeného do cievok) uložené na rotujúcej časti motoru, ale sú namiesto toho komutované elektronicky. Tieto motory sú efektívnejšie, výkonnejšie a taktiež majú dlhšiu životnosť v porovnaní s komutátorovými a bezjadrovými motormi. Sú však tiež v porovnaní s nimi o dosť drahšie a majú ťažšiu rotujúcu časť motoru než bezjadrové motory. [9, 17]

- Materiál prevodov – Ovplyvňuje odolnosť prevodovky voči opotrebovaniu. Delenie podľa materiálu prevodov je popísané v nasledujúcej sekcii.

- Uloženie prevodov – V lacnejších servách sú obvykle prevody uložené v plastovom púzdre, pričom u týchto serv sa prevody časom viac opotrebujú vplyvom trenia, než u drahších serv, ktoré môžu mať prevody uložené v guličkových ložiskách. Servá, ktoré majú prevody uložené v guličkových ložiskách sa označujú skratkou BB (Ball Bearing). Výstupný hriadeľ býva uložený v 0–2 guličkových ložiskách. [11, 17]

- Rozlíšenie – Určuje presnosť, s akou je vystavované rameno serva, keď dostane externý signál. Udáva sa v stupňoch a bežné servá majú rozlíšenie v rozmedzí 1–10°. Na tento parameter má dosť veľký vplyv riadiaca elektronika – servá s digitálnou riadiacou elektronikou dokážu presnejšie vystaviť rameno, teda s jemnejším rozlíšením. [15]

- Cena – Odvíja sa od viacerých spomenutých parametrov. Cena podľa veľkosti serva sa pohybuje v závislosti od toho, ako často sa servá danej veľkosti používajú — servá štandardnej veľkosti sú lacnejšie než veľké či nano servá. Servá s digitálnou riadiacou elektronikou bývajú značne drahšie než servá s analógovou riadiacou elektronikou, taktiež bývajú drahšie servá s vyšším ťahom, kvalitnejším motorom, kvalitnejším materiálom prevodov či s prevodovkou uloženou v guličkových ložiskách.

2.4 Delenie serv

1. Podľa veľkosti – Toto delenie je založené na hmotnosti a dĺžke serv (občas sa zarátava aj ťah serv). Štandardné pomenovanie kategórií sa môže líšiť v závislosti od výrobcov a taktiež niektorí výrobcovia spájajú niekoľko kategórií do jednej. [8, 17]

   (a) Nano – hmotnosť pod 6,3 g a dĺžka pod 19,8 mm

   (b) Sub-mikro – hmotnosť medzi 6,3 g a 12,5 g a dĺžka medzi 22 mm a 23,4 mm

   (c) Mikro – hmotnosť medzi 16,4 g a 23,5 g a dĺžka medzi 28 mm a 29,5 mm

   (d) Sub-mini – hmotnosť medzi 21,6 g a 34 g a dĺžka medzi 31 mm a 33 mm

   (e) Mini – hmotnosť medzi 25 g a 38 g a dĺžka medzi 35,5 mm a 36 mm

   (f) Štandardné – hmotnosť medzi 36,5 g a 58 g a dĺžka medzi 38,3 mm a 41 mm

(g) Veľké – hmotnosť nad 58 g a dĺžka nad 41 mm

2. Podľa typu riadiacej elektroniky:
   (a) Analógové
   (b) Digitálne
3. Podľa typu prevodov [9, 11, 17]:
   (a) Plastové – Všetky prevody v prevodovke sú z plastu. Najlacnejšia varianta a s najnižšou hmotnosťou, avšak s najnižšou spoľahlivosťou.
   (b) Kovové – Všetky prevody sú z kovu, obvykle sa táto varianta používa v kvalitnejších a silnejších servách, kde by sa plastové prevody opotrebovali rýchlejšie vzhľadom na vyšší ťah serva. Táto skupina serv sa označuje skratkou MG (Metal Gears). V rámci tejto skupiny existuje ešte podskupina označená skratkou TG (Titan Gears), ktorá má prevody vyrobené z titánu, ktoré majú nízku hmotnosť a zároveň vysokú pevnosť. Tieto servá sa používajú najmä v modeloch helikoptér a lietadiel, kde je hmotnosť významným faktorom.
   (c) Hybridné – Kombinácia kovových a plastových prevodov. Táto varianta sa označuje skratkou HG (Hybrid Gears).
4. Podľa typu pohybu ramena [7]:
   (a) Pozične rotačné – najčastejšie využívané servo, u ktorého sa riadiacim signálom nastavuje natočenie výstupného hriadeľa v rámci možných medzí.
   (b) Kontinuálne rotačné – bežne používné ako pohon u pohyblivých modelov, namiesto určovania natočenia výstupného hriadeľa sa nastavuje rýchlosť otáčania hriadeľa doprava/doľava.
   (c) Lineárne – používajú sa zriedkavo, napríklad na vysúvanie podvozku u leteckých modelov. Princíp je podobný ako u pozične rotačných serv, avšak s prevodmi navyše pre zmenu pohybu výstupnej osi z kruhovitého pohybu na lineárny.

2.5 Ovládanie serv

Ovládanie serv prebieha pomocou dátového (žltého) vodiča, ktorým sa prenáša riadiaci signál. Tento signál je obvykle modulovaný, pričom sa v súčasnosti používa na ovládanie RC serv výlučne pulzne šírkovo modulovaný (PWM) signál. [4]

2.5.1 Modulácia

Modulácia je proces ovplyvňovania nosného signálu modulačným signálom. Používa sa pri prenose elektrických signálov. Bežne sa využívajú modulované signály pre prenosy rozhlasových či televíznych signálov, dát v mobilných telefónoch a podobne.

V súvislosti s moduláciou sa používajú termíny modulátor a demodulátor. Jedná sa o prvky schopné signál modulovať, respektíve demodulovať. Demodulácia je operácia inverzná k modulácii, vykonávaná za účelom získania vstupného signálu na prijímacej strane.

Pre úplnosť uvádzam v tejto sekcii aj rozdelenie typov modulačných techník a informácie o nich, keďže niektoré z týchto techník sa používajú napríklad aj na ovládanie bežne predávaných diaľkovo ovládaných modelov vozidiel. Väčšina spomenutých modulačných techník sa však serv netýka, výnimka pulznej širkovej modulácie.

Existuje veľa druhov modulačných techník, ktoré sa delia do niekoľkých väčších skupín [10]:

1. Spojité modulačné techniky – nosný signál je spojitý, jedná sa o harmonický signál zapísateľný v tvare

$$y = A \ \cdot \sin(\Omega t + \phi) \tag{2.1}$$

   kde A je amplitúda signálu, Ω je uhlová frekvencia, t je čas a ϕ je fázovy posun. Výsledná hodnota y je hodnotou vstupného signálu s danými parametrami. V závislosti od typu modulačného signálu sa delia na dve kategórie:

   (a) Skupina analógových modulačných techník, do ktorej spadajú dve podskupiny – amplitúdová modulácia (AM) a uhlová modulácia. Do skupiny uhlovej modulácie spadá napríklad frekvenčná modulácia (FM), fázová modulácia (PM) či transpozičná modulácia (TM). Princípom týchto metód je zmena parametrov nosného harmonického signálu, pričom modulačný signál je spojitý. Táto skupina modulačných metód sa aplikuje kontinuálne na vstupný analógový informačný signál a ich použitie je notorické pri prenose rozhlasového signálu.

   (b) Skupina digitálnych modulačných techník. Do tejto skupiny spadajú napríklad amplitude-shift keying (ASK, odpovedá AM), frequency-shift keying (FSK, odpovedá FM) a phase-shift keying (PSK, odpovedá PM). Modulačný signál je u tejto skupiny modulačných techník diskrétny, teda môže nadobudnúť len konečného počtu stavov. Digitálne modulačné techniky sa používajú na prenos digitálnych dát po analógovom kanáli s pásmovým priepustom, napríklad po verejnej telefónnej sieti.

2. Diskrétne modulačné techniky – vstupný (nosný) signál nie je spojitý, zvykne sa nazývať taktovací signál. Cieľom týchto metód je modulovanie pulznej vlny tak, že sa prenáša úzkopásmový analógový signál po širokopásmovom analógovom kanáli formou dvojúrovňového signálu. Podobne ako u spojitých modulačných techník sa moduláciou upravujú parametre nosného signálu. Podľa typu modulačného signálu sa delia tieto techniky na:

   (a) Kvantované modulačné techniky – modulačný signál je diskrétny, teda nespojitý. Do tejto skupiny spadajú napríklad pulzne kódová modulácia (PCM), delta modulácia (DM) či delta-sigma modulácia ($\Delta\Sigma$). Delta a delta-sigma modulácie sú použiteľné pre generovanie pulzne šírkovo modulovaného signálu, používaného pre ovládanie serv.

   (b) Nekvantované modulačné techniky – používajú spojitý modulačný signál. Patria sem pulzná amplitúdová modulácia (PAM), pulzná šírková modulácia (PWM) a pulzná polohová modulácia (PPM).

Aj napriek tomu, že všetky spomenuté modulácie majú v praxi svoje využitie, v tejto práci sa budeme zaoberať len pulznou šírkovou moduláciou, ktorou modulovaný signál sa používa na ovládanie serv.

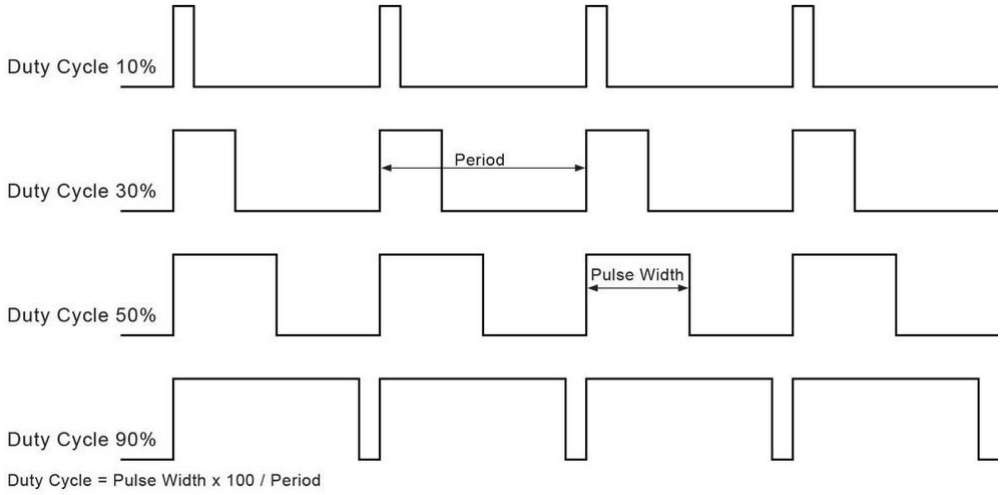
2.5.2 Pulzná šírková modulácia (PWM)

Pulzná šírková modulácia (PWM, Pulse Width Modulation) je diskrétna modulačná technika, ktorá zmení nosný signál na postupnosť impulzov s určitou periódou, pričom sa na

základe nosného signálu mení dĺžka impulzov. Modulovaný signál nadobúda len dva stavy – logická 0 a logická 1. Hodnota prenášaná touto moduláciou je určená pomerom medzi stavmi logickej jednotky a logickej nuly. Tento pomer je tiež známy ako strieda (alebo činiteľ plnenia, duty cycle po anglicky). Periódou u PWM nazývame interval medzi dvoma prenosmi striedy. Strieda sa vypočítava pomocou vzorca

$$D = \frac{\tau}{T} \tag{2.2}$$

kde D je strieda, τ je dĺžka impulzu a T je dĺžka periódy signálu.



Obr. 2.2: Priebeh PWM modulovaných signálov s rôznymi dĺžkami striedi[2]

Pulzná šírková modulácia má okrem riadenia serv aj viacero iných aplikácií. V telekomunikáciách sa používa pri kódovaní prenášaných informácií, taktiež je používaná pri napájaní, regulácii napätia, alebo pri spracovaní zvuku. Využitie pulznej šírkovej modulácie pri napájaní a regulácii napätia je založené na zmene striedy podľa požadovanej úrovne výstupného napätia, pričom sa používajú na výstupe filtre pre zmiernenie kolísania úrovne napätia. Takéto regulovanie je efektívne vďaka tomu, že dvojstavové prvky používané pri práci s pulzne šírkovo modulovanými signálmi majú veľmi nízku stratovosť v porovnaní s lineárnymi regulátormi. V spracovaní zvuku sa pulzná šírková modulácia používa v zosilňovačoch triedy D, ktoré produkujú pulzne šírkovo modulovaný signál ekvivalentný vstupnému analógovému signálu. Tieto zosilňovače sú charakteristické vysokou efektívnosťou ($\geq 90\%$), kompaktnou veľkosťou a nízkou hmotnosťou pre zariadenia s vysokovýkonnostným výstupom. [12, 16]

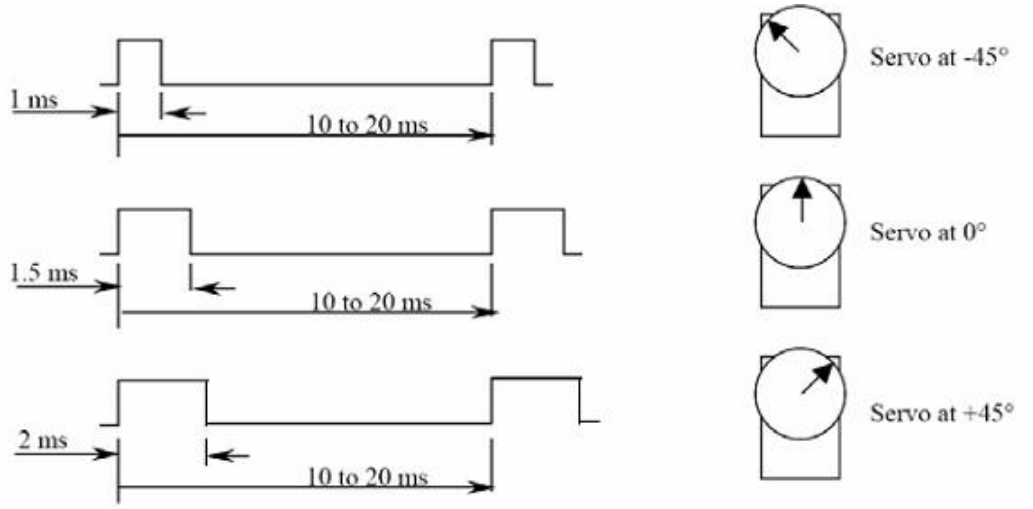
Charakteristiky PWM signálu používaného na riadenie serv

Pulzne šírkovo modulovaný signál, ktorý sa používa na riadenie serv je prenášaný pomocou dátového vodiča (obvykle bielej alebo žltej farby). Perióda používaného PWM signálu je spravidla 20 ms (50 Hz), pričom sa však môže líšiť dĺžka impulzu, ktorá vystaví rameno serva do rovnakej polohy u dvoch rôznych modeloch serv. Dosť často zaužívanou dĺžkou impulzu stredovej polohy (tj. polohy v strede medzi krajnými polohami, do ktorých je servo schopné sa maximálne vytočiť) je 1,5 ms, alebo teda 1500 μs. Skrátením alebo predĺžením

[2]Zdroj: *https://protostack.com.au/wp-content/uploads/atmega168a_pwm_02.jpg*

impulzu sa mení poloha ramena až po krajné polohy, pričom sa pre vystavenie ramena do krajných polôh predĺži (respektíve skráti) impulz obvykle o 0,25–1 ms (250–1000 μs). Posielanie riadiacich impulzov mimo konštrukčných parametrov serva môže poškodiť servo, keďže sa nadbytočne namáha mechanika, ktorá nebola navrhnutá pre vystavovanie ramena do zvolenej pozície a taktiež je celkom bežné, že sa na prevodoch v prevodovke serva môže nachádzať zarážka zabráňujúca pohybu za krajné polohy.

Význam riadiacich impulzov je trochu odlišný u serv s kontinuálne rotačným pohybom. Pri odosielaní impulzov, ktorých dĺžka odpovedá stredovej polohe serva je servo nečinné. Skracovaním alebo predĺžovaním impulzov sa mení rýchlosť a smer rotácie ramena. Odosielaním impulzov o dĺžke odpovedajúcej vystaveniu sa do krajnej pozície sa servo otáča maximálnou možnou rýchlosťou v danom smere. [4, 7, 9, 16]



Obr. 2.3: Zmena polohy ramena serva v závislosti od dĺžky impulzu riadiaceho signálu[3]

2.6 Platformy pre ovládanie serv

Vzhľadom na to, že je potrebné generovať pre každé servo vlastný riadiaci PWM signál, je nepraktické to vykonávať priamo pomocou počítača. Preto sa v súčasnosti pre túto úlohu využívajú rôzne minipočítače, ktoré majú za úlohu generovať riadiace impulzy pre servá a súčasne prijímať a spracovávať riadiace správy z počítača, prípadne ešte môžu vykonávať aplikačne špecifické činnosti, ako napríklad spracovanie dát zo senzorov. V tejto sekcii popíšem dve najpopulárnejšie platformy používané za týmto účelom.

2.6.1 Raspberry Pi

Raspberry Pi je séria jednodoskových malých počítačov, ktoré boli vyvinuté za účelom podpory výučby informatiky na školách. Tieto počítače rýchlo našli uplatnenie aj inde, než len vo výučbe na školách a používajú sa na rôzne účely, od robotiky až po prácu na sieti. Informácie o Raspberry Pi boli získané z portálov Raspberry Foundation [13, 14].

[3]Zdroj: *https://pamungkas99.files.wordpress.com/2010/02/servo2.jpg*

Všetky modely Raspberry Pi bežia na ARM architektúrach, disponujú SDRAM pamäťou, ktorá je zdieľaná s GPU Broadcom VideoCore IV. Tento grafický procesor poskytuje podporu napríklad OpenGL ES 2.0 či rôznym multimediálnym kodekom.

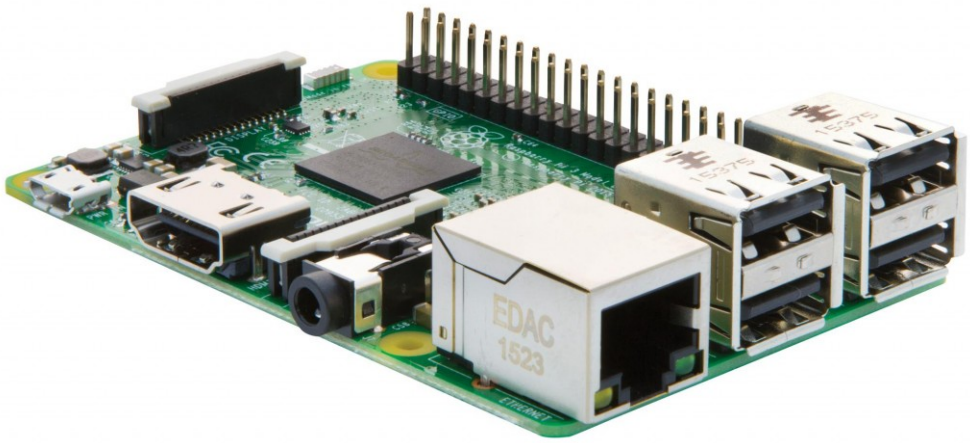
Prvým modelom Raspberry Pi bol model B, ktorý bol vydaný v apríli 2012 a jeho cena bola stanovená na US$35. Tento model disponoval jednojadrovým 32-bitovým ARM procesorom taktovaným na 700 MHz a pamäťou RAM o kapacite 256 MB, ktorú zdieľal s GPU. Taktiež tento model disponoval viacerými portmi, ako napríklad dvomi USB 2.0 portmi, HDMI portom, 3.5 mm jack konektorom pre audio výstup, slotmi pre viacero druhov pamäťových kariet, RJ45 portom či 8 GPIO pinmi. Táto doska však mala dosť vysoký menovitý výkon 3,5 W.

Vo februári 2013 bol vydaný lacnejší model A s cieľovou cenou US$25, ktorý neposkytoval toľko možností pripojenia periférií. Najviac povšimnuteľné bolo zredukovanie počtu USB portov na jeden a odstránenie RJ45 portu, avšak menovitý výkon bol len 1,5 W. Neskôr boli vydané generácie označené plus, ktoré mali určité vylepšenia v porovnaní s originálmi, ako napríklad ďalšie zníženie spotreby či zvýšenie počtu portov.

Taktiež boli vydané verzie v podobe výpočetných modulov, ktoré sa pripájali pomocou 200-pinového DDR2 DIMM konektora a ich cena sa pohybovala okolo US$30 (u lite verzie tretej generácie to bolo US$25). Tieto moduly poskytujú výpočetný výkon porovnateľný s výpočetným výkonom modelov B. Taktiež sú vydané typy Raspberry Pi s označením Zero, ktoré obsahujú jednojadrový 32-bitový procesor taktovaný na frekvenciu 1 GHz. Tieto modely majú cenu stanovenú na US$5, respektíve US$10 pre modely s bezdrôtovým pripojením.

Tretia (súčasná) generácia modelu B disponuje 64-bitovým procesorom taktovaným na frekvenciu 1,2 GHz, 1 GB SDRAM pamäte zdieľanej s GPU, slotom pre MicroSDHC kartu, RJ45 prípojkou, 802.11n bezdrôtovým adaptérom, či Bluetooth 4.1.

Raspberry Pi navyše umožňuje inštaláciu operačného systému podľa vlastnej vôle, pričom vďaka vysokému výpočetnému výkonu tejto platformy je možný výber z viacerých variánt, napríklad FreeBSD, CentOS, či Fedora 25. Vďaka možnosti pripojenia monitoru či myši a klávesnice je možný vývoj programu priamo na tejto doske.



Obr. 2.4: Raspberry Pi Model B tretej generácie[4]

[4]Zdroj: *http://www.raspberrypi-spy.co.uk/wp-content/uploads/2016/02/raspberry_pi_3_cpc_02-1024x470.jpg*

2.6.2 Arduino

Arduino je hardvérový a softvérový projekt, ktorý je založený na tvorbe mikrokontrolérových počítačov, ktoré sú navhrované za účelom tvorby zariadení schopných vnímať objekty v reálnom svete a manipulovať s nimi. Je spravovaný spoločnosťou so zhodným názvom a používateľskou komunitou. Celý tento projekt je na bázi open-source pod GNU Lesser General Public licenciou (LGPL), čím je umožnené hocikomu tvoriť Arduino dosky a distribuovať ich. Poznatky o Arduine a jeho vývojovom prostredí boli získané z [2, 3, 18].

Arduino dosky používajú rôzne mikroprocesory a ovládače a sú vybavené ako digitálnymi, tak aj analógovými vstupnovýstupnými pinmi. Tieto piny je možné použiť pre pripojenie sa k rozličným perifériám, ktorými môžu byť napríklad rozširujúce dosky (tiež známe pod anglickým termínom shields), či rozličné vlastné elektronické obvody. Do Arduino dosiek sa program nahráva z osobných počítačov prostredníctvom sériového rozhrania, ktorým je obvykle Universal Serial Bus (USB) rozhranie, ak je teda prístupné na danom modeli. Okrem toho tieto dosky obsahujú aj rôzne iné rozhrania pre sériovú komunikáciu.

Vývoj programov pre dosky Arduino sa obvykle vykonáva v jazykoch, ktoré sú dialektmi jazykov C a C++. Za účelom tvorby programov pre Arduino bolo vytvorené integrované vývojové prostredie (IDE) s názvom Arduino IDE, ktoré používateľom značne zjednodušuje vývoj a nahrávanie programov. Toto IDE je založené na základe open-source projektu Processing, ktorý je programovacím jazykom a tiež aj IDE pre tento jazyk. Tvorba programov bude podrobnejšie opísaná v nasledujúcej subsekcii.

Väčšina Arduino dosiek je vybavená 8-bitovým AVR mikrokontrolérom od spoločnosti Atmel, konkrétne z ATmega série, a sú taktované na frekvenciu 16 MHz. Taktiež má väčšina dosiek zabudovaný 5 V lineárny napäťový regulátor. Výnimkou sú napríklad modely menších rozmerov, ktoré kvôli rôznym obmedzeniam, akými je napríklad tvar dosky, nemôžu byť taktované na 16 MHz a teda sú taktované na nižšiu frekvenciu 8 MHz. Inou výnimkou je napríklad Arduino Due, ktoré je určené pre výpočetne náročnejšie aplikácie a je vybavené 32-bitovým ARM mikrokontrolérom a je taktované na podstatne vyššiu frekvenciu, konkrétne 84 MHz.

V porovnaní s Raspberry Pi majú Arduino dosky znateľne nižší výkon. Taktovacia frekvencia dosiek Arduino sa hýbe v úrovni jednotiek až desiatok MHz, flash pamäť pre program sa hýbe v úrovni desiatok až stoviek kB, EEPROM pamäť do zopár kB a SRAM pamäť v úrovni jednotiek kB. Avšak vďaka tomu, že Arduino je open-source projekt, existuje veľa klonov populárnych Arduino dosiek, ktoré je možné zakúpiť za zopár dolárov.

Väčšina I/O vstupov mikrokontroléru na Arduine je prístupná pre použitie inými elektronickými okruhmi. Napríklad Arduino Uno obsahuje 14 digitálnych pinov, z ktorých šesť dokáže produkovať pulzne šírkovo modulované signály (ktoré sa používajú na riadenie serv), alebo šesť vstupných analógových pinov, pomocou ktorých je možné prijímať informácie ako napríklad signály zo senzorov.

Vývoj programov pre dosky Arduino

Na rozdiel od Raspberry Pi, dosky Arduino nemajú dostatok výkonu a funkcií na to, aby bolo možné vyvíjanie programov priamo na samotnom Arduine. Preto sa programy pre Arduino vyvíjajú na osobných počítačoch a nahrávajú sa na dosku prostredníctvom sériového rozhrania, akým je napríklad Universal Serial Bus (USB). Programy pre Arduino je možné písať v ľubovoľnom programovacom jazyku, pre ktorý existuje kompilátor schopný skompilovať zdrojový kód do binárného kódu určeného pre cieľovú architektúru. Pre procesory od firmy Atmel sa takto dá vyvíjať v AVR Studiu, alebo Atmel Studiu. Obvykle

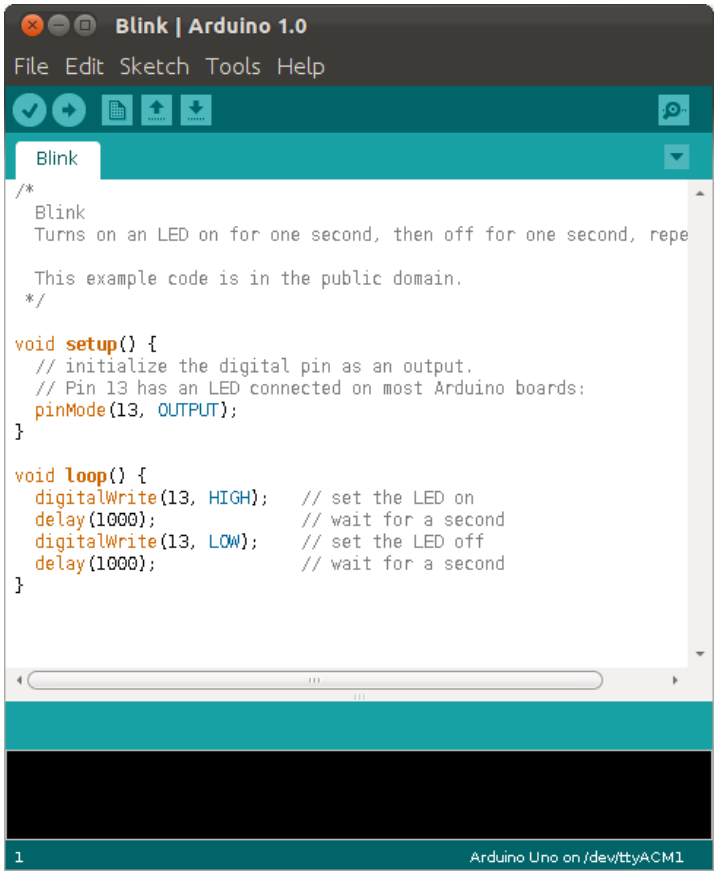
Obr. 2.5: Arduino Uno[5]

sa však pre tvorbu programov pre Arduino používa Arduino IDE (Integrated Development Environment = integrované vývojové prostredie).

Arduino IDE je multiplatformná aplikácia umožňujúca vývoj programov pre Arduino, ktorá pochádza z IDE pre jazyky Processing a Wiring. Základom tohto IDE je editor kódu, ktorý okrem tradičných operácií s textom ako sú vystrihovanie, kopírovanie, prilepovanie či hľadanie slov poskytuje aj viac programátorsky zamerané funkcie, akými sú napríklad automatické odsadzovanie, spájanie zátvoriek či zvýrazňovanie syntaxe. Taktiež je v Arduino IDE možné verifikovať, kompilovať a nahrať kód jediným kliknutím tlačidla, čo zjednodušuje vývoj programov. Editor obsahuje aj okno so správami, do ktorého sa zapisujú správy o aktuálnom stave kompilácie, alebo prípadné chyby. Ďalej Arduino IDE poskytuje funkcie ako napríklad monitorovanie sériového portu, či odosielanie dát cez sériové rozhranie. Tiež je poskytnutá možnosť si vybrať programátor z dostupnej ponuky.

V Arduino IDE je možné vyvíjať pomocou jazykov C alebo C++, pričom sa používajú mierne upravené pravidlá štruktúry programu. Zmena voči tradičným programom písaným v C alebo C++ je v tom, že sa pre beh programu nepoužíva funkcia `main()`, ale používajú sa dve funkcie: `setup()` a `loop()`. Funkcia `setup()` sa zavolá len raz po spustení programu a slúži pre inicializáciu premenných, prípadne pre vykonanie iných inicializačných operácií. Následne sa po vykonaní funkcie `setup()` volá funkcia `loop()` v nekonečnom cykle, ktorá už vykonáva bežnú funkciu programu.

[5]Zdroj: *https://cdn.solarbotics.com/products/photos/a0266346bdc1b2028b4066554730ddfa/50450-IMG_5222.jpg*



Obr. 2.6: Arduino IDE bežiace na OS Ubuntu s ukážkovým programom Blink[6]

Arduino IDE používa softvérovú knižnicu z projektu Wiring, ktorá poskytuje množstvo procedúr, akými sú napríklad vstupnovýstupné operácie. Tiež sú do Arduino IDE zabudované knižnice, ktoré poskytujú rôznu funkcionalitu pre časté aplikácie Arduina.

2.7 Spôsoby prenosu dát z PC na riadiacu platformu

Pri používaní mikroprocesorových platforiem na generovanie riadiacich impulzov pre servá je často žiadané, aby im bolo umožnené komunikovať s počítačom, ktorý môže napríklad spracovávať výstupy z riadiacej platformy, alebo odosielať dáta do riadiacej platformy (akými sú napríklad pokyny pre ovládanie serv).

2.7.1 Sériové rozhranie

Jednou možnosťou spojenia riadiacej platformy s počítačom je využitie sériového rozhrania. Pod termínom sériové rozhranie sa rozumie také rozhranie, ktoré prenáša dáta v poradí po jednotlivých bajtoch. Vďaka tomu, že sa dáta neprenášajú pomocou viacerých vodičov, zanikajú niektoré problémy prenosu dát, ktoré sa vyskytovali pri prenose pomocou paralelných rozhraní. Taktiež to značne znižuje nutný počet vodičov potrebných pre komunikáciu.

[6]Zdroj: *https://upload.wikimedia.org/wikipedia/en/d/d9/Arduino_1.0_IDE%2C_Ubuntu_11.10.png*

Príkladmi sériových rozhraní sú napríklad Ethernet, FireWire, či USB. Použiteľnosť prepojenia riadiacej platformy s počítačom prostredníctvom sériového rozhrania spočíva v tom, že riadiace platformy sú väčšinou vybavené aspoň jedným sériovým rozhraním, pomocou ktorého sa napríklad nahráva do platformy program.

Takáto forma spojenia má však aj určité problémy. Jednou nevýhodou je to, že je nutné fyzické spojenie káblom medzi počítačom a platformou. V prípade zariadení, kde sa nemení pozícia platformy to nie je veľký problém, pokiaľ nie je vyžadované aby bolo zariadenie vo väčšej vzdialenosti od platformy, alebo aby bola platforma na mieste, kam je problematické zaviesť kábel. V prípade pohyblivých zariadení, akými sú napríklad modely lodí, automobilov či lietadiel, je to však znateľne väčší problém, vzhľadom na to, že to limituje maximálnu možnú vzdialenosť medzi vozidlom a počítačom. Ďalšou nevýhodou tohto typu spojenia je to, že sa dáta musia na strane odosielateľa pred prenosom serializovať a následne sa musia po prenose na strane prijímateľa deserializovať. [19]

Serializácia a deserializácia

Serializácia a deserializácia sú procesy, ktoré sú vzájomne opačné. Serializácia je proces transformácie objektu z pamäti do podoby, v ktorej je možné ho uložiť do súboru, alebo preniesť. Používa sa teda pred prenosom dátových štruktúr pomocou sériových rozhraní. Deserializácia túto postupnosť prijatých bajtov spracuje a zrekonštruuje z nich objekt, ktorý bol na strane odosielateľa serializovaný. Medzi bežné formáty využívané na serializáciu patria XML, JSON či YAML.

2.7.2 Bluetooth

Bluetooth je štandard bezdrôtovej technológie, v minulosti štandardizovaný organizáciou IEEE ako IEEE 802.15.1 a teraz je spravovaný skupinou zvanou Bluetooth Special Interest Group (SIG). Bluetooth bol vyvinutý ako náhrada RS-232 sériového rozhrania a je určený k prenosu dát na krátke vzdialenosti.

Bluetooth využíva k prenosu frekvencie v pásme od 2,4 GHz do 2,485 GHz. Dáta sú prenášané po 79 kanáloch, pričom sa pri prenose paketov mení kanál, na ktorom bude paket prenesený. V minulosti sa používala len frequency-shift keying (FSK) modulačná technika, ale v súčasnosti sa používa aj phase-shift keying (PSK) modulačná technika, ktorá umožňuje vyššiu prenosovú rýchlosť. Bluetooth používa master/slave model komunikácie, teda počas komunikácie má jedno zariadenie autoritu a ovláda prenos dát, pričom sa druhé zariadenie prispôsobuje a používa pri prenose synchronizáciu od zariadenia v roli mastera. Zariadenia si môžu počas prenosu tieto role vymeniť.

Dosah Bluetooth vysielačov je závislý na viacerých faktoroch, akými sú napríklad stav batérie, konfigurácia antény, zakrytie vysielača materiálmi či samotné prostredie. Najpodstatnejšia je však výkonnostná trieda vysielača. Oficiálne má trieda 1 dosah až 100 metrov, pričom zariadenia s touto triedou by mali byť určené hlavne pre priemyselné použitie. Bežne sa však udáva dosah vysielačov triedy 1 na 20–30 m. Trieda 2 má dosah do 10 m, táto trieda vysielačov je najčastejšie nájdená v mobilných zariadeniach. Vysielače triedy 3 majú dosah do 1 m. Obecne platí, že maximálny povolený výkon vysielača v mW je zhodný s približným dosahom jeho triedy v metroch.

Prenosová rýchlosť je závislá hlavne na verzii Bluetooth. Bluetooth verzie 1.2 mal prenosovú rýchlosť do 1 Mbit/s. U verzie 2.0 + EDR (Enhanced Data Rate) bola rýchlosť až do 3 Mbit/s. Podstatné zvýšenie rýchlosti nastalo vo verzii 3.0, kde bola maximálna prenosová rýchlosť až 25 Mbit/s. Verzia 4.0 síce nenavýšila rýchlosť, ale zvýšila maximálny dosah a

priniesla kategóriu Bluetooth pod označením LE (Low Energy) pre zariadenia s nízkym odberom energie. Najnovšia verzia Bluetooth, verzia 5, ďalej zvýšila maximálnu prenosovú rýchlosť na 50 Mbit/s a priniesla nové funkcie určené pre IoT (Internet of Things) aplikácie Bluetooth.

V porovnaní s bežnými sériovými rozhraniami má Bluetooth výhodu v tom, že odpadá potreba fyzického spojenia platformy s počítačom, avšak stále existuje obmedzenie na maximálnu vzdialenosť medzi Bluetooth prijímačom a počítačom. Zostáva taktiež potreba serializácie a deserializácie dát pri prenose. Ďalším rizikom Bluetooth je to, že existuje hrozba útoku, respektíve neželaného pripojenia sa iným zariadením. Tento problém sa často rieši pomocou takzvaného párovania, napríklad pomocou PIN kódov alebo pomocou porovnávania čísel. [5, 6]

Kapitola 3

Analýza a návrh riešenia

Ako stroj, ktorý bude v rámci tejto práce skonštruovaný, som si zvolil model automobilu. Požiadavky na tento stroj boli:

- Pohyb má byť zabezpečený prostredníctvom serv
- Vozidlo by malo byť schopné pohybu tak, aby nebolo pripojené káblom k zdroju napájania alebo počítaču. Z toho plynie, že:
  - Ako zdroj napájania bude potrebné použiť batérie
  - Signály na riadenie serv bude potrebné generovať na riadiacej platforme, ktorá bude uložená na vozidle
  - Komunikácia platformy s počítačom bude musieť prebiehať prostredníctvom bezdrôtového spojenia
- Cena zrovnateľná s podobnými komerčnými produktami
- Prenositeľnosť

Okrem toho však ešte je potrebné vytvoriť program pre riadiacu platformu, ktorý bude prijímať správy od počítača a ovládať servá. Taktiež je potrebné vytvoriť program pre počítač, pomocou ktorého bude možné toto zariadenie ovládať. Nakoniec je ešte potrebné navrhnúť formát, v ktorom sa budú prenášať dáta do riadiacej platformy.

3.1 Možnosti prenosu inštrukcií pre pohyb

Existuje viacero spôsobov, akými je možné riadiť vozidlo pomocou počítača. V závislosti od spôsobu je rozdiel nielen v tom, koľko sa prenesie dát, avšak tiež aj v pomere spracovávania dát na strane odosielateľa a na strane prijímateľa.

Najprimitívnejším spôsobom by bolo odosielať priamo šírku signálu pre každé servo, ktoré je osadené na stroji. Tento spôsob vyžaduje minimálne spracovanie dát na strane prijímateľa a taktiež pre riadiacu platformu nie je potrebné uchovávať takmer žiadne informácie o servách, ktoré sú pripojené, výnimka informácií o tom, na ktorých pinoch sú tieto servá pripojené. Tento spôsob prenosu vyžaduje nejaký spôsob očíslovania serv a odosielanie dĺžky riadiaceho impulzu pre servo, čo môže byť spolu niekoľko bajtov odoslaných dát pre každé servo.

Alternatívnym spôsobom odosielania dát o vystavení serv by bolo posielanie vystavenia každého serva v stupňoch, avšak to by už vyžadovalo, aby riadiaca platforma uchovávala v sebe informácie o parametroch serv.

Ďalším spôsobom je odosielanie informácií o tom, ktorý typ pohybu je aktívny. Pre túto možnosť vzniká ten problém, že už je potrebné v riadiacej platforme uchovávať informácie nielen o tom, na akých pinoch sú servá pripojené, ale aj informácie o dĺžke impulzu pre stredovú polohu serva a rozsah dĺžky impulzov pre krajné hodnoty. Navyše však vzniká aj nutnosť udržiavania informácií o tom, že ktoré servá sú akým spôsobom využívané pri každom type pohybu. Tiež je potrebné na riadiacej platforme prepočítavať podľa aktívnych typov pohybu, že aká dĺžka impulzu sa má odosielať pre jednotlivé servá.

Prenos informácií o aktívnych typoch pohybu sa líši v počte úrovní, ktoré dokáže aplikácia na strane počítača a riadiaca platforma rozoznať pre každý typ pohybu. Pri rozoznávaní dvoch úrovní pohybu (aktívny, neaktívny) by bolo možné prenášať informácie o každom type pohybu na jedinom bite, čo by bolo mimoriadne efektívne pre prenos a tiež nenáročné pre výpočet riadiacich impulzov pre servá. Takýto spôsob prenosu by bolo možné jednoducho použiť v prípade, že by sa rozlišoval počet úrovní rovný mocnine čísla dva, čím by bolo možné jednoducho kódovať tieto informácie do bajtov. Problémom tohto spôsobu prenosu bez číslovania typu pohybov je to, že je nutné prenášať aj informácie o typoch pohybu, ktoré sú na hladine nula, teda neaktívne. Pri rozlišovaní väčšieho počtu úrovní by však už mohlo byť výhodnejšie číslovať jednotlivé typy pohybu a neaktívne typy pohybu ignorovať, čím by sa znížil počet prenášaných dát.

3.2 Návrh programu pre riadiacu platformu

Pre návrh programu pre riadiacu platformu je dôležité určiť, čo všetko má byť schopná táto platforma vykonávať a taktiež od toho bude závisieť aj prenosový protokol. V prípade, že riadiaca platforma má byť schopná spracovávať aj riadiace inštrukcie určujúce aktívne typy pohybu, alebo inštrukcie vystavenia ramena serva do určitého uhlu, vzniká potreba uchovávania informácií o servách (respektíve typoch pohybu) v riadiacej platforme. Za týchto okolností už vzniká požiadavka na možnosť prenosu týchto informácií o typoch pohybu, alebo o parametroch serv.

Program pre riadiacu platformu musí byť teda schopný:

- Prijímať správy prostredníctvom bezdrôtového rozhrania
- Spracovávať prichádzajúce správy, ktorým by mala platforma rozumieť
- Vypočítať dĺžku riadiaceho impulzu pre každé servo, ktorý vystaví rameno do požadovanej polohy a tieto impulzy odosielať na zvolených pinoch
- Zastaviť vozidlo v prípade, že nebola dlšiu dobu obdržaná riadiaca správa pre pohyb

Vzhľadom na to, že cieľom tejto práce je tvorba jednoduchého vozidla, nie je potrebné implementovať ďalšiu funkčnosť.

3.3 Aplikácia pre osobný počítač

Na rozdiel od programu pre riadiacu platformu, na túto aplikáciu je počet požiadavok vyšší. Od tejto aplikácie sa požaduje, aby dokázala:

- Spracovávať vstupy používateľa pre ovládanie vozidla
- Umožňovať pripojenie sa k zariadeniu prostredníctvom bezdrôtového rozhrania – v prvom rade musí byť teda schopná interakcie s bezdrôtovým rozhraním na počítači
- Tvoriť správy podľa prenosového protokolu, ktorým bude komunikácia s platformou prebiehať
- Zmeniť nastavenia serv a toho, ktoré servá sa ako používajú pri jednotlivých typoch pohybu
- Odosielať správy obsahujúce nastavenia do riadiacej platformy. To platí za predpokladu, že má byť podporovaný jeden z typov prenosu informácií o pohybe, ktorý vyžaduje uchovávanie informácií o servách, respektíve typoch pohybu na riadiacej platforme.

Je teda potrebné zvoliť vývojové prostredie, ktoré umožňuje splnenie týchto požiadavok pre aplikáciu riadiacu vozidlo zo strany počítača.

3.4 Návrh konštrukcie

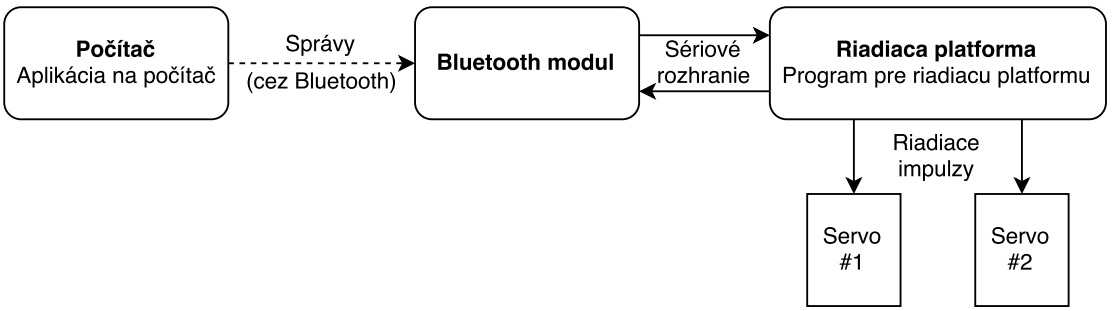
Mimo programovej časti a návrhu protokolu je však ešte potrebné skonštruovať vozidlo. Základná predstava o vozidle je taká, že bude mať dve nápravy – prednú a zadnú. Rovnako ako u bežných motorových vozidiel, predná náprava bude určená k zatáčaniu vozidla a zadná náprava bude slúžiť k pohonu vozidla. Ďalej je potrebné skonštruovať rám vozidla, na ktorý sa následne umiestnia komponenty vozidla, akými sú napríklad riadiaca platforma, modul pre bezdrôtovú komunikáciu, servá a iné. Navyše sa očakáva, že vozidlo bude dostatočne robustné na to, aby sa počas prevádzky nerozložilo na diely. Je teda požadované, aby bola konštrukcia vozidla vytvorená dielmi z pevných materiálov, na ktoré je možné pripevniť elektronické a mechanické komponenty zariadenia.

Kapitola 4

Realizácia návrhu

V tejto kapitole objasním, ktoré prostriedky boli zvolené pre realizáciu tejto práce a taktiež vysvetlím spôsob implementácie programov pre počítač a pre riadiacu platformu.

4.1 Hardvérová časť



Obr. 4.1: Schéma fungovania systému

4.1.1 Výber súčiastok

Pre konštrukciu môjho zariadenia som sa rozhodol, že použijem nasledujúce komponenty:

- **Riadiaca platforma** – vzhľadom na to, že nie je pre túto aplikáciu vyžadovaný vysoký výkon riadiacej platformy, som sa rozhodol pre Arduino Uno. Mimoriadne nízka cena jeho klonov a dostupnosť veľkého množstva knižníc, medzi ktoré patrí aj manipulovanie so servami, ho robí skvelou voľbou pre toto použitie. Taktiež sa na doske nachádza viacero sériových rozhraní, ku ktorým je možné pripojiť modul pre bezdrôtovú komunikáciu.
- **Bezdrôtový modul** – zvolil som Bluetooth modul HC-05. Tento modul, klasicky používaný pre podobné typy aplikácií, je možné jednoducho používať spolu s Arduinom a tiež je lákavou voľbou vzhľadom na jeho nízku cenu. Bohužiaľ, tento modul podporuje len Bluetooth verzie 2.1, avšak aj to je dostačujúce pre prenos dát pri bežnom používaní aplikácie.

- **Konštrukcia** – na konštrukciu rámu vozidla som si zvolil stavebnicu Merkur. Táto stavebnica umožňuje jednoduché spájanie dielov a teda aj konštruovanie rôznych modelov. Taktiež je vhodná z toho hľadiska, že servá štandardnej veľkosti majú obvykle otvory na prichytenie k modelu na vrchnej časti aj na spodnej časti, pričom na oboch stranách sú zvyčajne otvory pre pripevnenie dvoma skrutkami. Tieto otvory zvyknú mať medzi sebou rovnakú vzdialenosť, ako majú aj otvory na stavebnici Merkur. To umožňuje jednoduché prichytenie serv k modelom zloženým touto stavebnicou.

- **Servá** – výber serv je dôležitý, keďže tvoria základ motorickej časti vozidla. Nesprávna voľba serv môže v lepšom prípade spôsobiť len obmedzenie funkčnosti vozidla, v horšom prípade môže viesť aj k serióznemu poškodeniu zariadenia (napríklad u lietajúcich modelov). Zvolil som dve servá, ktoré sú minimom pre tvorbu ovládateľného vozidla – jedno pre vytáčanie prednej nápravy a druhé pre náhon zadnej nápavy. Vzhľadom na to, že u jazdiaceho modelu nie je voľba serv až tak kritická ako u iných typov modelov, som pri voľbe serv nedbal až tak na kvalitu ich prevedenia, než na pomer cena/výkon. Volil som servá dostatočne silné pre vykonávanie ich práce. Jedná sa o servá DS04-NFC a TowerPro MG996R, ktorých parametre uvádzam v tabuľke.

Servo	TowerPro MG996R	DS04-NFC
Veľkosť	štandard	štandard
Hmotnosť [g]	55	38
Rozmery (V x D x Š) [mm]	42.9 x 40.7 x 19.7	54 x 44 x 20
Ťah [kg/cm]	9.4 (pri 4.8 V), 11 (pri 6 V)	5.5 (pri 4.8 V)
Operačná rýchlosť [s/60°]	0.17 (pri 4.8 V), 0.14 (pri 6 V)	0.22 (pri 4.8 V)
Operačné napätie [V]	4.8–7.2	4.8–6
Odber prúdu [mA]	500–900	<1000
Prevádzková teplota [°C]	0–55	0-60

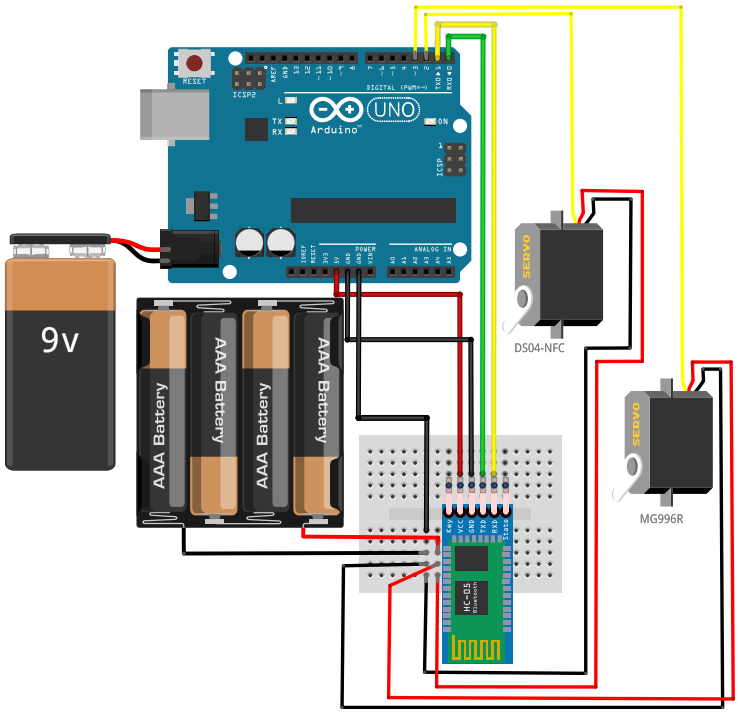
Tabuľka 4.1: Zvolené servá

Servo TowerPro MG996R je určené pre zatáčanie, keďže je to servo s pozične rotačným typom pohybu. Druhé servo, DS04-NFC, určené pre náhon vozidla, je servom s kontinuálne rotačným typom pohybu.

4.1.2 Konštrukcia vozidla

Pri konštrukcii boli použité dve malé sady stavebnice Merkur. Jedná sa o sady M 016 Buggy a Machinery Set Basic. Z dielov týchto dvoch spojených stavebníc bol následne skonštruovaný rám vozidla. Tento rám bol ďalej osadený servami a na jeho zadnú stranu bola pridaná zadná náprava. K servu určenému na zatáčanie bola pripojená predná náprava. Následne bolo na rám z vnútornej strany pripojené nepájivé kontaktné pole a Arduino bolo pripevnené na vrchnú dosku rámu vozidla. Taktiež boli pridané kladky a hnací remeň (ktorým je gumička). Napokon bol pridaný držiak na batérie a bola zapojená elektronika. Schému zapojenia elektroniky je možné vidieť na obrázku 4.2.

Po vykonaní týchto krokov je skonštruované vozidlo schopné prevádzky, pričom je ho možné vidieť na obrázku 4.3. Tento model bol ďalej však ešte doplnený o ďalšie diely za účelom zakrytia vnútorných dielov, čím by sa mali chrániť komponenty modelu.



Obr. 4.2: Schéma zapojenia elektroniky

4.2 Softvérová časť

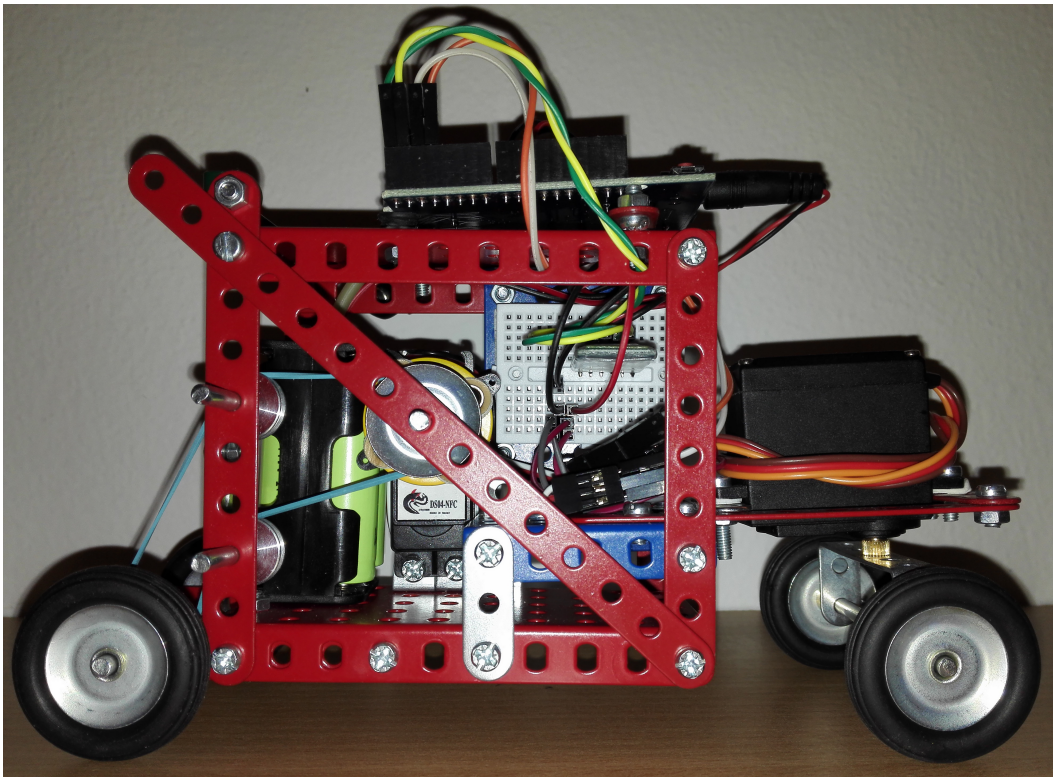
Softvérová časť, rozdelená na dve časti – aplikáciu pre počítač a program pre riadiacu platformu – bola navrhnutá tak, aby podporovala šesť rôznych typov pohybu a šesť rôznych serv. Tieto počty podporovaných serv a typov pohybu boli zvolené pre to, aby bolo možné prezentovať funkčnosť navrhnutého protokolu a jeho jednoduchú rozšíriteľnosť. Okrem toho ešte táto sekcia obsahuje popis formátu prenášaných správ z počítača do riadiacej platformy.

4.2.1 Formát prenášaných dát

Prenosový protokol musí byť tvorený na základe toho, aké správy má byť riadiaca platforma schopná spracovať. Rozhodol som sa, že budem pre moju prácu podporovať ako prenos dĺžky riadiacich impulzov pre polohovanie serv, tak aj prenos aktívnych typov pohybu. Vzhľadom na podporu prenosu aktívnych typov pohybu je však nutné podporovať na riadiacej platforme aj uchovávanie informácií o servách a pohyboch. Kvôli tomu je požadované, aby bol možný aj prenos nastavení.

Vymyslel som teda protokol, ktorý umožní prenos všetkých týchto informácií. Základom správy je dvojica *(ID_typu_správy, telo_správy)*, kde *ID_typu_správy* je číselné označenie typu správy a *telo_správy* je obsah správy charakteristický pre daný typ.

Prvým typom správy je prenos dĺžky riadiacich impulzov pre servá. Pre tento typ správy telo obsahuje najprv počet serv, pre ktoré sú informácie prenášané a za tým nasleduje toľko dvojíc *(ID_serva, dĺžka_impulzu)*, koľko bolo serv určených na začiatku tela správy. *ID_serva* je očíslovanie serva, pre ktoré je tento impulz určený a *dĺžka_impulzu* určuje dlžku riadiaceho impulzu udávanú v μs, ktorá má byť odosielaná do serva.



Obr. 4.3: Základný pojazdný model

Druhým typom správy je prenos informácií o aktívnych typoch pohybu. Obsah tela správy je podobný tomu, ktorý sa nachádza u prvého typu správy – prenosu dĺžky riadiacich impulzov. Na začiatku tela správy sa nachádza počet typov pohybu, o ktorých sú v správe informácie. Nasleduje rovnaký počet dvojíc *(ID_typu_pohybu, úroveň_pohybu)*. *ID_typu_pohybu* udáva číselné označenie označenie typu pohybu (napríklad podľa uloženia v nastaveniach), pre ktorý je táto úroveň určená. Nasledujúca *úroveň_pohybu* udáva mieru, v akej sa má tento typ pohybu vykonávať.

Posledným (tretím) typom správy je prenos nastavení. Vzhľadom na to, že môže existovať viacero druhov nastavení, ktoré sa budú prenášať, je tento typ správy najkomplikovanejší. Obsah tela správy je zložený ľubovoľným počtom segmentov obsahujúcich nastavenia určitého typu (konkrétne sa jedná o nastavenia parametrov serv, nastavenia typov pohybu a nastavenie maximálnej doby bez prijatej správy z počítača). Každý segment nastavení sa podobne ako samotná správa skladá z dvojice *(ID_typu_nastavení, telo_segmentu)*.

ID_typu_nastavení je očíslovaný typ nastavení. Môže to byť jedna z možností:

1. Parametre serv – slúži k prenosu informácií o servách pripojených k riadiacej platforme. Na začiatku tela segmentu je číslo, ktoré udáva o koľkých servách sa budú informácie prenášať. Nasleduje toľko sekcií obsahujúcich informácie o jednotlivých servách, koľko udáva toto číslo. Každá z týchto sekcií obsahuje:

   (a) číselné označenie serva (napríklad vychádzajúce z poradia uloženia v nastaveniach), pre ktoré sú tieto nastavenia prenášané

(b) údaj o tom, či je dané servo pripojené alebo nie (pre odpojené servá sa negenerujú riadiace impulzy)

(c) dĺžku riadiaceho impulzu pre vystavenie serva do stredovej polohy (dĺžka impulzu sa udáva v μs)

(d) rozsah riadiacich impulzov od stredovej polohy po krajné polohy (tiež udávaný v μs)

V prípade potreby by bolo možné prenášať v tejto sekcii aj ďalšie informácie o servách, akými by mohli byť napríklad číslo digitálneho pinu, ku ktorému má byť toto servo pripojené a podobne.

2. Parametre typov pohybu – podobne ako u parametrov serv, na začiatku tela segmentu sa prenáša počet typov pohybu, o ktorých sa budú informácie v tomto segmente prenášať. Nasleduje zhodný počet sekcií, pričom každá z týchto sekcií obsahuje nasledujúce informácie o daných typoch pohybu:

   (a) číselné označenie typu pohybu (napríklad podľa uloženia v nastaveniach)

   (b) údaj o tom, či je tento typ pohybu povolený

   (c) prioritu typu pohybu, ktorá sa využíva v prípade, že sa určité servo používa viacerými typmi pohybu, ktoré sú v danej chvíli aktívne

   (d) informácie o tom, ktoré servá sú pre tento typ pohybu využívané a v akom pomere – pre každé servo (aj nepoužité) sa prenáša dvojica *(ID_serva, pomer_využitia)*. Z tejto dvojice má *ID_serva* rovnaký význam, ako u prenosu parametrov serv či pri prenose riadiacich impulzov pre servá – jedná sa o číselné označenie serva. *Pomer_využitia* je číslo v intervale $\langle -1,0; 1,0\rangle$, ktoré udáva do ktorého smeru sa má vystaviť rameno serva pri pohybe a v akej miere.

3. Maximálna povolená odozva – telo segmentu obsahuje len jediné číslo. Toto číslo udáva dobu od okamihu poslednej prijatej správy, po ktorú sa maximálne bude uchovávať aktuálny stav pohybu v prípade, že medzitým nepríde žiadna ďalšia správa s pokynmi o riadení pohybu. Táto doba sa udáva v milisekundách. V prípade že uplynie dlhšia doba, než je maximálna povolená odozva, sa všetky aktivované servá vystavia do stredovej polohy až do okamihu príjmu ďalšej riadiacej správy pre pohyb.

4. Ukončujúci znak – slúži pre ukončenie obsahu tela správy pri prenose nastavení. Telo segmentu s týmto typom nastavení je prázdne.

Tento prenosový protokol je teda schopný prenášať požadované typy správ. Výhodou takto navrhnutého protokolu je malá veľkosť prenášaných správ, ktorá sa dá ďalej zmenšiť prenosom len nutných informácií, ak je na to pripravený program na riadiacej platforme. Napríklad je možné prenášať v správach pre vystavenie serv len informácie o tých servách, ktoré sú aktivované (vzhľadom na to, že vystavenie deaktivovaných serv sa aj tak nevykoná) a ktoré nie sú na stredovej polohe. Ďalej je pri prenose správ o typoch pohybu možné prenášať len aktuálne aktívne typy pohybu. Hlavným zefektívnením je však prenos len tých nastavení, ktoré sa zmenili. To však vyžaduje, aby boli počiatočné nastavenia zhodné ako na počítači, tak aj na riadiacej platforme. Ďalej je potrebné, aby program na počítači bol schopný rozlíšiť ktoré konkrétne nastavenia sa zmenili. Vďaka tomu je napríklad možné poslať len nastavenia o jednom serve, ktorého konfigurácia sa zmenila, namiesto odosielania všetkých nastavení.

Ďalšou výhodou tohto protokolu je rozšíriteľnosť. Vďaka tomu, že sa typy správ pri prenose odlišujú enumerovaným typom, je v prípade potreby možné jednoducho tvoriť ďalšie typy správ. Podobne je možné rozširovať nastavenia a jednoducho ich prenášať.

4.2.2 Aplikácia pre počítač

Pre tvorbu aplikácie na počítač som zvolil Qt Toolkit. Qt je multiplatformé vývojové prostredie, ktoré podporuje širokú škálu platform a operačných systémov, pričom nie je vyžadovaná takmer žiadna zmena kódu pri prechode na iné platformy. Qt je dostupný buď s komerčnou licenciou, alebo tiež pod open source licenciami. Vhodnosť Qt pre toto použitie spočíva v tom, že má dosť veľký počet modulov, ktoré umožňujú vykonávať rôznu funkcionalitu, akou môže byť napríklad vyžadovaná komunikácia prostredníctvom Bluetooth rozhrania. [1]

Qt Bluetooth, teda modul umožňujúci komunikáciu prostredníctvom Bluetooth, je však aktuálne ešte vo vývoji a teda nie sú podporované niektoré platformy pre určité súčasti Bluetooth API. Konkrétne sa jedná o Windows (ktorý vôbec nie je podporovaný), alebo WinRT (ktorý je podporovaný len pre základ Bluetooth LowEnergy). Vzhľadom na to, že použitý HC-05 modul využíva Classic Bluetooth API, teda nebolo možné vyvíjať túto aplikáciu pre Windows. Vývoj aplikácie prebiehal pre operačný systém Fedora 25, ktorý je založený na Linux kerneli.

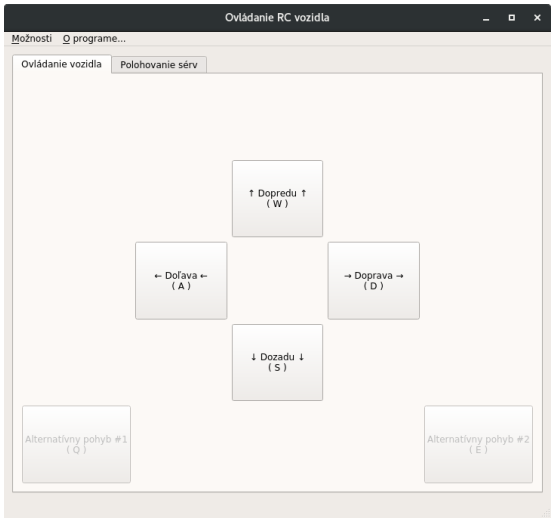
Hlavnou úlohou tejto aplikácie je tvorba riadiacich správ a ich odosielanie do riadiacej platformy prostredníctvom Bluetooth rozhrania. Tieto správy sú generované na základe používateľovych vstupov pre ovládanie vozidla, respektíve serv na ňom. Okrem toho je pre splnenie požadovanej funkčnosti vyžadované, aby aplikácia umožňovala vyhľadanie dostupných Bluetooth zariadení a pripojenie sa k zvolenému zariadeniu. Navyše je ešte žiadané, aby v nej bolo možné upravovať či už parametre serv, alebo parametre typov pohybu.

Vzhľad aplikácie

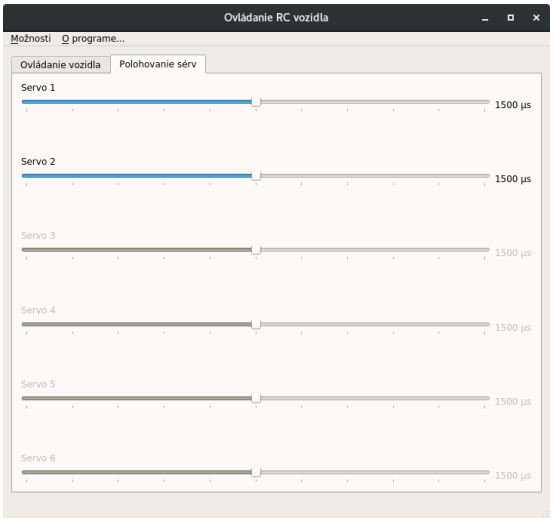
Jadrom aplikácie je hlavné okno, ktoré umožňuje buď ovládanie vozidla pomocou riadenia pohybov, alebo pomocou polohovania serv. Medzi týmito dvoma režimami je možné prepínať prostredníctvom kariet okna. Toto okno tiež obsahuje horné menu, kde je možné otvoriť okno s nastaveniami, alebo nastavenia importovať či exportovať. Ukážky rozhrania hlavného okna je možné vidieť na obrázkoch 4.4 a 4.5.

Okno s nastaveniami obsahuje tri základné karty. Na prvej karte je možné nastavovať parametre serv pripojených k Arduinu, konkrétne či je dané servo pripojené, dĺžku impulzu v μs pre vystavenie serva do stredovej polohy a rozsah impulzov po krajné hodnoty (obr. 4.6). Druhá karta slúži k nastaveniu typov pohybov, teda toho, ktoré servá tieto pohyby používajú, akú majú prioritu pohybu a či sú tieto typy pohybov povolené (obr. 4.7). Na tretej karte sa nachádza rozhranie pre vyhľadávanie dostupných Bluetooth zariadení a pripojenie sa k nim (obr. 4.8). Taktiež je tam možné nastaviť niektoré parametre prenášania správ na zariadenie, pričom sa jedná o:

- minimálny interval odosielania správ o pohybe, určený k prevencii príliš častého odosielania správ
- interval odosielania správ ak sa nemení stav ovládania, ktorý redukuje zbytočný prenos správ

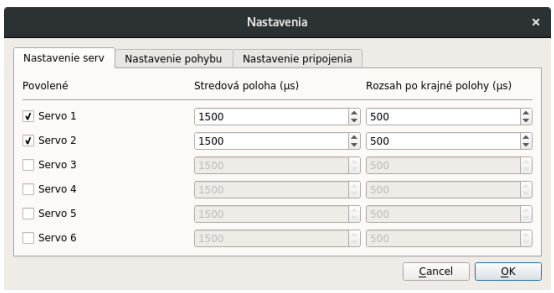


Obr. 4.4: Režim riadenia pohybov

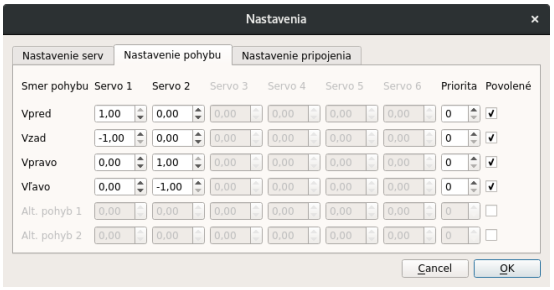


Obr. 4.5: Režim polohovania serv

- maximálnu povolenú dobu odozvy (používa sa v programe riadiacej platformy) – ak riadiaca platforma neobdrží správu o pohybe skôr, než uplynie táto doba od momentu prijatia poslednej správy, tak sa zresetujú všetky direktívy pohybu na pôvodné hodnoty (čo spôsobí zastavenie vozidla).



Obr. 4.6: Nastavenia serv



Obr. 4.7: Nastavenia pohybov

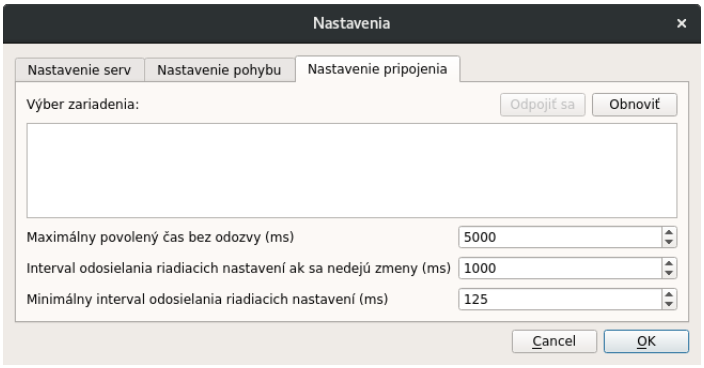
Implementácia

Aplikácia je tvorená štyrmi triedami, menovite `CarControlWindow`, `CarControlSettings`, `BluetoothAgent` a `TransferProtocol`. Ďalej ešte obsahuje hlavičkové súbory `Settings` a `TransferProtocolEnums` a súbor so zdrojovým kódom `Settings`.

`CarControlWindow` je trieda hlavného okna. Táto trieda má na starosti spracovanie vstupov z GUI a odosielanie príkazov pre pohyb do pripojeného zariadenia v požadovaných časových intervaloch, ktoré je možné upraviť v nastaveniach pripojenia.

Triedou okna nastavení je `CarControlSettings`. Táto trieda spracováva vstupy používateľa v nastaveniach, poskytuje funkcionalitu importu a exportu nastavení, volá podľa potreby metódy triedy `BluetoothAgent` a prijíma a spracováva od nej správy.

Komunikácia s Bluetooth rozhraním je vykonávaná triedou `BluetoothAgent`. Medzi úlohy tejto triedy patrí vyhľadávanie dostupných zariadení, ktoré sa vykonáva pomocou Qt Bluetooth triedy `QBluetoothDeviceDiscoveryAgent`, ďalej má táto trieda na starosti



Obr. 4.8: Nastavenia pripojenia

pripojenie sa k zariadeniu, respektíve odpojenie sa a odosielanie správ. Aktuálne pripojené zariadenie je dostupné cez vlastnosť `connectedDevice`, typu `QBluetoothSocket`.

Trieda `TransferProtocol` slúži pre tvorenie správ, ktoré zodpovedajú navrhnutému prenosovému protokolu. Pri tvorbe správ sa využíva očíslovaného dátového typu, ktorý je uložený v hlavičkovom súbore `TransferProtocolEnums`.

Nakoniec, hlavičkový súbor `Settings.h` obsahuje dátovú štruktúru, v ktorej sú uložené nastavenia a súbor so zdrojovým kódom `Settings.cpp` obsahuje len funkciu pre inicializáciu štruktúry uchovávajúcej nastavenia.

4.2.3 Program pre riadiacu platformu

Program pre riadiacu platformu – Arduino Uno – bol naprogramovaný vo vývojovom prostredí Arduino IDE. Pri tvorbe tohto programu boli použité nielen Arduino knižnice, ale aj niektoré súbory z aplikácie na počítač.

Použité knižnice

- `Arduino` – základná knižnica používaná v Arduino IDE. Je implicitnou knižnicou, teda je automaticky pridaná do všetkých zdrojových súborov, ktoré sa na Arduino IDE prekladajú. Táto knižnica poskytuje väčšinu potrebnej funkcionality, pričom mnoho funkcií v nej pochádza z knižníc jazyka C.
- `Servo` – knižnica určená k ovládaniu serv. Vďaka tejto knižnici je možné k Arduino doskám pripojiť servá aj na tie digitálne piny, ktoré nemajú bežne podporu PWM. Umožňuje teda pripojenie až 12 serv na dosku Arduino Uno. Z tejto knižnice sú používané štyri funkcie:
  - `attach()` – pripojí servo na digitálny pin určený parametrom funkcie. Ďalšie dva voliteľné parametre môžu určiť minimálnu a maximálnu dĺžku impulzu pre servo, avšak to v tomto programe nie je využívané.
  - `detach()` – odpojí servo z aktuálne pripojeného pinu.
  - `attached()` – vráti boolean hodnotu, či je servo pripojené k nejakému pinu. Používa sa k overovaniu, či sú zapnuté servá pripojené k patričným pinom alebo nie.

  - `writeMicroseconds()` – vygeneruje na pine, ku ktorému je pripojené servo, riadiaci impulz o požadovanej dĺžke v μs.

- `EEPROM` – poskytuje funkcie pre prácu s EEPROM pamäťou na doske. Je používaná pre ukladanie nastavení do pamäte, aby ich nebolo nutné opätovne prenášať. Používajú sa dve funkcie z tejto knižnice:
  - `put()` – zapíše na zadanú adresu EEPROM pamäte dáta z premennej danej druhým parametrom funkcie. Je schopná zapísať ako jednoduché typy, tak aj vlastné dátové štruktúry.
  - `get()` – načíta z adresy v EEPROM pamäti dáta do premennej danej druhým parametrom funkcie.

Okrem knižníc dostupných prostredníctvom Arduino IDE som v programe použil aj nasledujúce súbory z aplikácie pre počítač:

- `TransferProtocolEnums.h` – obsahuje dva očíslované dátové typy, ktoré sa používajú v správach pre identifikáciu blokov. Pre správnu komunikáciu je vyžadované, aby sa tieto označenia zhodovali s tými, ktoré sú na odosielateľovej strane.

- `Settings.h` a `Settings.cpp` – jedná sa o hlavičkový súbor ktorý obsahuje štruktúru pre ukladanie nastavení a súbor so zdrojovým kódom, ktorý obsahuje funkciu pre inicializáciu nastavení. Zo štruktúry nastavení boli odstránené dve premenné, ktoré mali význam len na strane odosielateľa.

Riadiace premenné a direktívy

Na začiatku zdrojového súboru s programom sú definované premenné a konštanty, ktorými je možné modifikovať beh programu. Medzi dostupné nastavenia patrí napríklad to, či sa majú zapisovať nastavenia do pamäte EEPROM a adresa EEPROM, na ktorú sa majú nastavenia zapisovať. Ďalej je dostupná štruktúra obsahujúca nastavenia, pole objektov serv, prostredníctvom ktorých sa generujú riadiace impulzy, či pole čísel pinov, ku ktorým sa servá majú pripájať. Taktiež sú tu definované premenné pre udržiavanie času, kedy bola prijatá posledná správa s pokynmi pre riadenie alebo polia, do ktorých sa ukladajú informácie o prijatých pokynoch pre riadenie vozidla.

Funkcia `setup()`

Táto funkcia, ako bolo už spomenuté v sekcii o vývoji programov pre platformu Arduino, sa zavolá len raz po spustení programu. Nastavuje sa v nej rýchlosť prenosu pre sériové rozhranie, pričom tá je určená konštantou `SERIAL_BAUD_RATE`. Ďalej sa načítajú nastavenia z pamäte EEPROM, ak sú uložené, inak sa inicializuje štruktúra nastavení a v prípade, že je definovaná direktíva `WRITE_SETTINGS_TO_EEPROM`, tak sa uložia nastavenia do EEPROM. Posledným krokom v tejto sekcii je pripojenie serv podľa nastavení.

Funkcia `loop()`

Vo funkcii `loop()` prebieha zapisovanie hodnôt z globálnych premenných pre ukladanie riadiacich pokynov do objektov serv, ktoré následne generujú riadiace impulzy. V prípade, že je aktívne riadenie pohybov, sa volá pre každé servo funkcia, ktorá vypočíta požadovanú

polohu serva. Taktiež sa v tejto funkcii kontroluje, či neuplynula doba väčšia než je doba maximálnej povolenej odozvy. V prípade, že tento jav nastal, sa vynulujú globálne premenné obsahujúce pokyny pre riadenie vozidla, čím by sa malo vozidlo zastaviť (za predpokladu, že nie sú chybne dané nastavenia).

Spracovanie vstupu

Spracovávanie prijímaných dát prebieha vo funkcii `serialEvent()`, ktorá je bežne volaná za každým volaním funkcie `loop()`, ak sú na sériovom rozhraní dostupné dáta. V tejto funkcii sa používa sada statických premenných, ktoré udržiadavajú kontext spracovania aktuálnej správy. Takto je to implementované vzhľadom na to, že je funkcia `serialEvent()` volaná aj keď je na sériovom rozhraní dostupný jediný znak, ktorý tvorí len časť správy.

Kvôli tomuto spôsobu spracovávania je však problematické zotavovanie sa z nečakaných stavov, akými je napríklad neznámy typ správy či nastavení, alebo neočakávané prijaté ID serva či typu pohybu. Nečakané prijaté ID serva či pohybu je možné vyriešiť ignorovaním danej sekcie, ktorá patrí neznámemu servu. Problémom je však to, ak nesúhlasí konštanta určujúca počet serv v aplikácii na počítači a na platforme, keďže sa na základe tohto čísla očakáva určitý počet bajtov pri prenose nastavení o typoch pohybu.

Ďalšou komplikáciou využívania tejto funkcie je to, že sa volá len raz po každom skončení funkcie `loop()`. Spracovanie jediného znaku pri zavolaní tejto funkcie by bolo problémom, ak by funkcia `loop()` vyvolávala nejaké čakanie, čím by sa zabránilo čítaniu znakov zo sériového rozhrania. Tento problém je však v mojom programe riešený tak, že ak je definovaná direktíva `ALLOW_CYCLIC_SERIALEVENT_CALLING`, tak sa opäť zavolá funkcia `serialEvent()` na konci jej tela, ak sú na sériovom rozhraní dostupné znaky na prečítanie. Týmto sa prakticky prečítajú všetky dostupné znaky.

Priorita typov pohybu

Vzhľadom na to, že viacero typov pohybu môže používať to isté servo, tak vzniká problém, ak sa majú súčasne vykonávať rôžne typy pohybu, ktoré používajú to isté servo. Za účelom riešenia tohto problému bolá pridaná priorita typov pohybu. V mojom programe slúži pre výpočet odchýlky dĺžky impulzu od stredovej polohy funkcia `getServoOffset()`. Táto funkcia pre dané servo zistí, či sa používa pre aktívne typy pohybov a v prípade konfliktov vypočíta výslednú polohu. Pri určovaní, ktorý pohyb má prednosť pre dané servo, sa používa:

1. Priorita pohybu – poloha sa vypočíta na základe aktívneho typu pohybu s najvyššou prioritou.

2. Vystavenie serva typom pohybu – používa sa v prípade, ak majú aktívne typy pohybu rovnakú prioritu. Vyberie sa ten typ pohybu, ktorý má najvyššiu absolútnu hodnotu vystavenia serva.

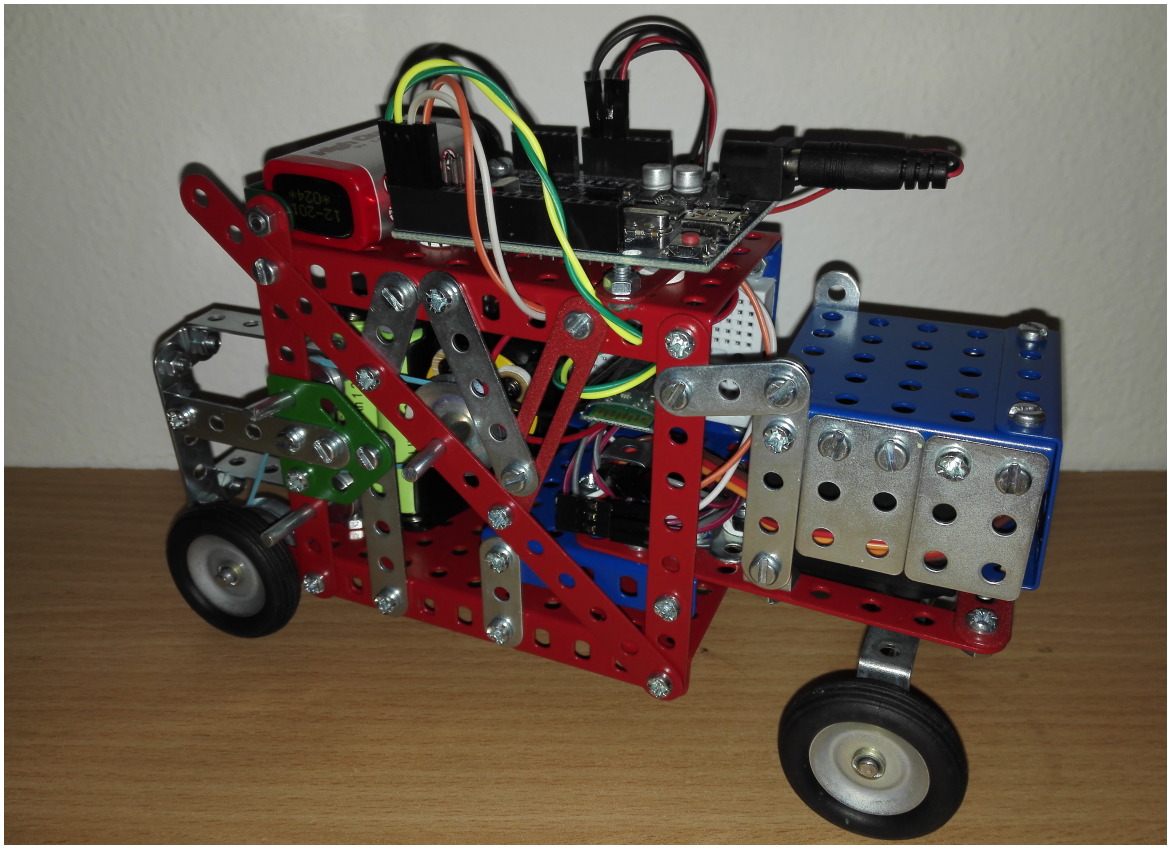
3. Poradie uloženia typov pohybu v nastaveniach – použitý len v prípade, ak konfliktné aktívne typy pohybu majú rovnakú prioritu aj vystavenie serva. Výsledná poloha ramena serva sa vypočíta podľa prvého typu pohybu s najvyššou výchylkou, ktorý toto servo využíva.

Výsledné zariadenie

Po skonštruovaní vozidla, zapojení elektronických súčiastok podľa uvedenej schémy a nahratí programu do riadiacej platformy disponujeme vlastným modelom vozidla, ktoré je možné bezdrôtovo ovládať prostredníctvom Bluetooth rozhrania. Toto vozidlo je ovládateľné pomocou vytvorenej počítačovej aplikácie, pričom je možné aj do značnej miery meniť konfiguráciu vozidla. Vytvorené vozidlo je možné vidieť na obrázku 5.1.

Kapitola 5

Testovanie



Obr. 5.1: Vzhľad dokončeného vozidla

Rozmery vozidla

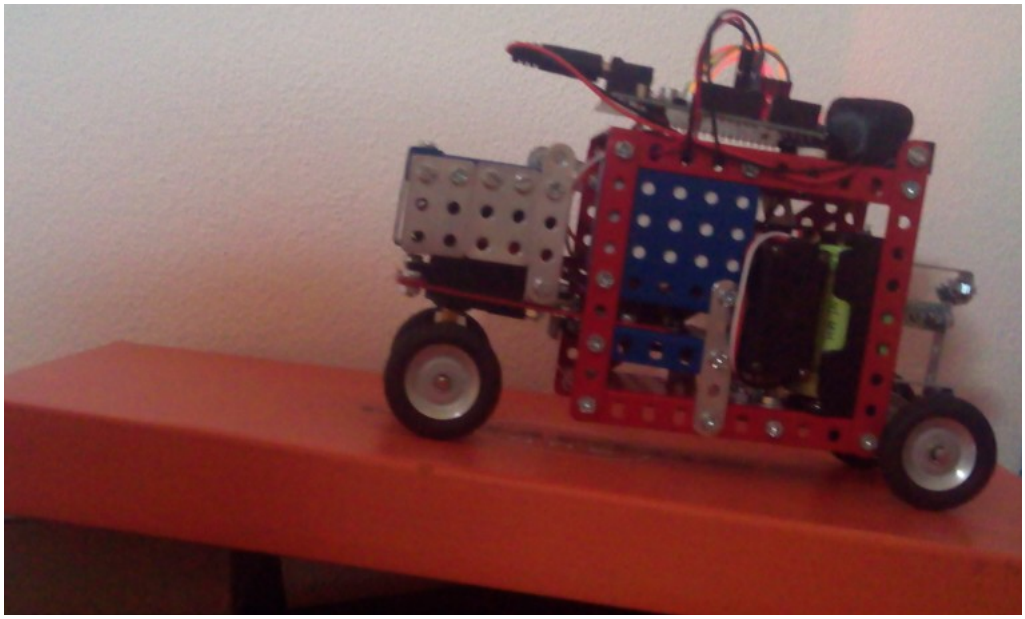
- Dĺžka: 200 mm
- Šírka: 82 mm
- Výška 127 mm (bez vodičov zapojených do Arduina)

Ovládanie vozidla

Testovanie prebiehalo pripojením sa k HC-05 modulu na vozidle prostredníctvom Bluetooth, následne odosielaním pokynov pre riadenie vozidla a skúšaním zmien v nastaveniach a sledovaním reakcií správania sa vozidla na tieto zmeny. Tieto pokusy preukázali, že pripojenie sa k zariadeniu a jeho ovládanie funguje správne, rovnako ako aj prenos nastavení a správanie sa vozidla zodpovedá očakávaniam.

Nedostatky sú však badateľné na mechanickej stránke vozidla. Náhon zadnej nápravy prostredníctvom remeňu (gumičky) je hlavným problémom. Vadou tohto náhonu je to, že remeň má tendenciu prekĺzavať, ak má nedostatočnú treciu silu s kolesami po ktorých je vedený. To sa dá riešiť silnejším napnutím tohto remeňu, avšak následne vzniká ten problém, že sa časom gumička rýchlejšie opotrebuje a môže sa roztrhnúť. Dôsledkom týchto problémov je že vozidlo nemá dostatočný ťah na to, aby bola možná jazda pod strmejším stúpaním.

Experimentálne som zistil, že je vozidlo schopné stúpania po rovine naklonenej o 15°, avšak pri zvýšení náklonu roviny na 20° už vozidlo nebolo schopné pokračovať v jazde, pretože remeň prekĺzaval na kolesách. Pokus o stúpanie na rovine naklonenej o 20° je možné vidieť na obrázku 5.2.



Obr. 5.2: Skonštruované vozidlo nedokáže vystúpať 20° sklon

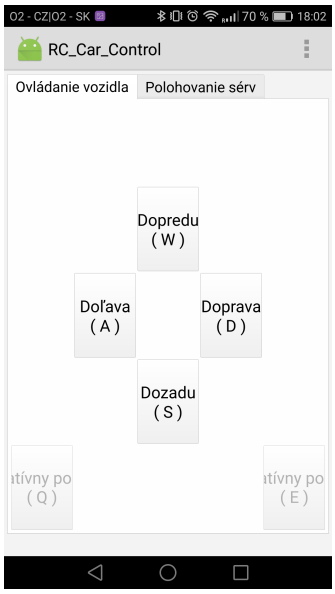
Ďalším menším nedostatkom mechanickej stránky vozidla je to, že je predná náprava pripojená priamo na servo. Za dôsledok to má to, že pri zatáčaní sa vozidlo mierne nakloní do opačnej strany, než sa zatáča. Taktiež sa pri vytáčaní musia kolesa vystaviť do danej pozície, čím sa jedno koleso prednej nápravy vždy musí hýbať do opačnej strany, než je aktuálny smer pohybu.

Multiplatformnosť Qt

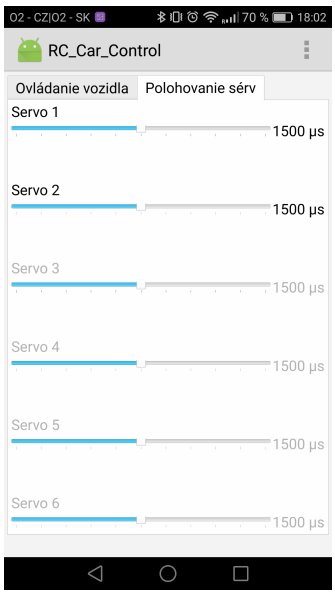
Vzhľadom na to, že Qt je multiplatformé a vytvorená aplikácia by mala byť preložiteľná na rôzne architektúry a operačné systémy, som sa v rámci testov rozhodol, že skúsim preložiť

aplikáciu pre počítač aj ako aplikáciu pre mobilný telefón. Táto aplikácia bola síce väčšinou funkčná, ale trpela určitými problémami, akými boli napríklad určité chyby odosielania nastavení, alebo grafické používateľské rozhranie. Na grafickom rozhraní bolo evidentné, že nebolo robené pre mobilné telefóny, ale pre osobné počítače. Okrem toho, že na ňom nefungoval multitouch, boli niektoré prvky viditeľné len čiastočne, iné boli veľmi limitovane ovládateľné. Hlavným problémom bolo však zobrazenie okna s nastaveniami, na ktorom nebolo možné vidieť tlačidlá. Základ aplikácie však bol funkčný, teda po úpravách GUI by bolo možné používať aplikáciu aj pre mobilné telefóny.

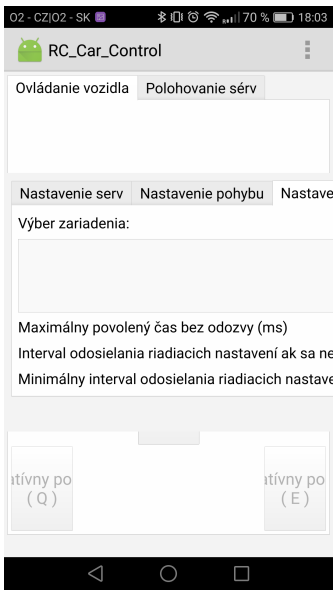
Na obrázku 5.3 je možné vidieť, že sa nezobrazuje všetok text v tlačidlách pri vertikálnom zobrazení. Ďalej na obrázku 5.4 je síce rozloženie prvkov v poriadku, avšak samotné posuvníky sú príliš malé na to, aby s nimi bolo možné jednoducho manipulovať. Otvorenie nastavení pri vertikálnom zobrazení je možné vidieť na obrázku 5.5, kde nie je možné vidieť takmer polovicu daného okna. Tiež je možné vidieť, že sa okno nastavení otvorí centrované na vertikálny stred obrazovky, ale nie je možné s ním manipulovať. Ďalším problémom je to, že nie je možné otvoriť okno zobrazujúce informácie o aplikácii, ako to je možné spraviť na počítači.



Obr. 5.3: Ovládanie pohybu

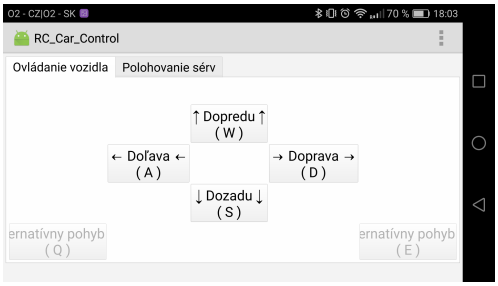


Obr. 5.4: Ovládanie serv

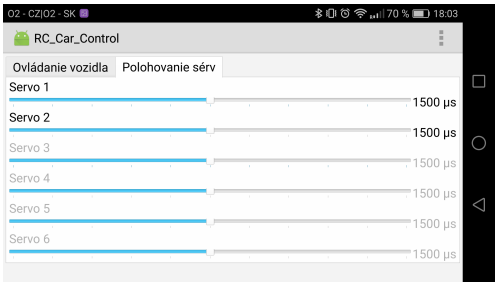


Obr. 5.5: Nastavenia

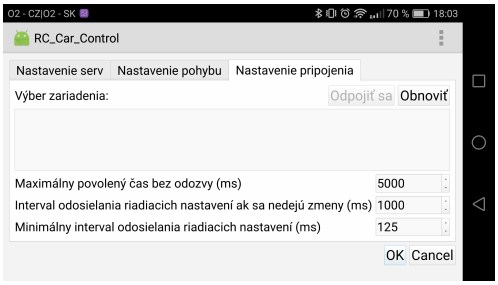
Niektoré zo spomenutých problémov sú buď čiastočne, alebo úplne neprítomné pri horizontálnom zobrazení aplikácie. Pri ovládaní pohybov je značne redukované množstvo textu na tlačidlách, ktorý nie je možné vidieť (obr. 5.6). U polohovania serv nie je badateľná žiadna podstatná zmena (obr. 5.7). Konkrétne je však okno nastavení možné vidieť celé pri jeho otvorení v horizontálnom zobrazení (viď obrázok 5.8). Problém je však v tom, že pri zmene orientácie zostáva pozícia okna rovnaká, teda ak sa okno nastavení otvorí vo vertikálnom zobrazení a následne sa prejde na horizontálne zobrazenie, tak je okno posunuté nižšie a teda opäť nie je možné vidieť spodnú časť okna. Tento jav je viditeľný na obrázku 5.9.



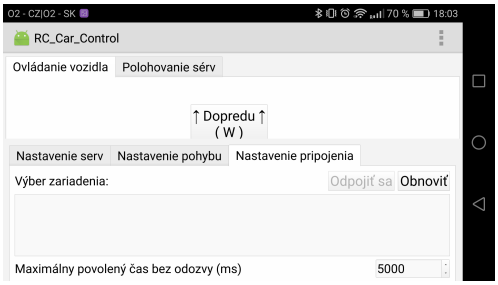
Obr. 5.6: Ovládanie pohybov v horizontálnom zobrazení



Obr. 5.7: Ovládanie serv v horizontálnom zobrazení



Obr. 5.8: Okno nastavení pri horizontálnom zobrazení



Obr. 5.9: Okno nastavení po prechode z vertikálneho zobrazenia na horizontálne

Kapitola 6

Záver

Úlohou tejto práce bolo navrhnutie a skonštruovanie modelu vozidla riadeného modelárskymi servami, ktorý je možné ovládať pomocou počítača. Tento cieľ sa mi podarilo úspešne dosiahnuť. V rámci tejto práce sa mi podarilo vytvoriť jednoduché vozidlo, ktoré je možné ovládať prostredníctvom počítača. Jedná sa o model automobilu, poskladaný zo stavebnice Merkur, osadený servami a riadený platformou Arduino Uno. Tento model je tiež osadený Bluetooth modulom, ktorý mu umožňuje prijímať bezdrôtovo pokyny z počítača. Okrem skonštruovaného vozidla boli v tejto práci vytvorené aj ďalšie komponenty – jedným je protokol, pomocou ktorého sa prenášajú príkazy z počítača na riadiacu platformu. Ďalej bol vytvorený program pre riadiacu platformu, ktorý spracováva prijaté správy, prenášané pomocou spomenutého protokolu a generuje riadiace impulzy pre servá. Ako posledná časť bola vytvorená aplikácia, ktorá umožňuje toto vozidlo bezdrôtovo ovládať pomocou Bluetooth. Tieto časti spolu tvoria celok, ktorým je vozidlo ovládateľné pomocou počítača. Za pomoci vytvorenej aplikácie je možné jednoducho vytvoriť vozidlo, ktoré bude podporovať až šesť typov pohybu a môže byť osadené až šiestimi servami, pričom je možné jednoducho tieto počty navýšiť menšími úpravami v aplikácii.

Venovanie sa tejto práci rozšírilo moje znalosti o viacerých oblastiach, ktorými sú ovládanie serv, typy modulácií, prenos dát prostredníctvom Bluetooth či jednoprocesorové minipočítače. Vďaka tejto práci som mal taktiež možnosť nadobudnuté znalosti vyskúšať pri tvorbe vozidla a navrhovaní spôsobu jeho ovládania.

Ako pokračovanie na projekte by bolo možné napraviť mechanické nedostatky vozidla, ktoré boli vytýčené v kapitole o testovaní. Ďalej je možné rozšíriť vozidlo o ďalšie súčiastky, akými sú napríklad senzory, a doplniť program riadiacej platformy o podporu týchto súčiastok; respektíve obecne rozšíriť program riadiacej platformy o ďalšiu funkčnosť. Rozšírenie aplikácie pre počítač by bolo možné viacerými spôsobmi: navyšením podporovaného počtu serv na 12, čo je maximum podporované na doskách Arduino Uno; reagovaním na správy prijímané od zariadenia, ktoré by mohlo odosielať napríklad po pridaní senzorov; upraviť podporovaný počet typov pohybu tak, aby bolo možné zadať premenný počet; pridať u serv možnosť výberu digitálneho pinu, ku ktorému budú pripojené; či pridať možnosť iného pripojenia, než je Bluetooth rozhranie.

Literatúra

[1] *About Qt – Qt Wiki*. [Online; navštívené 29.04.2017].
URL http://wiki.qt.io/About_Qt

[2] *Arduino – Home*. [Online; navštívené 07.02.2017].
URL https://www.arduino.cc/

[3] *Arduino – Wikipedia*. [Online; navštívené 07.02.2017].
URL https://en.wikipedia.org/wiki/Arduino

[4] *Bastlení – jak funguje modelářské servo*. [Online; navštívené 22.01.2017].
URL http://vlastikd.webz.cz/bastl/serva.htm

[5] *Bluetooth Technology Website*. [Online; navštívené 16.02.2017].
URL https://www.bluetooth.com

[6] *Bluetooth – Wikipedia*. [Online; navštívené 16.02.2017].
URL https://en.wikipedia.org/wiki/Bluetooth

[7] *Introduction to Servo Motors*. [Online; navštívené 23.01.2017].
URL http://www.sciencebuddies.org/science-fair-projects/project_ideas/Robotics_ServoMotors.shtml

[8] *Midwest RC Society*. [Online; navštívené 24.01.2017].
URL http://www.theampeer.org/midwest/articles/servos.html

[9] *Modelářská serva - základní informace*. [Online; navštívené 22.01.2017].
URL http://www.pojezdy.eu/view.php?cisloclanku=2011070004

[10] *Modulation – Wikipedia*. [Online; navštívené 28.01.2017].
URL https://en.wikipedia.org/wiki/Modulation

[11] *PELIKANDANIEL.COM – Serva*. [Online; navštívené 23.01.2017].
URL http://www.pelikandaniel.com/?sec=page&id=22

[12] *Pulse-width modulation – Wikipedia*. [Online; navštívené 29.01.2017].
URL https://en.wikipedia.org/wiki/Pulse-width_modulation

[13] *Raspberry Pi – Teach, Learn and Make with Raspberry Pi*. [Online; navštívené 05.01.2017].
URL https://www.raspberrypi.org/

[14] *RIi Hub – eLinux.org*. [Online; navštívené 06.01.2017].
URL http://elinux.org/RPi_Hub

[15] *Servo Motors | Types, Properties, Control.* [Online; navštívené 22.01.2017].
URL http://www.robotiksistem.com/servo_motor_types_properties.html

[16] *Understanding PWM – EBLDC.COM.* [Online; navštívené 29.01.2017].
URL http://ebldc.com/?p=48

[17] *Understanding RC Servos – Digital, Analog, Coreless, Brushless.* [Online; navštívené 24.01.2017].
URL http://www.rchelicopterfun.com/rc-servos.html

[18] *What is an Arduino? – learn.sparkfun.com.* [Online; navštívené 07.02.2017].
URL https://learn.sparkfun.com/tutorials/what-is-an-arduino

[19] *What is a serial interface?* [Online; navštívené 15.02.2017].
URL http://www3.nd.edu/~lemmon/courses/ee224/web-manual/web-manual/lab9/node4.html